WP JobRoom

Comprehensive Documentation Guide

A Swiss Job Matching Platform for WordPress
Connecting Applicants and Companies

Version:	0.27.0
Author:	Marco Graetsch
PHP:	8.3+
WordPress:	6.0+
License:	Proprietary

Generated on March 07, 2026
src.bundespruefstelle.ch/magdev/wp-jobroom

About This Document
This guide combines all WP JobRoom documentation into a single reference. It covers everything from end-user features to technical architecture and deployment.

Chapters 1–2 are written for non-technical users (applicants, companies, site administrators).
Chapters 3–7 are aimed at developers, designers, DevOps engineers, and QA testers.

Contents

1 User Guide

1.1 Getting Started

1.1.1 Registering as an Applicant

1. Navigate to `/register/applicant/` on the website
2. Fill in the registration form:
   - Email address (must be unique)
   - Password (minimum requirements may apply)
   - First and last name
3. Submit the form
4. Check your email for a verification link
5. Click the verification link to activate your account

1.1.2 Registering as a Company

1. Navigate to `/register/company/` on the website
2. Fill in the registration form:
   - Company email address
   - Password
   - Company name
   - Contact person details
3. Submit the form
4. Verify your email address
5. Choose a subscription plan (optional - free tier available)

1.1.3 Email Verification

After registration, you will receive an email with a verification link. This link:

- Is valid for 24 hours
- Must be clicked to activate your account
- Can be resent from the login page if expired

1.1.4 Logging In

1. Navigate to `/login/` or the WordPress login page
2. Enter your email and password
3. If two-factor authentication is enabled, enter your verification code
4. You will be redirected to your dashboard

1.1.5 Resetting Your Password

1. Click “Forgot Password” on the login page
2. Enter your email address
3. Check your email for reset instructions
4. Click the reset link and enter a new password

1.2 Profile Management

1.2.1 Applicant Profile

Your applicant profile is your digital CV. It includes:

Personal Information

- Full name and contact details
- Profile picture (uploaded as featured image)
- Location (canton and city)
- Languages spoken with proficiency levels

Professional Details

- **Work Experience**: Add multiple positions with:
  - Company name and job title
  - Start and end dates
  - Description of responsibilities
  - **Achievements**: One or more bullet-point accomplishments per position
- **Education**: Add qualifications with:
  - Institution name
  - Degree/certification
  - Field of study
  - Graduation date
- **Skills**: Select skills from the skill catalog and rate your proficiency (1-5):
  - 1 = Beginner
  - 2 = Basic
  - 3 = Intermediate
  - 4 = Advanced
  - 5 = Expert

Certificates & Documents

Upload certificates, diplomas, and other documents to your profile:

1. Navigate to your profile edit page
2. Scroll to the “Certificates & Documents” section
3. Click “Add Certificate” or drag and drop a file onto the upload zone
4. Fill in certificate details:
   - **Title**: Certificate name (e.g., “AWS Solutions Architect”)
   - **Issuer**: Issuing authority (e.g., “Amazon Web Services”)
   - **Issued Date**: When the certificate was issued

  - **Expiry Date**: When it expires (leave empty for no expiry)
5. Optionally link the certificate to a work experience or education entry
6. Click “Save Changes”

Supported file types: PDF, JPEG, PNG, WebP (max 10 MB per file)

Certificates are visible on your public profile when the “Documents” privacy section is set to public or registered.

Personal References

Add professional references so matched companies can contact them directly:

1. Navigate to your profile edit page
2. Scroll to the “Personal References” section
3. Click “Add Reference”
4. Fill in the reference details:
   - **Name**: Full name of the reference (required)
   - **Company**: Organisation where they work
   - **Relationship**: How they know you (e.g. “Former Manager”, “Colleague”)
   - **Email**: Contact email address
   - **Phone**: Contact phone number
5. Repeat for additional references
6. Click “Save Changes”

References are controlled by the **References** privacy setting, which defaults to “Matched” — they are only shown to companies that have a matching score with your profile. You can change this in the “Privacy Settings” section.

Tip: References can also be imported automatically when you upload a CV that contains a “References” or “Referenzen” section.

Availability Settings

- **Status**: Immediately available, available from date, or not looking
- **Employment Type**: Full-time, part-time, freelance, internship
- **Workload**: Minimum and maximum percentage (e.g., 80-100%)
- **Desired Locations**: Cantons where you want to work
- **Remote Work**: Preference for remote, hybrid, or on-site

Swiss-Specific Details

- Work permit type (Swiss citizen, B permit, C permit, EU/EFTA, etc.)
- Cross-border worker (Grenzgänger) status

1.2.2 Editing Your Profile

1. Navigate to your profile page
2. Click “Edit Profile”
3. Modify any section
4. Click “Save Changes”

1.2.3 Company Profile

Company profiles showcase your organization to potential applicants:

Company Information

- Company name and logo
- Website URL
- Industry classification
- Company size category
- Canton and address

Culture and Benefits

Select from 26 predefined benefits including:

- Home office options
- Flexible working hours
- Health insurance
- Pension plan (BVG+)
- Childcare support
- Professional development
- And more…

Contact Persons

Add one or more contact persons to your company profile. Each contact includes:

- Name and position
- Email and phone number

The number of contact persons depends on your subscription tier (Free/Basic: 1, Professional: up to 5, Enterprise: unlimited). When creating a job offer, you can select which contact person is displayed on that listing. The first contact is used as the default.

1.2.4 Profile Privacy Settings

Control who can see different sections of your profile:

Privacy Level	Who Can View
Public	Anyone, including search engines
Registered	Only logged-in users
Matched	Only users with a match score above threshold
Private	Only you and administrators

Controllable sections and their defaults:

Section	Default
Basic Info	Public
Contact	Matched
Work Experience	Registered
Education	Registered
Skills	Public
Languages	Public
Documents	Registered
References	Matched

To change privacy settings:

1. Go to your profile edit page
2. Find the "Privacy Settings" section
3. Set visibility for each profile section
4. Save changes

1.3 Searching

1.3.1 Finding Jobs (For Applicants)

Navigate to `/search/jobs/` to search for job offers.

Filters Available

- **Keywords**: Search in title and description
- **Location**: Filter by canton or remote options
- **Employment Type**: Full-time, part-time, freelance
- **Workload**: Percentage range
- **Skills**: Filter by required skills
- **Industry**: Filter by industry sector
- **Salary Range**: Minimum expected salary
- **Position Type**: Permanent, temporary, try & hire

Understanding Match Scores

Each job shows a match score (0-100%) based on:

- Skills match (how well your skills match requirements)
- Experience level alignment
- Location compatibility
- Employment type match
- Availability match
- Work permit compatibility

1.3.2 Finding Applicants (For Companies)

Navigate to `/search/applicants/` to find potential candidates.

Filters Available

- **Keywords**: Search in profile content
- **Skills**: Required skills with minimum proficiency
- **Location**: Canton or remote availability
- **Experience Level**: Years of experience
- **Employment Type**: Full-time, part-time preference
- **Availability**: Immediately available, specific date
- **Work Permit**: Filter by permit type

1.3.3 Finding Companies

Navigate to `/search/companies/` to browse company profiles.

Filters Available

- **Industry**: Company sector
- **Location**: Canton
- **Company Size**: Small, medium, large
- **Keywords**: Search company descriptions

1.3.4 Saving Searches

Save frequently used searches for quick access and notifications:

1. Perform a search with your desired filters
2. Click "Save Search"
3. Enter a name for the search
4. Choose notification frequency (daily, weekly, or none)
5. Access saved searches from your dashboard

1.4 Job Applications

1.4.1 Applying for a Job (Applicants)

1. Find a job offer through search or your matches
2. Click "Apply" or navigate to `/job-offers/{slug}/apply/`
3. Review the job requirements
4. Write a cover letter (optional but recommended)
5. Submit your application

Your full profile will be shared with the employer.

1.4.2 Application Status Workflow

Status	Meaning
Pending	Application submitted, awaiting review
Reviewing	Employer is reviewing your application
Shortlisted	You made the shortlist
Interview	Interview scheduled or completed
Offered	Job offer extended
Hired	You got the job!
Rejected	Application not successful
Withdrawn	You withdrew your application

1.4.3 Tracking Applications

View your application history at `/my-applications/`:

- See all submitted applications
- Check current status
- View employer messages
- Withdraw applications (if not in final status)

1.4.4 Managing Applications (Companies)

Access received applications at `/applications/`:

- View all applications for your job offers
- Filter by job offer or status
- Update application status
- Send messages to applicants
- Compare applicants side by side

1.5 Messaging

1.5.1 Inbox

Access your messages at `/messages/`:

- View all conversations
- See unread message indicators
- Sort by date or read status

1.5.2 Sending Messages

1. Navigate to `/messages/compose/` or click “Message” on a profile

2. Select recipient (if not pre-selected)
3. Write your message
4. Click “Send”

1.5.3 Conversation Threads

All messages with a user are grouped in a conversation thread:

- View full message history
- Reply directly in the thread
- Messages appear in chronological order

1.5.4 Message Notifications

Control message notifications in your account settings:

- Email notification for each new message
- Include in daily/weekly digest only
- No email notifications (in-app only)

1.6 Account Settings

1.6.1 Dashboard

Access your dashboard at `/dashboard/`:

For Applicants:

- Profile views over time
- Search appearances
- Application statistics
- Matching job recommendations

For Companies:

- Job offer performance
- Application statistics
- Profile views
- Matching applicant recommendations

1.6.2 Account Settings

All account settings are available on a single page at `/account/settings/` with sidebar navigation:

1.6.3 Notification Preferences

Configure notifications at `/account/settings/#section-notifications`:

Notification Type	Options
New Messages	Immediate, digest, or none
Application Received	Immediate, digest, or none
Application Status	Immediate, digest, or none
Matching Digest	Daily, weekly, or none

1.6.4 Security Settings

Access security options at `/account/settings/#section-security`:

Two-Factor Authentication (2FA)

Add an extra layer of security to your account:

1. Click "Enable 2FA"
2. Scan the QR code with an authenticator app
3. Enter the verification code
4. Save your backup codes securely

Backup Codes

If you lose access to your authenticator:

1. Use one of your 10 backup codes
2. Each code can only be used once
3. Generate new codes after using them

1.6.5 API Keys (For Developers)

Generate API keys at `/account/settings/#section-api-keys` for:

- Building integrations
- Automating workflows
- Accessing data programmatically

Important: Keep your API keys secure. You can only view the full key once when created.

1.6.6 Invitations

When the platform is in invite-only mode, registration requires a valid invite code. Registered users can invite others from `/account/invites/`:

1. Enter the email address of the person you want to invite
2. Optionally add a personal note
3. Click "Send Invitation"

The recipient receives an email with a registration link. You can see the status of your invitations (pending, redeemed, expired) on the same page. The number of invitations you can send is limited by your account settings.

You can also join via a direct link (`/join/{code}/`) shared by another user or administrator.

1.6.7 Data Export (GDPR)

Request a copy of all your data at `/account/data-export/`:

1. Click "Request Data Export"
2. Wait for the export to be generated
3. Download your data package (JSON format)

Exported data includes:

- Profile information
- Applications
- Messages
- Saved searches
- Activity history

1.6.8 Cookie Settings

Manage your cookie preferences:

- **Essential Cookies**: Required for site functionality (cannot be disabled)
- **Functional Cookies**: Remember your preferences
- **Analytics Cookies**: Help us improve the service
- **Marketing Cookies**: Personalized content and ads

Access cookie settings via the cookie banner or the `[jr_cookie_settings]` link.

1.7 Subscription Plans (For Companies)

Companies can choose from different subscription tiers:

Feature	Free	Basic	Professional	Enterprise
Job Postings	1	3	10	Unlimited
Applicant Visibility	Limited	Standard	Enhanced	Full
Analytics	Basic	Standard	Advanced	Full
Support	Community	Email	Priority	Dedicated

1.7.1 Managing Your Subscription

Access subscription management at `/subscription/`:

- View current plan and status
- Upgrade or downgrade plans
- View billing history
- Cancel subscription

1.7.2 Invoices

Access your invoices at `/invoices/`:

- Download PDF invoices
- View payment history
- Check upcoming charges

1.8 Support Staff Dashboard

Users assigned the `jr_supporter` role have access to a dedicated ticket queue dashboard at `/dashboard/`. The dashboard is automatically displayed when a support staff member logs in and visits the `/dashboard/` URL — no additional configuration is required.

1.8.1 Dashboard Sections

Stats Bar — Four quick-view counters at the top of the page: Open, In Progress, Waiting (for customer), and SLA Overdue (shown in red when SLA deadlines have been missed).

My Assigned Tickets — A table listing all tickets currently assigned to the support worker. Each row shows priority badge (colour-coded), subject (link to ticket detail), status badge, creation date, and SLA breach indicator. Rows with missed SLA deadlines are highlighted in red.

Unassigned Tickets — A table of tickets not yet assigned to any support worker. Each row includes a **View** button to open the ticket detail page in WP Admin where the ticket can be claimed and worked on.

Quick Links — Three action cards: Full Queue in WP Admin (complete ticket management interface), Account Settings, and Logout.

1.8.2 Navigation Menu

The profile dropdown in the navigation bar shows a simplified menu for support staff:

- Dashboard
- Ticket Queue (WP Admin)
- Account Settings
- Logout

1.9 Support Tickets

The built-in helpdesk lets you create support tickets and track their resolution.

1.9.1 Creating a Ticket

1. Navigate to `/support/new/` or click **New Ticket** from the support page
2. Fill in the form:
   - **Subject**: Brief description of your issue
   - **Category**: Select the most relevant category
   - **Priority**: Available options depend on your subscription tier
   - **Message**: Detailed description of your issue
3. Click **Submit Ticket**

1.9.2 Viewing Your Tickets

Access your ticket list at `/support/`:

- Filter tickets by status or category
- See ticket statistics (open, resolved, total)
- Click a ticket to view its full conversation

1.9.3 Replying to a Ticket

On the ticket detail page:

1. Type your reply in the message box
2. Click **Send Reply**
3. Your reply is visible to both you and the support staff

When you reply to a ticket that is in "Waiting for Customer" or "Resolved" status, it automatically reopens.

1.9.4 Ticket Statuses

Status	Meaning
New	Just created, awaiting staff review
Open	Acknowledged by staff
In Progress	Staff is actively working on it
On Hold	Temporarily paused
Waiting for Customer	Staff needs more information from you
Resolved	Issue has been addressed
Closed	Ticket is finalized

1.9.5 Priority Levels

Available priorities depend on your subscription tier:

Tier	Available Priorities
Free / Basic	Low, Normal
Professional	Low, Normal, High
Enterprise	Low, Normal, High, Urgent

Higher priorities receive shorter SLA response times.

1.9.6 Satisfaction Rating

After a ticket is resolved or closed, you can rate your experience (1-5 stars). This helps improve support quality.

1.9.7 Email Notifications

You receive email notifications when:

- Your ticket status changes
- Staff replies to your ticket
- Your SLA deadline is approaching (if applicable)

You can also reply to tickets via email if the platform has reply-by-email configured.

1.9.8 Enterprise Dedicated Support

Enterprise-tier companies receive dedicated support with:

- Faster SLA response times
- Access to all priority levels including Urgent
- Dedicated support channel

1.10 Need Help?

If you need assistance:

1. Check this user guide for answers
2. Contact the site administrator
3. Use the messaging system to reach support

---

2 Administrator Guide

2.1 Installation

2.1.1 Requirements

- WordPress 6.0 or higher
- PHP 8.3 or higher
- MySQL 5.7+ or MariaDB 10.3+
- HTTPS enabled (required for security features)

2.1.2 Installation Steps

1. Download the plugin ZIP file
2. In WordPress Admin, go to **Plugins** > **Add New** > **Upload Plugin**
3. Select the ZIP file and click **Install Now**
4. Click **Activate Plugin**

2.1.3 Post-Installation

After activation:

1. Navigate to **JobRoom** > **Settings** to configure the plugin
2. Go to **Settings** > **Permalinks** and click "Save Changes" to flush rewrite rules
3. Activate your license (if applicable)

2.2 License Activation

2.2.1 Obtaining a License

Purchase a license from the plugin vendor. You will receive:

- License key
- License server URL
- Server secret (for secure validation)

2.2.2 Activating the License

1. Go to **JobRoom** > **Settings** > **License**
2. Enter the License Server URL
3. Enter the Server Secret

4. Enter your License Key
5. Click **Activate License**

2.2.3 License Status

The license panel shows:

- **Valid**: License is active and valid
- **Invalid**: License key is incorrect or revoked
- **Expired**: License has expired, renewal required
- **Domain Mismatch**: License is registered to a different domain

2.2.4 Troubleshooting License Issues

- Ensure your server can reach the license server (check firewall)
- Verify the domain matches your license registration
- Contact support if issues persist

2.3 Settings Configuration

Settings are organized into 10 main tabs with subtabs for detailed configuration.

2.3.1 General Tab

Settings Subtab

- **Delete Data on Uninstall**: When enabled, removes all plugin data when uninstalled
- **Platform Owner**: 9 fields describing the legal entity operating the platform (company name, address, contact person, phone, email, VAT/UID, responsible person, commercial register, regulatory authority, DPO name/email). Used by the `[jr_platform_owner]` and `[jr_imprint]` shortcodes.

Pages Subtab

Configure which WordPress pages are used for select JobRoom functionality:

Page Type	Description	Default Content
Pricing	Subscription plans display	`[jr_pricing_table]` shortcode
Imprint	Legal imprint page	`[jr_imprint]` shortcode

For each page type:

- **Dropdown**: Select an existing WordPress page
- **Create Button**: Generate a default page with the appropriate shortcode
- **Edit Button**: Open the selected page in the editor

> **Note**: Search pages, the dashboard, and the registration page are handled directly by the plugin router and do not require a dedicated WordPress page.

Permalinks Subtab

Configure URL slugs for post types:

Post Type	Default Slug	Example URL
Applicant	`applicants`	`/applicants/john-doe/`
Company	`companies`	`/companies/acme-corp/`
Job Offer	`job-offers`	`/job-offers/senior-developer/`

Note: After changing permalinks, the plugin automatically flushes rewrite rules.

Languages Subtab

- **Primary Language**: The default content language (German is primary for Swiss market)
- **Enabled Languages**: German, French, Italian, English

Web App Subtab

Configure the Progressive Web App (PWA) features:

- **Enable Web App**: Toggle PWA functionality (manifest, meta tags, service worker)
- **App Name**: Full application name shown in install prompts
- **Short Name**: Abbreviated name for home screen (max 12 characters recommended)
- **Description**: Application description
- **Display Mode**: `standalone` (default), `fullscreen`, `minimal-ui`, or `browser`
- **Orientation**: Screen orientation preference (`any`, `portrait`, `landscape`, etc.)
- **Theme Color**: Browser chrome color (hex color picker)
- **Background Color**: Splash screen background color (hex color picker)
- **Start URL**: Landing page when launched from home screen (default: `/`)
- **App Icon**: Upload a source icon (minimum 512x512, recommended 1024x1024) — auto-resized to 14 sizes including maskable icons
- **Push Notifications**: Enable browser push notifications for messages, application status changes, and ticket updates
- **VAPID Keys**: Generate VAPID key pair for Web Push protocol (required for push notifications)

Icon Requirements:

- Minimum 512x512 pixels, recommended 1024x1024
- PNG format preferred
- Default icons generated from `assets/icon/icon.png` if no custom icon uploaded
- 14 sizes auto-generated including favicon, Apple touch icon, Android/Chrome, and maskable variants

Countries Subtab

Configure platform availability:

- **Primary Country**: Switzerland (always enabled)
- **EFTA Countries**: Liechtenstein, Norway, Iceland
- **EU Countries**: 27 EU member states for worker scouting
- **EU Scouting**: Enable cross-border recruitment

2.3.2 Data Tab

Skills Subtab

- **Export Skills**: Download skills as CSV or JSON
- **Import Skills**: Upload skills from file
- **Seed Data**: Load sample IT skills (90 skills)

Occupations Subtab

Manage occupation taxonomy:

- View occupation-industry relationships
- Import/export occupations

Industries Subtab

Manage industry taxonomy:

- View industry hierarchy
- Import/export industries

Profiles Subtab

Configure profile fields:

- CV import settings
- Profile export options
- GDPR compliance settings

Utilities Subtab

- Clear caches
- Reset options
- Debug information

2.3.3 Matching Tab

Configure the matching algorithm weights:

Criterion	Default Weight	Description
Skills	30%	Skill proficiency vs job requirements
Experience	20%	Years of experience alignment
Location	15%	Canton match and remote compatibility
Employment Type	10%	Job type preference match
Workload	10%	Percentage range overlap
Availability	10%	Start date compatibility
Work Permit	5%	Permit type compatibility

Minimum Threshold: Score below which matches are not displayed (default: 50%)

2.3.4 Business Tab

Payment Gateway Subtab

Configure Payrexx integration:

- **Instance Name**: Your Payrexx instance
- **API Key**: Payrexx API key
- **Webhook Secret**: For payment notifications

Options Subtab

- **Enable Subscriptions**: Toggle subscription system
- **Trial Period**: Days of free trial for new companies

Plans Subtab

Configure subscription tiers:

Tier	Monthly (CHF)	Yearly (CHF)	Job Limit	Contact Persons
Free	0	0	1	1
Basic	49	490	3	1
Professional	149	1,490	10	5
Enterprise	299	2,990	Unlimited	Unlimited

Both job limits and contact person limits are configurable per tier.

2.3.5 Marketing Tab

Newsletter Subtab

Configure Listmonk integration:

- **API URL**: Listmonk instance URL
- **Username/Password**: API credentials (stored encrypted)
- **List Mapping**: Assign lists to user roles

Newsletter Widget color variant: The sidebar widget exposes a **Color variant** dropdown (Default, Primary, Secondary, Success, Danger, Warning, Info, Light, Dark). Selecting a variant applies the matching Bootstrap 5 `text-bg-*` class to the widget card and adjusts the subscribe button color for contrast automatically. The same `variant` attribute is available on the `[jr_newsletter_form]` and `[jr_newsletter_widget]` shortcodes.

Social Subtab

- **Enable Social Sharing**: Show share buttons on profiles
- **Enable Social Links**: Allow profile social media links
- **Platforms**: LinkedIn, Xing, GitHub, Facebook, Instagram, Fediverse, Signal

SEO Subtab

- **Open Graph Tags**: Enable for social sharing previews
- **Twitter Cards**: Enable Twitter/X card meta tags

- **JSON-LD**: Enable structured data for search engines
- **Sitemap**: Include profiles in WordPress sitemap

2.3.6 Integrations Tab

REST API Subtab

- **Enable API**: Toggle REST API access
- **Rate Limiting**: Requests per minute/hour
- **Default Scopes**: Permissions for new API keys

API Keys Subtab

Manage admin-level API keys:

- Create keys with custom scopes (read, write, delete, admin)
- View active keys
- Revoke keys

Documentation Subtab

Links to the interactive API documentation (Swagger UI) at `/api-docs/`. Also shows quick reference information:

- **Documentation URL**: `/api-docs/`
- **API Base URL**: `/wp-json/jobroom/v1/`
- **OpenAPI Spec**: `/wp-json/jobroom/v1/openapi.json`
- **Authentication methods**: Bearer token and X-API-Key header

Solr Subtab

- **Enable Solr**: Toggle Apache Solr search integration
- **Connection Settings**: Host, port, core name, scheme, timeout
- **Test Connection**: Verify Solr is reachable
- **Create Core**: Create the Solr core (or use `docker exec <container> solr create -c jobroom`)
- **Apply Schema**: Apply field definitions and copy fields to Solr
- **Reindex All**: Batch reindex all posts with progress tracking
- **Clear Index**: Remove all documents from the Solr index
- **Statistics**: Document counts per entity type (Solr vs WordPress)

Help Pages Subtab

Import user-facing help pages into WordPress:

- **Per-Locale Import**: Download WXR XML or import directly for `en_US`, `de_DE`, `de_CH`
- **8 Pages per Locale**: Parent/child hierarchy (root index, Getting Started, Applicant Guide, Company Guide, Applications & Messages, Account & Security, Subscriptions & Billing, Support)
- **Force Update**: Checkbox to overwrite previously imported pages (identified by `_jr_help_page` meta)
- **Conflict Detection**: Pages with matching slugs not owned by the importer are flagged as conflicts and never modified
- **Status Icons**: Per-item results showing created, updated, skipped, conflict, or error status

Media Offloader Subtab

This subtab only appears when the Advanced Media Offloader plugin is installed and active.

- **Private File Offloading**: Enable/disable cloud storage for private files (certificates, helpdesk attachments)
- **Presigned URL Expiry**: How long download links remain valid (1–60 minutes, default: 5)
- **Current Provider**: Shows the configured cloud provider and auto-offload status from the offloader plugin

When enabled, private files are uploaded to cloud storage (DigitalOcean Spaces, AWS S3, MinIO, etc.) with a `private` ACL instead of `public-read`. Downloads go through the existing permission check (ownership, admin, supporter role), then redirect to a time-limited presigned URL. If the file is not yet offloaded or the integration is disabled, files are served locally as before.

Prometheus Subtab

- **Enable Metrics**: Expose 51 JobRoom-specific metrics for Prometheus
- **Metrics Endpoint**: `/wp-json/prometheus/metrics`
- **Grafana Dashboard**: Pre-configured dashboard with 59 panels across 11 sections
- **Metric Groups**: Profiles, applications, matching, subscriptions, communication, search, taxonomy, Solr, users/verification, security/compliance, invoices/revenue, job lifecycle, newsletter, GDPR consent, API keys, data quality

2.3.7 Compliance Tab

Cookie Consent Subtab

- **Enable Cookie Banner**: Show GDPR cookie consent
- **Banner Position**: Bottom, top, corner positions
- **Banner Style**: Default, dark, minimal
- **Consent Version**: Increment to re-prompt users

Two-Factor Auth Subtab

- **Enable MFA**: Allow users to enable 2FA
- **Issuer Name**: Shown in authenticator apps
- **Required Roles**: Force MFA for specific roles

Password Policy Subtab

- **Enable Policy**: Enforce password rules
- **Minimum Length**: Characters required
- **Require Uppercase**: At least one uppercase letter
- **Require Lowercase**: At least one lowercase letter
- **Require Numbers**: At least one digit
- **Require Symbols**: At least one special character
- **Pwned Password Check**: Optionally reject passwords that have appeared in known data breaches by querying the Have I Been Pwned Pwned Passwords API — no API key required. Uses k-anonymity: only the first 5 characters of the SHA1 hash are sent to the API; the plaintext password is never transmitted. Results are cached for 7 days. If the API is unreachable, the check is skipped and registration proceeds normally (fail-open behaviour)

2.3.8 Invitations Tab

- **Enable Invite-Only Mode**: When enabled, registration requires a valid invite code. Public pages (landing, pricing, search) remain accessible

- **Allow User Invites**: Let registered users send invitations to others
- **User Invite Limit**: Maximum number of invites per registered user (default: 3)
- **Default Expiry (Admin)**: Days until admin-generated invites expire (0 = never)
- **Default Expiry (User)**: Days until user-generated invites expire (default: 30)

Invitation Management (Settings > Invitations):

- **Create Invitations**: Generate single or bulk invite codes with optional email targeting, role locking, multi-use, and custom expiry
- **Invitations List Table**: Filterable by status (active/revoked/expired), searchable by code/email/note, bulk revoke/delete
- **Invitation Statistics**: Total codes, active, redeemed, expired counts

2.3.9 Helpdesk Tab

General Subtab

- **Enable Helpdesk**: Toggle the support ticket system on/off
- **Support Email**: Reply-To address for ticket email notifications
- **Auto-Close Days**: Automatically close resolved tickets after this many days without activity (1–90)
- **Max Attachments**: Maximum file attachments per ticket or reply (0–20)
- **Max Attachment Size**: File size limit in MB (1–50)

SLA Subtab

- **Enable SLA**: Toggle SLA tracking and enforcement
- **Response Times**: Configurable per subscription tier (Free/Basic/Professional/Enterprise) and priority level (Low/Medium/High/Critical)
- **Warning Threshold**: Percentage of SLA time remaining before a warning is shown

Email Subtab

- **Enable Email Replies**: Allow users to reply to tickets via email
- **Create Tickets from Email**: Create new tickets from incoming emails that don't match existing threads
- **Auto-Reply**: Send automatic confirmation for new tickets
- **Signature Strip Patterns**: Custom regex or plain text patterns to remove email signatures
- **IMAP Polling**: Configure IMAP mailbox for incoming email processing (host, port, encryption, credentials)
- **Webhook Secret**: Shared secret for email webhook verification

Categories Subtab

- Manage ticket categories (`jr_ticket_category` taxonomy)
- Create, edit, and delete support categories

2.3.10 Performance Tab

Caching Subtab

- **Enable Search Cache**: Cache search results
- **Cache TTL**: Time-to-live in minutes
- **Count Cache TTL**: Statistics cache duration
- **Clear Cache**: Manual cache clearing

Assets Subtab

- **Enable Minification**: Minify CSS/JS files
- **Enable Concatenation**: Combine CSS/JS files into single bundles per context (frontend and admin) to reduce HTTP requests; requires minification to be enabled first
- **Lazy Loading**: Enable image lazy loading
- **Defer Scripts**: Defer non-critical JavaScript
- **Build Assets**: Generate minified files and bundles
- **Clear Minified Cache**: Remove cached assets and bundles
- **Build Status**: Shows minification/concatenation state, file counts, bundle counts, and total savings

Search Subtab

- **Pagination Mode**: Choose how users navigate through multiple pages of search results.
  - **Pagination** (default): Numbered page links appear below the results grid. Clicking a page number reloads the results for that page. Compatible with all browsers and SEO-friendly.
  - **Infinite Scroll**: The numbered page links are hidden. Instead, additional results are loaded and appended automatically when the user scrolls to the bottom of the list, using the browser's native `IntersectionObserver` API. No button click is required.

Optimization Subtab

- **Preload Hints**: Add resource hints

2.3.11 License Tab

See License Activation section above.

2.4 Content Management

2.4.1 Managing Applicants

Access via **JobRoom > Applicants**

List View

- Filter by status (published, draft, pending)
- Search by name or skills
- Sort by date, match score, or name
- Bulk actions: publish, unpublish, delete

Editing Applicants

Applicant profiles have multiple meta boxes:

- **Personal Information**: Name, email, phone, location
- **Work Experience**: Employment history (repeater field); each entry includes an **Achievements** sub-repeater for bullet-point accomplishments
- **Education**: Educational qualifications (repeater field)
- **Skills**: Skill selection with proficiency levels
- **Languages**: Language proficiency
- **Availability**: Work preferences

- **Certificates**: Upload certificates/documents via WordPress media uploader, with title, issuer, dates, and related CV entry linking
- **Personal References**: Repeater field for professional references — name, company, relationship, email, phone; visibility defaults to "Matched"
- **Matching Jobs**: View jobs that match this applicant

2.4.2 Managing Companies

Access via **JobRoom > Companies**

Company Meta Boxes

- **Company Information**: Name, website, industry
- **Address**: Full Swiss address format
- **Contact Persons**: Repeater table for multiple contact persons (name, position, email, phone). Count gated by subscription tier
- **Culture & Benefits**: Company benefits selection
- **Subscription Override**: Admin can override subscription tier

2.4.3 Managing Job Offers

Access via **JobRoom > Job Offers**

Job Offer Meta Boxes

- **Job Details**: Title, description, employment type
- **Company**: Link to company profile
- **Compensation**: Salary (yearly/monthly/hourly), 13th month option
- **Location**: Canton, city, remote options
- **Requirements**: Skills, experience level, work permits
- **Matching Applicants**: View applicants that match this job

2.4.4 Taxonomy Management

Skills Taxonomy

Access via **JobRoom > Skills**:

- Hierarchical skill categories
- Relationship to occupations
- Import/export functionality

Occupations Taxonomy

Access via **JobRoom > Occupations**:

- Job titles and roles
- Linked to industries
- Used for filtering and matching

Industries Taxonomy

Access via **JobRoom > Industries**:

- Business sectors
- Company classification

- Search filtering

2.5 User Management

2.5.1 User Roles

WP JobRoom creates three custom roles:

Role	Slug	Capabilities
Applicant	`jr_applicant`	Manage own profile, apply to jobs, messaging
Company	`jr_company`	Manage company, post jobs, review applications
Supporter	`jr_supporter`	Helpdesk-only admin access, read-only profile access

2.5.2 Role Capabilities

Applicant Role:

- `read` - View content
- `edit_jr_applicant` - Edit own profile
- `delete_jr_applicant` - Delete own profile

Company Role:

- `read` - View content
- `edit_jr_company` - Edit company profile
- `edit_jr_job_offers` - Create/edit job offers
- `delete_jr_job_offers` - Delete job offers

Supporter Role:

- `read` - View content (read-only access to applicant/company profiles)
- `jr_manage_tickets` - Create, reply to, and manage support tickets
- `jr_view_ticket_reports` - View helpdesk reports in admin

Supporters can log in to WordPress admin and see only the JobRoom > Tickets and JobRoom > Reports (Helpdesk tab) pages. They are blocked from all other admin areas.

2.5.3 Linking Users to Profiles

Each user is linked to one profile via `_jr_linked_post_id` user meta.

To manually link a user to a profile:

1. Edit the user in WordPress Admin
2. Find the “JobRoom Profile” section
3. Select the appropriate profile post

2.6 Subscription Management

2.6.1 Subscriptions Overview

Access via **JobRoom > Subscriptions**

Subscription List

View all subscriptions with:

- Company name
- Plan tier
- Status (active, cancelled, expired)
- Billing cycle
- Next renewal date
- Revenue

Subscription Statuses

Status	Description
Pending	Awaiting payment
Active	Valid subscription
Cancelled	User cancelled, valid until period end
Expired	Period ended, renewal failed
Failed	Payment failed

2.6.2 Invoice Management

Access via **JobRoom > Invoices**

Invoice List

- Filter by company or status
- View invoice details
- Download PDF invoices

Invoice Numbering

Format: `JR-YYYY-NNNNN` (e.g., JR-2026-00001)

2.6.3 Subscription Override

Administrators can override company subscriptions:

1. Edit the company profile
2. Find “Subscription Override” meta box
3. Select override tier and expiration
4. Save the company

2.7 Frontend Admin Dashboard

Administrators visiting `/dashboard/` on the frontend are shown a platform-wide overview dashboard (instead of the applicant/company dashboard). No additional configuration is required — any WordPress user with the `manage_options` capability automatically sees the admin dashboard.

2.7.1 Sections

Platform Overview — Six stat cards showing: Total Applicants, Total Companies, Active Job Offers, Total Applications, Active Subscriptions, and Monthly Recurring Revenue (MRR in CHF).

Helpdesk Overview — Status pills row showing ticket counts per status, SLA overdue badge (shown in red when breached tickets exist), and a table of the 10 most recent tickets with subject, priority, status, creation date, and SLA indicator. A link to the full WP Admin ticket list is provided below.

Platform Health — Three cards: Average Match Score with a progress bar and jobs/match counts; Search Engine showing Solr status (Available/Unavailable) or WordPress Search fallback; MRR breakdown by subscription tier.

Quick Actions — Four action cards linking to: Admin Reports, Manage Users (WP Admin Users list), Plugin Settings, and All Tickets (WP Admin helpdesk list).

2.8 Helpdesk Administration

The helpdesk system provides a support ticket workflow for applicants and companies.

2.8.1 Helpdesk Settings

Navigate to **JobRoom** > **Settings** > **Helpdesk** to configure:

General Subtab

- **Enable Helpdesk**: Toggle the helpdesk system on or off
- **Support Email**: The "From" address for helpdesk emails
- **Support Email Domain**: Domain used for reply-by-email Message-ID headers
- **Auto-Close Days**: Automatically close resolved tickets after N days of inactivity (0 to disable)
- **Satisfaction Surveys**: Enable post-resolution satisfaction rating prompts

SLA Subtab

Configure per-tier, per-priority response time targets:

Tier	Low	Normal	High	Urgent
Free	48h	24h	—	—
Basic	36h	18h	—	—
Professional	24h	12h	6h	—
Enterprise	12h	6h	2h	1h

- Values are editable per cell; leave blank to disable a priority for that tier
- SLA deadlines are calculated from ticket creation time
- Overdue tickets are highlighted in the admin list and trigger warning emails
- The pricing page automatically reflects the configured response times

Categories Subtab

Manage ticket categories (e.g., Billing, Technical, Account). Categories help staff route and filter tickets.

- Add, edit, or delete categories
- Categories are displayed to users when creating tickets

IMAP Subtab

Configure inbound email polling for reply-by-email support:

- **IMAP Host/Port/Encryption**: Mail server connection settings
- **Username/Password**: IMAP credentials (password stored encrypted)
- **Folder**: Mailbox folder to poll (default: INBOX)
- **Error Folder**: Folder for unparseable emails

2.8.2 Managing Tickets

Navigate to **JobRoom** > **Tickets** to access the admin ticket list.

Ticket List Page

- Filter by status, priority, category, or assignee
- Sort by date, priority, or last reply
- Bulk actions for status changes
- Statistics cards showing open, unassigned, and SLA-breached counts

Ticket Detail Page

Click a ticket to view its full detail:

- View the original message and all replies
- Add public replies or internal notes (visible only to staff)
- Change status, priority, or category
- Assign the ticket to a staff member
- View SLA deadline and breach status

2.8.3 Status Workflow

Tickets follow a defined status workflow:

```
new → open → in_progress → resolved → closed
         ↘ on_hold ↗        ↗
         ↘ waiting_customer ↗
```

Terminal states (no further transitions): `closed`, `spam`

When a customer replies to a ticket in "Waiting for Customer" or "Resolved" status, it automatically transitions to "In Progress".

2.8.4 SLA Management

SLA response times are configured per subscription tier and priority:

Tier	Low	Normal	High	Urgent
Free	48h	24h	-	-
Basic	36h	18h	-	-
Professional	24h	12h	6h	-
Enterprise	12h	6h	2h	1h

SLA deadlines are calculated from ticket creation time. Breached tickets are flagged in the admin list and trigger warning emails.

2.8.5 Email Notifications

The helpdesk sends the following notification emails:

Event	Recipients
Ticket created	Admins, assigned staff
New reply	Reporter or staff (depending on who replied)
Status changed	Reporter
Ticket assigned	Assigned staff member
SLA warning	Assigned staff, admins
Satisfaction survey	Reporter (after resolution)

2.8.6 IMAP Polling

When configured, the system polls an IMAP mailbox every 5 minutes to process incoming email replies:

- Matches emails to tickets via `X-JR-Thread-ID` header, `References` / `In-Reply-To` headers, or plus-addressing
- Strips email signatures, quoted text, and client-specific formatting
- Creates new tickets from unmatched emails
- Moves unparseable emails to the error folder

2.8.7 WP-CLI Commands

Manage helpdesk operations from the command line:

```
# View helpdesk status overview
wp jobroom helpdesk status
```

2.8.8 Prometheus Metrics

When Prometheus integration is enabled, the following helpdesk metrics are collected:

- `jobroom_helpdesk_tickets_total` - Total tickets
- `jobroom_helpdesk_tickets_by_status_total` - Tickets by status
- `jobroom_helpdesk_tickets_by_priority_total` - Tickets by priority
- `jobroom_helpdesk_sla_breached_total` - SLA-breached tickets
- `jobroom_helpdesk_avg_satisfaction` - Average satisfaction rating
- `jobroom_helpdesk_unassigned_total` - Unassigned open tickets

2.9 Reports and Analytics

Access via **JobRoom > Reports**

2.9.1 Overview Report

Platform-wide statistics:

- Total applicants, companies, job offers
- Active users
- Registration trends

2.9.2 Users Report

- New registrations over time
- Role distribution
- Geographic distribution

2.9.3 Subscriptions Report

- Active subscriptions by tier
- Monthly Recurring Revenue (MRR)
- Churn rate
- Revenue trends

2.9.4 Matching Report

- Average match scores
- Jobs with high-quality matches
- Matching algorithm effectiveness

2.9.5 Activity Report

- Profile views
- Search activity
- Application rates
- Most viewed profiles

2.9.6 Helpdesk Report

Ticket and support metrics:

- Ticket volume over time (new, open, resolved)

- SLA compliance rate
- Average response and resolution times
- Ticket distribution by category and priority
- Per-agent performance (tickets assigned, resolved, avg. response time)

> **Note**: The Helpdesk tab is only visible to users with the `jr_view_ticket_reports` capability (administrators and supporters).

2.9.7 Exporting Reports

Each report supports export to:

- CSV format
- JSON format

2.10 Maintenance

2.10.1 Cache Management

Clear various caches:

1. **Search Cache**: Cached search results
2. **Count Cache**: Dashboard statistics
3. **Match Cache**: Cached match scores
4. **Asset Cache**: Minified CSS/JS files

Access via **JobRoom > Settings > Performance**

2.10.2 Database Optimization

The plugin creates one custom table:

- `{prefix}jr_analytics_events` - Event tracking data

Periodic optimization recommended for large installations.

2.10.3 Backup Recommendations

Include in backups:

- `wp_posts` (all `jr_*` post types)
- `wp_postmeta` (profile data)
- `wp_usermeta` (user preferences, API keys)
- `wp_terms`, `wp_term_taxonomy` (skills, occupations, industries)
- `{prefix}jr_analytics_events` (analytics data)
- `wp_options` (all `wp_jobroom_*` options)

2.10.4 Updates

1. Backup your site before updating

2. Download the new version
3. Deactivate the current plugin
4. Delete the old plugin files
5. Upload and activate the new version
6. Check the changelog for required actions

2.11 Troubleshooting

2.11.1 Common Issues

Permalinks Not Working

Symptom: Profile pages show 404 errors

Solution: 1. Go to **Settings** > **Permalinks** 2. Click "Save Changes" (even without changes) 3. Clear any caching plugins

Match Scores Not Calculating

Symptom: Match scores show 0% or don't appear

Solution: 1. Go to **JobRoom** > **Settings** > **Matching** 2. Verify weights are configured 3. Clear match cache via **Settings** > **Performance** 4. Ensure profiles have skills assigned

Email Notifications Not Sending

Symptom: Users don't receive notification emails

Solution: 1. Verify WordPress email is working (`wp_mail()`) 2. Check spam folders 3. Configure SMTP plugin if needed 4. Verify notification settings enabled

REST API Returns 401

Symptom: API requests fail with authentication error

Solution: 1. Verify API is enabled in settings 2. Check API key is valid and not revoked 3. Verify correct header: `Authorization: Bearer {key}` 4. Check rate limiting hasn't been exceeded

Prometheus Metrics Empty

Symptom: No metrics appearing at `/wp-json/prometheus/metrics`

Solution: 1. Enable Prometheus in **Settings** > **Integrations** > **Prometheus** 2. Verify wp-prometheus plugin is installed 3. Check for PHP errors in logs

2.11.2 Debug Mode

Enable WordPress debug mode for troubleshooting:

```
// wp-config.php
define( 'WP_DEBUG', true );
define( 'WP_DEBUG_LOG', true );
```

Check `wp-content/debug.log` for errors.

2.11.3 Support

For support:

1. Check this documentation
2. Review the changelog for known issues
3. Contact the plugin vendor with:
   - WordPress version
   - PHP version
   - Plugin version
   - Error messages
   - Steps to reproduce

2.12 Security Considerations

2.12.1 User Data

- All user data is stored in WordPress database
- Sensitive data (API keys, TOTP secrets) is encrypted
- Passwords use WordPress native hashing

2.12.2 API Security

- API keys are hashed (only shown once on creation)
- Rate limiting prevents abuse
- Scope-based permissions

2.12.3 GDPR Compliance

- Cookie consent banner available
- Data export functionality
- Data erasure on user request
- Privacy policy integration

2.12.4 Best Practices

1. Keep WordPress and the plugin updated
2. Use strong admin passwords
3. Enable 2FA for all admin users
4. Regular backups
5. Monitor access logs
6. Use HTTPS everywhere

---

3 Developer Guide

3.1 Architecture Overview

3.1.1 Design Principles

WP JobRoom follows these design principles:

- **Singleton Pattern**: Core components use singleton instances for global access
- **PSR-4 Autoloading**: All classes autoloaded via Composer
- **WordPress Standards**: Uses WordPress APIs and coding standards
- **Security First**: All inputs sanitized, outputs escaped, capabilities checked

3.1.2 Technology Stack

Component	Technology
Backend	PHP 8.3+, WordPress Plugin API
Templates	Twig 3.0
Frontend	Vanilla JavaScript, CSS Custom Properties
Database	WordPress custom tables + post meta
API	WordPress REST API

3.1.3 Plugin Bootstrap

```
// wp-jobroom.php
define( 'WP_JOBROOM_VERSION', '0.26.3' );
define( 'WP_JOBROOM_PLUGIN_DIR', plugin_dir_path( __FILE__ ) );

// Composer autoload
require_once WP_JOBROOM_PLUGIN_DIR . 'vendor/autoload.php';

// Initialize plugin
add_action( 'plugins_loaded', function() {
    \Magdev\WpJobroom\Plugin::get_instance();
} );
```

3.2 Directory Structure

wp-jobroom/

```
assets/
|-- css/                                  Stylesheets
|-- js/                                   JavaScript files
`-- icon/                                 Plugin icon

data/                                     Skill seeds, Grafana dashboards

docs/                                     7 guides, OpenAPI spec

languages/                                .pot template, 13 locale .po files

lib/
`-- wc-licensed-product-client/           License client (git submodule)

src/
|-- Admin/                                Admin UI and settings
|-- Analytics/                            Event tracking and statistics
|-- Communication/                        Messages and applications
|-- Core/                                 Base classes, traits, utilities
|-- DataExport/                           PDF and GDPR export
|-- DataImport/                           CV import
|-- Debug/                                Debug Bar integration (dev only)
|-- Frontend/                             Frontend controllers and views
|-- GDPR/                                 Cookie consent, privacy
|-- Helpdesk/                             Support ticket system
|-- Invitation/                           Invitation system
|-- License/                              License management
|-- Marketing/                            Newsletter, social, SEO
|-- Matching/                             Matching algorithm
|-- PostType/                             Custom post types
|-- Profile/                              Profile meta boxes
|-- Prometheus/                           Metrics integration
|-- REST/                                 REST API endpoints
|-- Role/                                 User roles
|-- Security/                             MFA, encryption, passwords, media offloader
|-- Solr/                                 Apache Solr search integration
|-- Subscription/                         Subscription and billing
|-- Swiss/                                Swiss-specific features
|-- Taxonomy/                             Custom taxonomies
`-- WebApp/                               PWA, push notifications

templates/              95+ Twig templates (account, auth, blocks, dashboard,
                        emails, export, gdpr, layouts, marketing, messages,
                        profiles, search, shortcodes, subscription, widgets)

vendor/                                   Composer dependencies

|-- wp-jobroom.php                        Main plugin file
|-- composer.json                         Dependencies
`-- CHANGELOG.md, README.md, LICENSE, PLAN.md
```

24 namespaces · 95+ templates · 13 locales · 1,200+ tests

Figure 1: Directory Structure

3.3 Core Classes

3.3.1 Plugin Class

`Magdev\WpJobroom\Plugin` - Main plugin singleton

```
// Get plugin instance
$plugin = \Magdev\WpJobroom\Plugin::get_instance();

// Access services
$plugin->get_template();      // Twig template engine
$plugin->get_frontend();      // Frontend controller
```

3.3.2 Key Manager Classes

Class	Purpose
`License\Manager`	License validation and activation
`Matching\Calculator`	Match score calculation
`Matching\Cache`	Match result caching
`Frontend\Template`	Twig template rendering
`Frontend\Router`	URL routing
`REST\ApiAuthentication`	API key management
`Analytics\EventTracker`	Event recording
`Analytics\Statistics`	Statistics calculation
`Analytics\Dashboard\AdminDashboardController`	Admin and support worker frontend dashboards
`Subscription\Payment\PayrexxClient`	Payment gateway
`Core\QueryOptimizer`	Database query optimization
`Core\SearchCache`	Search result caching
`Debug\DebugCollector`	Debug data collector (dev only)
`Debug\DebugBarPanel`	Debug Bar panel rendering
`Admin\HelpPageImporter`	Help page generation and WordPress import
`Core\LocationFormatter`	Centralized location display formatting
`Core\SingletonTrait`	Reusable singleton pattern trait

3.3.3 Base Classes and Traits

AjaxHandler Abstract Class

```
abstract class AjaxHandler {
    abstract protected function get_action_name(): string;
    abstract protected function get_nonce_action(): string;
    abstract protected function handle_request(): void;

    protected function verify_nonce(): bool;
    protected function check_capability( string $capability ): bool;
    protected function send_success( $data = null ): void;
    protected function send_error( string $message, int $code = 400 ): void;
}
```

NonceHandler Trait

```
trait NonceHandler {
    protected function verify_nonce( string $action, string $nonce_field =
'_wpnonce' ): bool;
    protected function create_nonce( string $action ): string;
    protected function get_nonce_field( string $action ): string;
}
```

SingletonTrait

```
trait SingletonTrait {
    public static function get_instance(): static;
    private function __clone(): void;
    public function __wakeup(): void; // throws \RuntimeException
}
```

Used by 40+ core singleton classes across all subsystems (Frontend, Communication, Marketing, Subscription, Helpdesk, Solr, Analytics, Security, REST).

MetaDataHelper Trait

```
trait MetaDataHelper {
    protected function get_post_meta_value( int $post_id, string $key, $default =
'' );
    protected function update_post_meta_value( int $post_id, string $key, $value ):
void;
    protected function get_user_meta_value( int $user_id, string $key, $default =
'' );
    protected function get_post_terms( int $post_id, string $taxonomy ): array;
}
```

3.4 Hooks and Filters

3.4.1 Action Hooks

Profile Events

```
// Fired when a profile is viewed
do_action( 'wp_jobroom_profile_viewed', int $post_id, string $post_type );

// Fired when a profile is updated
do_action( 'wp_jobroom_profile_updated', int $post_id, string $post_type );
```

Application Events

```
// Fired when an application is submitted
do_action( 'wp_jobroom_application_submitted', int $application_id, int
$applicant_id, int $job_id );

// Fired when application status changes
do_action( 'wp_jobroom_application_status_changed', int $application_id, string
$old_status, string $new_status );
```

Message Events

```
// Fired when a message is sent
do_action( 'wp_jobroom_message_sent', int $message_id, int $sender_id, int
$recipient_id );
```

Subscription Events

```
// Fired when subscription status changes
do_action( 'wp_jobroom_subscription_status_changed', int $subscription_id, string
$old_status, string $new_status );

// Fired on successful payment
do_action( 'wp_jobroom_payment_received', int $subscription_id, float $amount );
```

Search Events

```
// Fired when a search is performed
do_action( 'wp_jobroom_search_performed', string $search_type, array $filters, int
$result_count );
```

Helpdesk Events

```
// Fired when a support ticket is created
do_action( 'wp_jobroom_ticket_created', int $ticket_id, int $user_id, string $tier,
array $data );

// Fired when a reply is added to a ticket
do_action( 'wp_jobroom_ticket_reply_added', int $reply_id, int $ticket_id, int
$user_id, bool $is_internal );

// Fired when a ticket status changes
do_action( 'wp_jobroom_ticket_status_changed', int $ticket_id, string $new_status,
string $old_status, int $user_id );

// Fired when a ticket is assigned to staff
do_action( 'wp_jobroom_ticket_assigned', int $ticket_id, int $assignee_id, int
$old_assignee );
```

Push Notification Integration

Push notifications are dispatched alongside email for these events:

- **New message**: `on_message_sent()` in `NotificationManager`
- **Application status change**: `on_application_status_changed()` in `NotificationManager`
- **Ticket reply**: `on_reply_added()` in `HelpdeskNotificationManager`
- **Ticket status change**: `on_status_changed()` in `HelpdeskNotificationManager`

To check push availability:

```
use Magdev\WpJobroom\WebApp\Push\PushNotificationSender;

if ( PushNotificationSender::is_available() ) {
    PushNotificationSender::get_instance()->send_message_notification(
        $recipient_id, $sender_name, $subject, $thread_id
    );
}
```

Push notification methods:

- `send_message_notification( int $recipient_id, string $sender_name, string $subject, int $thread_id )`
- `send_application_status_notification( int $applicant_user_id, string $job_title, string $new_status_label, string $job_url )`
- `send_ticket_reply_notification( int $user_id, int $ticket_id, string $subject, string $reply_by )`
- `send_ticket_status_notification( int $user_id, int $ticket_id, string $subject, string $new_status_label )`

3.4.2 Filter Hooks

Matching Algorithm

```
// Modify matching weights
$weights = apply_filters( 'wp_jobroom_matching_weights', array $weights );

// Modify match score calculation
```

```
$score = apply_filters( 'wp_jobroom_match_score', float $score, int $applicant_id,
int $job_id );

// Modify match criteria
$criteria = apply_filters( 'wp_jobroom_match_criteria', array $criteria );
```

Profile Display

```
// Modify profile data before display
$data = apply_filters( 'wp_jobroom_applicant_profile_data', array $data, int
$post_id );
$data = apply_filters( 'wp_jobroom_company_profile_data', array $data, int
$post_id );
$data = apply_filters( 'wp_jobroom_job_offer_data', array $data, int $post_id );

// Modify profile sections visibility
$visible = apply_filters( 'wp_jobroom_profile_section_visible', bool $visible,
string $section, int $post_id );
```

Search Results

```
// Modify search query arguments
$args = apply_filters( 'wp_jobroom_search_args', array $args, string $search_type );

// Modify search results
$results = apply_filters( 'wp_jobroom_search_results', array $results, string
$search_type );
```

Email Templates

```
// Modify email content before sending
$content = apply_filters( 'wp_jobroom_email_content', string $content, string
$template, array $data );

// Modify email headers
$headers = apply_filters( 'wp_jobroom_email_headers', array $headers, string
$template );
```

Settings

```
// Add custom settings tabs
$tabs = apply_filters( 'wp_jobroom_settings_tabs', array $tabs );

// Modify skill proficiency levels
$levels = apply_filters( 'wp_jobroom_skill_proficiency_levels', array $levels );

// Modify subscription tiers
$tiers = apply_filters( 'wp_jobroom_subscription_tiers', array $tiers );
```

REST API

```
// Modify API response data
$data = apply_filters( 'wp_jobroom_api_response', array $data, string $endpoint );

// Modify rate limiting
$limits = apply_filters( 'wp_jobroom_api_rate_limits', array $limits, int
$user_id );
```

3.5 REST API

3.5.1 Authentication

API requests require authentication via API key:

```
GET /wp-json/jobroom/v1/applicants
Authorization: Bearer jr_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

Or via header:

```
GET /wp-json/jobroom/v1/applicants
X-API-Key: jr_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

3.5.2 Available Scopes

Scope	Description
`read`	Read access to resources
`write`	Create and update resources
`delete`	Delete resources
`admin`	Administrative operations

3.5.3 Rate Limiting

Default limits:

- 60 requests per minute
- 1000 requests per hour

Rate limit headers included in responses:

```
X-RateLimit-Limit: 60
X-RateLimit-Remaining: 58
X-RateLimit-Reset: 1707148800
```

3.5.4 Interactive Documentation

The full API reference is available as an interactive Swagger UI at `/api-docs/`. The OpenAPI 3.1 specification is served dynamically at:

```
GET /wp-json/jobroom/v1/openapi.json
```

The spec endpoint automatically sets the server URL to match the site, so it works in any environment.

3.5.5 Endpoints

See OpenAPI Specification for the raw spec file, or visit `/api-docs/` for the interactive explorer.

Key Endpoints

Method	Endpoint	Description
GET	`/jobroom/v1/`	API information
GET	`/jobroom/v1/applicants`	List applicants
POST	`/jobroom/v1/applicants`	Create applicant
GET	`/jobroom/v1/applicants/{id}`	Get applicant
PUT	`/jobroom/v1/applicants/{id}`	Update applicant
DELETE	`/jobroom/v1/applicants/{id}`	Delete applicant
GET	`/jobroom/v1/companies`	List companies
GET	`/jobroom/v1/job-offers`	List job offers
GET	`/jobroom/v1/search/jobs`	Search jobs
GET	`/jobroom/v1/search/applicants`	Search applicants
GET	`/jobroom/v1/matching/jobs/{id}`	Get matching jobs
GET	`/jobroom/v1/matching/applicants/{id}`	Get matching applicants
GET	`/jobroom/v1/messages`	List messages
POST	`/jobroom/v1/messages`	Send message
GET	`/jobroom/v1/applications`	List applications
POST	`/jobroom/v1/applications`	Submit application
GET	`/jobroom/v1/manifest.json`	Web App Manifest (public)
POST	`/jobroom/v1/me/push/subscribe`	Subscribe to push (cookie auth)
DELETE	`/jobroom/v1/me/push/subscribe`	Unsubscribe from push
GET	`/jobroom/v1/me/push/subscriptions`	List push subscriptions
GET	`/jobroom/v1/me/push/vapid-key`	Get VAPID public key

3.5.6 Response Format

All responses follow this format:

```
{
  "success": true,
  "data": { ... },
```

```
  "meta": {
    "total": 100,
    "page": 1,
    "per_page": 10,
    "pages": 10
  }
}
```

Error responses:

```
{
  "success": false,
  "code": "invalid_api_key",
  "message": "Invalid or expired API key"
}
```

3.6 Template System

3.6.1 Twig Integration

WP JobRoom uses Twig 3.0 for templating.

Template Engine Access

```
$template = \Magdev\WpJobroom\Plugin::get_instance()->get_template();
$html = $template->render( 'profiles/applicant/single.html.twig', [
    'applicant' => $applicant_data,
    'can_edit' => current_user_can( 'edit_post', $post_id ),
] );
```

Available Twig Functions

Function	Description
`__( 'text', 'wp-jobroom' )`	Translation
`esc_html( value )`	HTML escaping
`esc_attr( value )`	Attribute escaping
`esc_url( url )`	URL escaping
`wp_nonce_field( action )`	Nonce field
`wp_create_nonce( action )`	Create nonce value
`get_avatar_url( user_id )`	User avatar
`get_the_post_thumbnail_url( post_id )`	Featured image
`jr_social_links( post_id )`	Social media links
`jr_share_links( post_id )`	Share buttons
`do_shortcode( tag, atts )`	Execute a WordPress shortcode and return HTML

Template Inheritance

```
{# templates/profiles/applicant/single.html.twig #}
{% extends 'layouts/single.html.twig' %}

{% block content %}
    <div class="jr-applicant-profile">
        {# Profile content #}
    </div>
{% endblock %}
```

3.6.2 Overriding Templates

Create templates in your theme:

```
your-theme/
└── wp-jobroom/
    └── profiles/
        └── applicant/
            └── single.html.twig
```

The plugin checks for theme templates first.

3.6.3 Template Locations (Priority Order)

1. `{theme}/wp-jobroom/{template}`
2. `{child-theme}/wp-jobroom/{template}`
3. `{plugin}/templates/{template}`

3.7 Multi-Project Theme Chain

WP JobRoom is not a single repository. It is a three-project ecosystem where each project is a separate git repository with its own release cycle. Understanding how the three projects interact is essential for any developer working on templates, styling, or frontend rendering.

3.7.1 The Three-Project Ecosystem

Project	Type	Location
wp-jobroom	Plugin	`wp-content/plugins/wp-jobroom/`
wp-bootstrap	Parent Theme	`wp-content/themes/wp-bootstrap/`
wp-jobroom-theme	Child Theme	`wp-content/themes/wp-jobroom-theme/`

Each project has its own `CLAUDE.md`, its own git branches (`dev` / `main`), and its own version tags. They communicate through WordPress hooks, Twig template resolution, and shared CSS/JS conventions.

3.7.2 Dependency Chain

```
wp-bootstrap (parent theme -- Bootstrap 5 FSE + Twig rendering)
  +-- wp-jobroom-theme (child theme -- overrides plugin templates with Bootstrap 5)
        +-- wp-jobroom (plugin -- provides post types, logic, base Twig templates)
```

Data flows upward: the plugin defines post types, business logic, and base templates; the child theme overrides those templates with Bootstrap 5 markup; the parent theme provides the Bootstrap framework and the outer page shell.

3.7.3 How It Works

1. **Plugin** (`wp-jobroom`): Registers custom post types (`jr_applicant`, `jr_company`, `jr_job_offer`, `jr_message`, `jr_application`), business logic (matching, subscriptions, helpdesk, analytics), and 95+ base Twig templates. All plugin templates use `jr-*` CSS classes with pure WordPress styling and no framework dependencies. This means the plugin works with any WordPress theme out of the box.
2. **Parent Theme** (`wp-bootstrap`): A standalone Full Site Editing (FSE) theme built on Bootstrap 5. It provides Bootstrap CSS and JS, Bootstrap Icons (web font), dark mode support via the `data-bs-theme` HTML attribute, and 15 color palette style variations. The parent theme also integrates Twig rendering via its own `TemplateController`.
3. **Child Theme** (`wp-jobroom-theme`): Inherits from `wp-bootstrap` via the WordPress `Template: wp-bootstrap` declaration in `style.css`. It overrides the plugin's base Twig templates to use Bootstrap 5 components (`card`, `btn`, `form-floating`, `table`, `offcanvas`, etc.). The child theme hooks into the plugin's Twig loader via the `TemplateOverride` class, which prepends its own `templates/` directory to the Twig `FilesystemLoader`.

3.7.4 Template Resolution Order

When the plugin renders a Twig template (e.g., `profiles/applicant/single.html.twig`):

1. **Theme templates** checked first: `wp-content/themes/wp-jobroom-theme/templates/` (highest priority)

2. **Plugin templates** as fallback: `wp-content/plugins/wp-jobroom/templates/`

The child theme's `TemplateOverride` class hooks into `init` at priority 20 (after plugin init at priority 0) and calls `prependPath()` on the Twig `FilesystemLoader` to insert the theme's template directory before the plugin's. This means any template file placed in the theme with the same relative path as a plugin template will automatically take precedence.

3.7.5 Template Contract (Plugin to Theme)

Templates receive context variables via `Template::render()` and `Template::render_themed_page()`. The plugin guarantees these variables are available for each page type:

Profile pages:

- `post` – the WordPress post object
- `meta` – all `_jr_*` post meta values
- `skills` – skill taxonomy terms with proficiency levels
- `matches` – matching results (scores, matched profiles)
- `is_own_profile` – boolean, whether the current user owns this profile
- `privacy` – per-section privacy levels and visibility flags
- `contacts` – array of contact persons (company profiles)
- `references` – array of personal references (applicant profiles)

Search pages:

- `filters` – active filter values
- `filter_options` – available options for each filter
- `filter_values` – current selected values
- `search_type` – `applicant`, `company`, or `job`
- `results` – array of matching post objects
- `total_results` – total result count
- `current_page` – current pagination page
- `max_pages` – total number of pages

Auth pages:

- `nonce` – WordPress nonce for form submission
- `page_title` – page heading text
- `form_action` – form target URL
- `errors` – array of validation error messages

Application pages:

- `application` – application post object
- `job_offer` – linked job offer
- `applicant` – linked applicant profile
- `status` – current application status
- `can_withdraw` – boolean, whether the applicant can withdraw

Message pages:

- `threads` – array of message thread summaries
- `messages` – messages in the current thread
- `thread_id` – active thread identifier
- `participants` – users involved in the thread
- `subject` – thread subject line

3.7.6 CSS Class Conventions

- Plugin templates use the `jr-*` prefix for all CSS classes (e.g., `jr-card`, `jr-search-form`, `jr-applicant-profile`)
- The child theme maps `jr-*` classes to Bootstrap 5 equivalents but preserves `jr-repeater__*` and `jr-skill-editor-*` classes because the plugin's JavaScript targets those selectors
- Body classes use the `jr-page-*` prefix (e.g., `jr-page-account`, `jr-page-messages`, `jr-page-dashboard`) to avoid collision with content CSS

3.7.7 JavaScript Globals

The plugin exposes these global objects for frontend JavaScript:

- `jrCommunication` – AJAX URLs, nonces, and i18n strings for the messaging and application systems
- `jrFrontend` – General frontend configuration (ajax_url, nonce, site-wide settings)
- `jrCookieConsent` – Cookie consent API with methods `hasConsent()`, `hasCategory()`, and `openSettings()`
- `jrEmojiPicker` – Emoji picker API with `init()` method for message textareas

3.7.8 Text Domain Rules

This is a critical area where mistakes cause silent translation failures:

- Use `'wp-jobroom'` for **all** strings that exist in the plugin's `.pot` file, including strings used in theme template overrides
- The plugin's `Template.php` registers `__()` and `_n()` as Twig functions with `'wp-jobroom'` as the default domain
- The child theme has its own text domain `'wp-jobroom-theme'` but `load_child_theme_textdomain()` does **not** resolve reliably at Twig render time
- To check which domain to use: `grep -c 'msgid "My String"' languages/wp-jobroom.pot` – if found, use `'wp-jobroom'`

3.7.9 Plugin Rendering Pipeline

The plugin's `Template::render_themed_page()` method handles page rendering with a 3-tier theme detection strategy:

1. **wp-bootstrap Twig**: If the parent theme is `wp-bootstrap`, delegates to the parent theme's Twig pipeline via `render_via_theme_twig()`, injecting plugin content as `post.content`
2. **Block theme**: If the active theme supports Full Site Editing, uses `block_template_part()` for the outer shell
3. **Classic theme**: Falls back to standard WordPress template loading with `get_header()` / `get_footer()`

Key integration points:

- The `_theme_wrapped` context flag is set to `true` when the parent theme provides the page shell. Templates check this flag to suppress the plugin's own wrapper, breadcrumbs, and sidebar (avoiding double rendering).
- Filter `wp_jobroom_render_page` – allows the child theme to intercept and handle page rendering before the plugin's default pipeline
- Filter `wp_jobroom_is_theme_wrapped` – allows the child theme to signal that it has already wrapped the content in a page shell

3.7.10 Plugin Twig Functions Available to Theme

These functions are registered in `Template.php` and available in all Twig templates (both plugin and theme overrides):

- `jr_applicant_location()`, `jr_company_location()` – formatted location strings
- `jr_get_cantons()` – Swiss canton list
- `jr_get_employment_types()`, `jr_get_remote_options()` – employment option arrays
- `jr_get_size_categories()` – company size categories
- `jr_get_benefits_list()` – available benefits
- `jr_get_languages()`, `jr_get_proficiency_levels()` – language and proficiency data
- `jr_get_availability_options()` – availability status options
- `jr_social_links( post_id )`, `jr_share_links( post_id )` – social media link arrays
- `url()` – alias for `home_url()`
- `do_shortcode( tag, atts )` – execute a WordPress shortcode

Twig filters: `|trans` (translation), `|pluralize` (pluralization), `|esc_html`, `|esc_attr`, `|esc_url`, `|number_format_i18n`.

3.7.11 Post Types and Meta Key Prefixes

Each custom post type uses a consistent meta key prefix:

Post Type	Meta Prefix
`jr_applicant`	`_jr_applicant_*`
`jr_company`	`_jr_company_*`
`jr_job_offer`	`_jr_job_offer_*`
`jr_message`	`_jr_message_*`
`jr_application`	`_jr_application_*`

3.7.12 Development Workflow

1. **One session per project.** Each session reads its own `CLAUDE.md` and stays within its own git repository. Do not mix changes across repos in a single session.
2. **Plugin changes first.** When a change affects templates consumed by the theme (new variables, renamed classes, new templates), make the plugin change first. Then switch to the theme session and update the overrides.
3. **Communicate changes explicitly.** When switching sessions, describe what changed upstream – new context variables, renamed CSS classes, new template files, changed data structures.
4. **Shared Docker environment.** All three projects are bind-mounted into the same Docker container (`jobroom-wordpress`), so changes are visible live without rebuilds.
5. **Clear Twig cache after template changes.** The plugin caches compiled Twig templates in `cache/twig/`. Delete this directory after structural template changes to force recompilation.
6. **Cross-project release order.** When a release touches both plugin and theme, always release the theme first, then the plugin (because the plugin's `Dockerfile` references the theme version).

3.8 Database Schema

3.8.1 Custom Post Types

Post Type	Slug	Description
Applicant	`jr_applicant`	Job seeker profiles
Company	`jr_company`	Employer profiles
Job Offer	`jr_job_offer`	Job listings
Message	`jr_message`	Internal messages
Application	`jr_application`	Job applications
Subscription	`jr_subscription`	Company subscriptions
Invoice	`jr_invoice`	Subscription invoices
Saved Search	`jr_saved_search`	User saved searches

3.8.2 Custom Taxonomies

Taxonomy	Slug	Description
Skills	`jr_skill`	Technical/soft skills
Occupations	`jr_occupation`	Job titles/roles
Industries	`jr_industry`	Business sectors

3.8.3 Meta Keys

Applicant Meta

Key	Type	Description
`_jr_applicant_email`	string	Contact email
`_jr_applicant_phone`	string	Phone number
`_jr_applicant_canton`	string	Swiss canton
`_jr_applicant_work_experience`	array	Work history; each entry includes an `achievements` key (array of strings)
`_jr_applicant_education`	array	Education history
`_jr_applicant_skills`	array	Skills with proficiency
`_jr_applicant_languages`	array	Language proficiency
`_jr_applicant_availability`	string	Availability status
`_jr_applicant_work_permit`	string	Work permit type
`_jr_applicant_certificates`	array	Certificates with attachment IDs, metadata, and related entries
`_jr_applicant_references`	array	Professional references (name, company, relationship, email, phone)

Company Meta

Key	Type	Description
`_jr_company_website`	string	Company website
`_jr_company_size`	string	Company size category
`_jr_company_canton`	string	Swiss canton
`_jr_company_benefits`	array	Selected benefits
`_jr_company_contacts`	array	Contact persons repeater (each: entry_id, name, position, email, phone)
`_jr_company_contact_*`	string	Legacy single contact fields (auto-migrated to `_jr_company_contacts`)

Job Offer Meta

Key	Type	Description
`_jr_job_offer_company_id`	int	Linked company post ID
`_jr_job_offer_contact_person_id`	string	Selected contact person entry_id from company contacts
`_jr_job_offer_position_type`	string	permanent/temporary/try_hire
`_jr_job_offer_employment_type`	string	full-time/part-time/etc
`_jr_job_offer_workload_min`	int	Min workload percentage
`_jr_job_offer_workload_max`	int	Max workload percentage
`_jr_job_offer_salary_*`	mixed	Salary configuration
`_jr_job_offer_skills`	array	Required skills
`_jr_job_offer_matches`	array	Cached match results

3.8.4 Custom Tables

Analytics Events Table

```
CREATE TABLE {prefix}jr_analytics_events (
    id bigint(20) UNSIGNED NOT NULL AUTO_INCREMENT,
    event_type varchar(50) NOT NULL,
    user_id bigint(20) UNSIGNED DEFAULT NULL,
    post_id bigint(20) UNSIGNED DEFAULT NULL,
    visitor_hash varchar(64) DEFAULT NULL,
    event_data longtext DEFAULT NULL,
    created_at datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (id),
    KEY event_type (event_type),
    KEY user_id (user_id),
    KEY post_id (post_id),
    KEY created_at (created_at)
);
```

3.9 Extending the Plugin

3.9.1 Adding Custom Meta Fields

```
add_action( 'add_meta_boxes', function() {
    add_meta_box(
        'my_custom_meta',
```

```
        __( 'Custom Fields', 'my-plugin' ),
        'render_my_custom_meta',
        'jr_applicant',
        'normal',
        'default'
    );
} );

function render_my_custom_meta( $post ) {
    $value = get_post_meta( $post->ID, '_my_custom_field', true );
    wp_nonce_field( 'my_custom_meta', 'my_custom_meta_nonce' );
    ?>
    <input type="text" name="my_custom_field" value="<?php echo
esc_attr( $value ); ?>">
    <?php
}

add_action( 'save_post_jr_applicant', function( $post_id ) {
    if ( ! wp_verify_nonce( $_POST['my_custom_meta_nonce'] ?? '',
'my_custom_meta' ) ) {
        return;
    }
    if ( isset( $_POST['my_custom_field'] ) ) {
        update_post_meta( $post_id, '_my_custom_field',
sanitize_text_field( $_POST['my_custom_field'] ) );
    }
} );
```

3.9.2 Adding Custom Matching Criteria

```
add_filter( 'wp_jobroom_match_criteria', function( $criteria ) {
    $criteria['certification'] = [
        'weight' => 5,
        'label' => __( 'Certifications', 'my-plugin' ),
    ];
    return $criteria;
} );

add_filter( 'wp_jobroom_match_score', function( $score, $applicant_id, $job_id ) {
    // Add certification matching logic
    $applicant_certs = get_post_meta( $applicant_id, '_certifications', true );
    $required_certs = get_post_meta( $job_id, '_required_certifications', true );

    // Calculate additional score
    $cert_score = calculate_certification_match( $applicant_certs,
$required_certs );

    return $score + ( $cert_score * 0.05 ); // 5% weight
}, 10, 3 );
```

3.9.3 Adding Custom REST Endpoints

```
add_action( 'rest_api_init', function() {
    register_rest_route( 'jobroom/v1', '/my-endpoint', [
        'methods' => 'GET',
        'callback' => 'my_endpoint_handler',
        'permission_callback' => function() {
            return current_user_can( 'read' );
        },
    ] );
} );

function my_endpoint_handler( $request ) {
    return new WP_REST_Response( [
        'success' => true,
        'data' => [ 'message' => 'Hello World' ],
    ] );
}
```

3.9.4 Adding Custom Shortcodes

```
add_shortcode( 'my_jr_shortcode', function( $atts ) {
    $atts = shortcode_atts( [
        'count' => 5,
    ], $atts );

    $template = \Magdev\WpJobroom\Plugin::get_instance()->get_template();

    return $template->render( 'my-template.html.twig', [
        'count' => intval( $atts['count'] ),
    ] );
} );
```

3.9.5 Built-in Shortcodes Reference

Shortcode	Description	Key Attributes
`[jr_pricing_table]`	Subscription pricing cards	—
`[jr_dashboard]`	User dashboard with stats	—
`[jr_newsletter_form]`	Full newsletter subscription form (card)	`title`, `description`, `button_text`, `consent_text`, `show_name` (true/false), `layout` (stacked/inline), `variant`, `class`
`[jr_newsletter_widget]`	Compact newsletter form for sidebars/embeds	`title`, `button_text`, `consent_text`, `layout` (wide), `variant`, `class`
`[jr_share_buttons]`	Social share buttons	`style` (horizontal/vertical/icons)
`[jr_cookie_settings]`	Cookie consent settings link	`text`, `class`, `type`
`[jr_platform_owner]`	Platform owner information	`format` (full/compact), `show_name` (yes/no)
`[jr_imprint]`	Legal imprint page content	—
`[jr_register_applicant]`	Applicant registration link/button	—
`[jr_register_company]`	Company registration link/button	—

Newsletter Shortcode Variants

Both `[jr_newsletter_form]` and `[jr_newsletter_widget]` support a `variant` attribute that applies a Bootstrap 5 `text-bg-*` contextual color to the surrounding card. Accepted values:

Value	Card background	Button style
(empty)	Default (theme-controlled)	`btn-primary`
`primary`	Blue	`btn-light`
`secondary`	Grey	`btn-light`
`success`	Green	`btn-light`
`danger`	Red	`btn-light`
`warning`	Yellow	`btn-dark`
`info`	Cyan	`btn-dark`
`light`	White	`btn-dark`
`dark`	Black	`btn-light`

The button contrast class is chosen automatically; no additional attributes are required.

```
[jr_newsletter_form variant="primary" layout="inline"]
[jr_newsletter_form variant="dark" title="Stay in the loop"]
[jr_newsletter_widget variant="success"]
[jr_newsletter_widget variant="warning" layout="wide"]
```

The `[jr_platform_owner]` and `[jr_imprint]` shortcodes read data from **Settings** > **General** > **Settings** > **Platform Owner** (9 configurable fields: company name, address, contact person, phone, email, VAT/UID, responsible person, commercial register, regulatory authority, DPO name/email).

Using `do_shortcode()` in Twig templates:

```
{# Execute a shortcode and render its output #}
{{ do_shortcode('jr_pricing_table') | raw }}

{# With attributes #}
{{ do_shortcode('jr_platform_owner', {format: 'compact'}) | raw }}
```

3.9.6 Adding Custom Gutenberg Blocks

```
add_action( 'init', function() {
    register_block_type( 'my-plugin/my-block', [
        'render_callback' => 'render_my_block',
        'attributes' => [
            'count' => [ 'type' => 'number', 'default' => 5 ],
        ],
    ] );
} );

function render_my_block( $attributes ) {
    return '<div class="my-block">Content here</div>';
}
```

3.10 Security Patterns

3.10.1 Input Sanitization

Always sanitize user input:

```
$email = sanitize_email( $_POST['email'] ?? '' );
$text = sanitize_text_field( $_POST['text'] ?? '' );
$html = wp_kses_post( $_POST['content'] ?? '' );
$int = absint( $_POST['count'] ?? 0 );
$array = array_map( 'sanitize_text_field', $_POST['items'] ?? [] );
```

3.10.2 Output Escaping

Always escape output:

```
echo esc_html( $text );           // Plain text
echo esc_attr( $attribute );      // HTML attributes
echo esc_url( $url );             // URLs
echo esc_js( $javascript );       // JavaScript strings
echo wp_kses_post( $html );       // Safe HTML
```

3.10.3 Nonce Verification

Protect forms and AJAX:

```
// In form
wp_nonce_field( 'my_action', 'my_nonce' );

// On submission
if ( ! wp_verify_nonce( $_POST['my_nonce'], 'my_action' ) ) {
    wp_die( 'Security check failed' );
}

// For AJAX
check_ajax_referer( 'my_ajax_action', 'nonce' );
```

3.10.4 Capability Checks

Verify user permissions:

```
if ( ! current_user_can( 'manage_options' ) ) {
    wp_die( 'Unauthorized' );
}

if ( ! current_user_can( 'edit_post', $post_id ) ) {
    wp_die( 'Cannot edit this post' );
}
```

Plugin-Specific Capabilities

Capability	Granted To	Description
`jr_manage_tickets`	`jr_supporter`, admins	Create, reply to, and manage helpdesk tickets
`jr_view_ticket_reports`	`jr_supporter`, admins	View helpdesk reports in admin

The `jr_supporter` role provides helpdesk-only admin access — supporters see only JobRoom > Tickets and JobRoom > Reports (Helpdesk tab).

```
if ( current_user_can( 'jr_manage_tickets' ) ) {
    // Show ticket management controls
}

if ( current_user_can( 'jr_view_ticket_reports' ) ) {
```

```
    // Show helpdesk analytics
}
```

3.10.5 Private File Storage

Sensitive uploads (certificates, ticket attachments) are stored in a protected directory inaccessible via direct HTTP requests. The `SecureFileHandler` class (`src/Security/SecureFileHandler.php`) manages the entire lifecycle:

Directory structure:

- `wp-content/uploads/jr-private/` — protected directory with `.htaccess` denying all access
- `.htaccess` contains `Deny from all` (Apache) or equivalent rules

Uploading private files:

```
use Magdev\WpJobroom\Security\SecureFileHandler;

$secure = SecureFileHandler::get_instance();

// Wrap any wp_handle_upload() / media_handle_upload() call:
$secure->enable_private_upload();
$attachment_id = media_handle_upload( 'file_field', $parent_post_id );
$secure->disable_private_upload();
```

When enabled, the `upload_dir` filter redirects uploads to the `jr-private/` subdirectory. Always call `disable_private_upload()` afterward — the filter is global.

Generating download URLs:

```
// Returns a nonce-protected URL through admin-post.php
$url = SecureFileHandler::get_download_url( $attachment_id );

// Check if an attachment is in the private directory
if ( SecureFileHandler::is_private_attachment( $attachment_id ) ) {
    // Use secure download URL
} else {
    // Use wp_get_attachment_url() for public files
}
```

Permission checks on download:

The `handle_download()` method verifies:

1. Valid nonce (`jr_download_{attachment_id}`)
2. Attachment exists and is in the private directory
3. User has permission: administrators always allowed; for others, checks post authorship (certificate owner, ticket creator, reply author)

Avatars and logos remain public — they use the standard WordPress uploads directory and `get_avatar_url()` .

3.10.6 Media Offloader Integration (Cloud Storage for Private Files)

When the Advanced Media Offloader plugin is installed, private files can optionally be offloaded to S3-compatible cloud storage (DigitalOcean Spaces, AWS S3, MinIO, etc.) with `private` ACL. The `MediaOffloaderIntegration` class (`src/Security/MediaOffloaderIntegration.php`) manages this:

How it works:

1. Files uploaded via `SecureFileHandler` flow through the standard WordPress media pipeline
2. The offloader's `wp_generate_attachment_metadata` hook auto-uploads them to cloud storage
3. The `advmo_object_acl` filter sets ACL to `private` (instead of `public-read`) for `jr-private/` files
4. The `wp_get_attachment_url` filter (priority 5, before offloader's 10) suppresses public cloud URL exposure
5. On download, `SecureFileHandler::handle_download()` generates a time-limited presigned URL after permission checks

Generating presigned URLs:

```
use Magdev\WpJobroom\Security\MediaOffloaderIntegration;

// Returns a presigned URL or null if not offloaded/not enabled
$url = MediaOffloaderIntegration::get_presigned_url( $attachment_id );

// Check if integration is active
if ( MediaOffloaderIntegration::is_enabled() ) {
    // Cloud storage is configured and enabled for private files
}
```

Configuration:

- Enable in **Settings → Integrations → Media Offloader**
- Presigned URL expiry: 1–60 minutes (default: 5)
- Requires a cloud provider configured in the Advanced Media Offloader plugin
- The integration tab only appears when the offloader plugin is detected

Fallback behavior:

When the integration is disabled or a file is not yet offloaded, the existing local `readfile()` serving path is used — no behavior change for existing installations.

3.10.7 SQL Safety

Use prepared statements:

```
global $wpdb;

// Safe query with prepare()
$results = $wpdb->get_results( $wpdb->prepare(
    "SELECT * FROM {$wpdb->posts} WHERE post_type = %s AND post_status = %s",
    'jr_applicant',
    'publish'
) );

// Safe insert
$wpdb->insert( $wpdb->prefix . 'jr_analytics_events', [
    'event_type' => $event_type,
```

```
    'user_id' => $user_id,
], [ '%s', '%d' ] );
```

3.11 Performance Optimization

3.11.1 Using QueryOptimizer

```
use Magdev\WpJobroom\Core\QueryOptimizer;

$optimizer = QueryOptimizer::get_instance();

// Prime post meta for multiple posts
$optimizer->prime_post_meta( $post_ids, [
    '_jr_applicant_skills',
    '_jr_applicant_canton',
] );

// Prime taxonomy terms
$optimizer->prime_post_terms( $post_ids, 'jr_skill' );

// Cached counts
$count = $optimizer->cached_post_count( 'jr_applicant', 'publish' );
```

3.11.2 Using SearchCache

```
use Magdev\WpJobroom\Core\SearchCache;

$cache = SearchCache::get_instance();

// Get or set cached results
$results = $cache->get_or_set( 'job_search', $filters, function() use ( $filters ) {
    return perform_search( $filters );
} );

// Invalidate cache
$cache->invalidate_search_type( 'job_search' );
```

3.11.3 Asset Optimization

```
use Magdev\WpJobroom\Core\AssetOptimizer;

$optimizer = AssetOptimizer::get_instance();

// Check if minification enabled
if ( $optimizer->is_optimization_enabled() ) {
```

```
    // Load minified asset
    wp_enqueue_style( 'jr-style', $optimizer->get_minified_url( 'style.css' ) );
}
```

3.12 Debug Bar Integration

WP JobRoom includes a custom Debug Bar panel that surfaces runtime debugging data. The integration has **zero production overhead** — it only initializes when `WP_DEBUG` is `true` and the Debug Bar plugin is active.

3.12.1 How It Works

Two classes power the integration:

- **`DebugCollector`** — A static class that accumulates data during the request lifecycle. All methods are no-ops when disabled (static boolean check).
- **`DebugBarPanel`** — Extends `Debug_Bar_Panel`, renders collected data as HTML tables in the "JobRoom" tab.

3.12.2 Initialization

The integration is conditionally initialized in `Plugin::init()`:

```
if ( defined( 'WP_DEBUG' ) && WP_DEBUG && class_exists( 'Debug_Bar_Panel' ) ) {
    DebugCollector::init();
    add_filter( 'debug_bar_panels', [ DebugBarPanel::class, 'register' ] );
}
```

When disabled, the PSR-4 autoloader never even loads the Debug classes.

3.12.3 Panel Sections

The Debug Bar panel displays 7 sections:

Section	Data Shown
Request Info	Matched route, action, page, method, URI, duration
Search & Solr	Solr status, queries with timing, fallback reasons
Query Optimizer	Post/user meta primed, terms primed, cache hits/misses
Templates	Twig templates rendered with timing, template errors
Search Cache	Cache hits/misses, lookup keys
Assets	Minification, concatenation, lazy loading, deferred scripts
Configuration	Solr, caching, search cache, match threshold settings

3.12.4 Adding Custom Debug Data

You can add your own data to the Debug Bar panel from anywhere in the codebase:

```
use Magdev\WpJobroom\Debug\DebugCollector;

// Log a key-value pair in a section
DebugCollector::log( 'my_section', 'my_key', 'my_value' );

// Append to an array in a section
DebugCollector::append( 'my_section', 'events', [ 'type' => 'something' ] );

// Increment a counter
DebugCollector::increment( 'my_section', 'api_calls' );

// Time an operation
DebugCollector::start_timer( 'my_operation' );
// ... do work ...
DebugCollector::stop_timer( 'my_operation' );
```

All calls are no-ops when Debug Bar is not active — no guard checks needed for timers.

3.12.5 Instrumented Components

Component	What's Tracked
`Frontend\Template`	Template render times, template errors
`Core\QueryOptimizer`	Post/user meta primed counts, cache hit/miss
`Solr\Search\SolrSearchAdapter`	Solr queries (entity type, keyword, results, time, fallback)
`Core\SearchCache`	Cache lookups with hit/miss per key
`Frontend\Router`	Matched route action and page

3.13 Development Setup

3.13.1 Requirements

- PHP 8.3+
- Composer 2.x
- WordPress 6.0+ (local installation)
- Git

3.13.2 Installation

```
# Clone the repository
git clone https://src.bundespruefstelle.ch/magdev/wp-jobroom.git
```

```
# Install dependencies
composer install

# Initialize submodules
git submodule update --init --recursive
```

3.13.3 Coding Standards

- Follow WordPress Coding Standards
- Use PSR-4 autoloading
- Document classes and methods with PHPDoc
- Use type hints where possible

3.13.4 Git Workflow

1. Work on `dev` branch
2. Create feature branches for large changes
3. Merge to `main` for releases
4. Tag releases with `vX.X.X` format

3.14 Testing

3.14.1 Running Tests

```
# Unit tests (when available)
composer test

# PHP CodeSniffer
composer phpcs
```

3.14.2 Manual Testing Checklist

- ☐ Registration flow (applicant and company)
- ☐ Profile creation and editing
- ☐ Search functionality
- ☐ Matching algorithm
- ☐ Application workflow
- ☐ Messaging system
- ☐ Subscription purchase
- ☐ REST API endpoints
- ☐ Admin settings

This documentation was generated for WP JobRoom v0.26.1. For user documentation, see the User Guide. For administrator documentation, see the Admin Guide.

4 Designer Guide

4.1 Overview

WP JobRoom uses **Twig 3.0** as its template engine, providing a clean separation between logic and presentation. All frontend templates are located in the `templates/` directory and can be overridden in your theme.

4.1.1 Key Concepts

- **Twig Templates**: HTML templates with Twig syntax for dynamic content
- **BEM Methodology**: CSS class naming follows Block-Element-Modifier pattern
- **CSS Custom Properties**: Design tokens for consistent theming
- **Component-Based**: Reusable components for cards, forms, buttons, etc.
- **WordPress Integration**: Full access to WordPress functions and filters

---

4.2 Template System

4.2.1 How Templates Are Loaded

Templates are loaded in the following priority order:

1. `{your-theme}/wp-jobroom/{template}` - Theme override
2. `{child-theme}/wp-jobroom/{template}` - Child theme override
3. `{plugin}/templates/{template}` - Plugin default

4.2.2 Basic Twig Syntax

```
{# This is a comment #}

{# Output a variable (auto-escaped) #}
{{ variable }}

{# Output with explicit escaping #}
{{ title|esc_html }}
{{ url|esc_url }}
{{ attribute|esc_attr }}

{# Conditionals #}
{% if condition %}
    <p>Condition is true</p>
```

```
{% elseif other_condition %}
    <p>Other condition is true</p>
{% else %}
    <p>Neither condition is true</p>
{% endif %}

{# Loops #}
{% for item in items %}
    <li>{{ item.name }}</li>
{% else %}
    <li>No items found</li>
{% endfor %}

{# Include another template #}
{% include 'components/card.html.twig' with {'title': 'My Card'} %}

{# Extend a base template #}
{% extends 'layouts/single.html.twig' %}

{% block content %}
    <p>Override the content block</p>
{% endblock %}
```

4.3 Template Structure

4.3.1 Directory Layout

```
templates/
├── base.html.twig              # Root base template
├── layouts/                    # Layout templates
│   ├── single.html.twig        # Single post/profile layout
│   ├── archive.html.twig       # Archive/list layout
│   └── form.html.twig          # Form page layout
├── profiles/                   # Profile templates
│   ├── applicant/
│   │   ├── single.html.twig    # Full applicant profile
│   │   ├── card.html.twig      # Applicant card component
│   │   └── edit.html.twig      # Profile edit form
│   ├── company/
│   │   ├── single.html.twig    # Full company profile
│   │   ├── card.html.twig      # Company card component
│   │   └── edit.html.twig      # Company edit form
│   ├── job-offer/
│   │   ├── single.html.twig    # Full job offer page
│   │   └── card.html.twig      # Job offer card component
│   └── privacy.html.twig       # Privacy settings
├── search/                     # Search templates
│   ├── results.html.twig       # Search results page
```

```
│   ├── filters.html.twig       # Search filter sidebar
│   ├── applicants.html.twig    # Applicant search
│   ├── companies.html.twig     # Company search
│   ├── jobs.html.twig          # Job search
│   ├── saved-searches.html.twig
│   └── save-search-modal.html.twig
├── components/                 # Reusable components
│   ├── card.html.twig          # Generic card
│   ├── pagination.html.twig    # Pagination links
│   └── privacy-settings.html.twig
├── auth/                       # Authentication templates
│   ├── register-applicant.html.twig
│   ├── register-company.html.twig
│   ├── verify-email.html.twig
│   └── reset-password.html.twig
├── messages/                   # Messaging templates
│   ├── inbox.html.twig
│   ├── conversation.html.twig
│   ├── compose.html.twig
│   └── message-item.html.twig
├── applications/               # Application templates
│   ├── list.html.twig          # Company's received applications
│   ├── history.html.twig       # Applicant's application history
│   ├── apply-form.html.twig
│   ├── status-badge.html.twig
│   └── error.html.twig
├── dashboard/                  # User dashboard
│   ├── overview.html.twig
│   ├── analytics.html.twig
│   ├── admin.html.twig         # Admin platform overview (administrator role)
│   └── support.html.twig       # Support ticket queue (jr_supporter role)
├── account/                    # Account settings
│   ├── notification-settings.html.twig
│   ├── data-export.html.twig
│   └── api-keys.html.twig
├── subscription/               # Subscription pages
│   ├── pricing.html.twig
│   ├── checkout.html.twig
│   ├── manage.html.twig
│   ├── invoices.html.twig
│   ├── success.html.twig
│   ├── cancel.html.twig
│   └── upgrade-prompt.html.twig
├── emails/                     # Email templates (HTML)
│   ├── base.html.twig
│   ├── new-message.html.twig
│   ├── application-received.html.twig
│   ├── application-status-changed.html.twig
│   ├── new-matching-jobs.html.twig
│   └── new-matching-applicants.html.twig
├── marketing/                  # Marketing components
│   ├── newsletter/
│   │   ├── subscribe-form.html.twig   # Full form; accepts variant, layout, title,
description, show_name, class
│   │   └── subscribe-widget.html.twig # Compact widget; accepts variant, layout,
```

```
title, class
│   └── social/
│       ├── profile-links.html.twig
│       └── share-buttons.html.twig
├── export/                     # PDF export templates
│   ├── applicant-pdf.html.twig
│   ├── company-pdf.html.twig
│   └── job-offer-pdf.html.twig
├── gdpr/                       # GDPR components
│   ├── cookie-banner.html.twig
│   └── cookie-settings-modal.html.twig
├── security/                   # Security pages
│   ├── settings.html.twig
│   └── mfa-verify.html.twig
├── blocks/                     # Gutenberg block templates
│   ├── recent-offers.html.twig
│   ├── new-applicants.html.twig
│   ├── related-offers.html.twig
│   └── search-form.html.twig
├── shortcodes/                 # Shortcode templates
│   ├── recent-offers.html.twig
│   ├── recent-applicants.html.twig
│   ├── companies.html.twig
│   └── search-form.html.twig
└── widgets/                    # Sidebar widget templates
    ├── recent-offers.html.twig
    ├── quick-search.html.twig
    └── featured-companies.html.twig
```

4.3.2 Template Inheritance

The template hierarchy uses Twig's `extends` and `block` system:

```
{# base.html.twig - Root template #}
<div class="jr-wrapper">
    {% block notifications %}...{% endblock %}
    <main class="jr-main">
        {% block breadcrumbs %}...{% endblock %}
        {% block content %}{% endblock %}
    </main>
    {% block sidebar %}{% endblock %}
</div>
{% block scripts %}{% endblock %}
```

```
{# layouts/single.html.twig - Extends base #}
{% extends "base.html.twig" %}

{% block content %}
    <article class="jr-single jr-single--{{ post_type }}">
        {% block article_header %}...{% endblock %}
        {% block article_content %}...{% endblock %}
        {% block article_footer %}{% endblock %}
```

```
    </article>
{% endblock %}
```

```
{# profiles/applicant/single.html.twig - Extends single layout #}
{% extends "layouts/single.html.twig" %}

{% block article_content %}
    {# Applicant-specific content #}
{% endblock %}
```

4.3.3 Available Blocks

Block	Description	Parent Template
`head`	Before wrapper	`base.html.twig`
`notifications`	Flash messages	`base.html.twig`
`breadcrumbs`	Breadcrumb navigation	`base.html.twig`
`content`	Main content area	`base.html.twig`
`sidebar`	Sidebar area	`base.html.twig`
`scripts`	Before closing body	`base.html.twig`
`article_header`	Article header	`layouts/single.html.twig`
`article_meta`	Article metadata	`layouts/single.html.twig`
`article_actions`	Action buttons	`layouts/single.html.twig`
`article_content`	Article body	`layouts/single.html.twig`
`article_footer`	Article footer	`layouts/single.html.twig`

4.4 CSS Architecture

4.4.1 CSS Custom Properties

All colors, spacing, and typography are defined as CSS custom properties in `:root`:

```
:root {
    /* Colors */
    --jr-primary-color: #2563eb;
    --jr-primary-hover: #1d4ed8;
    --jr-secondary-color: #64748b;
    --jr-success-color: #22c55e;
```

```
    --jr-error-color: #ef4444;
    --jr-warning-color: #f59e0b;
    --jr-info-color: #3b82f6;

    /* Text colors */
    --jr-text-color: #1e293b;
    --jr-text-light: #64748b;
    --jr-text-muted: #94a3b8;

    /* Background colors */
    --jr-bg-color: #ffffff;
    --jr-bg-light: #f8fafc;
    --jr-bg-muted: #f1f5f9;

    /* Borders */
    --jr-border-color: #e2e8f0;
    --jr-border-radius: 8px;
    --jr-border-radius-sm: 4px;
    --jr-border-radius-lg: 12px;

    /* Shadows */
    --jr-shadow-sm: 0 1px 2px 0 rgb(0 0 0 / 0.05);
    --jr-shadow: 0 4px 6px -1px rgb(0 0 0 / 0.1);
    --jr-shadow-lg: 0 10px 15px -3px rgb(0 0 0 / 0.1);

    /* Typography */
    --jr-font-family: system-ui, -apple-system, BlinkMacSystemFont, 'Segoe UI',
Roboto, sans-serif;
    --jr-font-size-sm: 0.875rem;
    --jr-font-size-base: 1rem;
    --jr-font-size-lg: 1.125rem;
    --jr-font-size-xl: 1.25rem;
    --jr-font-size-2xl: 1.5rem;

    /* Spacing */
    --jr-spacing-xs: 0.25rem;
    --jr-spacing-sm: 0.5rem;
    --jr-spacing-md: 1rem;
    --jr-spacing-lg: 1.5rem;
    --jr-spacing-xl: 2rem;
    --jr-spacing-2xl: 3rem;

    /* Layout */
    --jr-max-width: 1200px;
    --jr-content-width: 720px;
}
```

4.4.2 Overriding CSS Variables

To customize the appearance in your theme, override the CSS variables:

```
/* In your theme's style.css */
:root {
    --jr-primary-color: #8b5cf6;      /* Purple instead of blue */
```

```
    --jr-primary-hover: #7c3aed;
    --jr-border-radius: 4px;           /* Sharper corners */
    --jr-font-family: 'Inter', sans-serif;
}
```

4.4.3 BEM Naming Convention

All CSS classes follow the BEM (Block-Element-Modifier) methodology:

```
/* Block */
.jr-card { }

/* Element (part of block) */
.jr-card__header { }
.jr-card__body { }
.jr-card__footer { }
.jr-card__title { }
.jr-card__meta { }

/* Modifier (variation) */
.jr-card--applicant { }
.jr-card--company { }
.jr-card--job-offer { }
.jr-card--small { }
.jr-card--large { }
```

4.4.4 Common CSS Prefixes

All JobRoom CSS classes use the `jr-` prefix to avoid conflicts:

Prefix	Purpose
`jr-`	Base prefix for all classes
`jr-wrapper`	Container wrapper
`jr-main`	Main content area
`jr-card`	Card components
`jr-button`	Button elements
`jr-form`	Form elements
`jr-single`	Single post pages
`jr-search`	Search components
`jr-notification`	Notifications
`jr-badge`	Status badges
`jr-avatar`	User avatars
`jr-icon`	SVG icons
`jr-modal`	Modal dialogs
`jr-skill-tag`	Skill badges
`jr-match-score`	Match score displays

4.5 Component Library

4.5.1 Cards

Cards are used for displaying profiles, job offers, and other items in lists:

```
<article class="jr-card jr-card--{{ type }} jr-card--{{ size }}">
    <a href="{{ permalink }}" class="jr-card__link">
        <div class="jr-card__header">
            <div class="jr-card__avatar">...</div>
            {% if match_score %}
                <span class="jr-card__score jr-match-score">{{ match_score }}%</
span>
            {% endif %}
        </div>
        <div class="jr-card__body">
            <h3 class="jr-card__title">{{ title }}</h3>
            <p class="jr-card__subtitle">{{ subtitle }}</p>
            <div class="jr-card__meta">...</div>
            <div class="jr-card__skills">...</div>
```

```
        </div>
        <div class="jr-card__footer">...</div>
    </a>
</article>
```

Card Modifiers:

- `jr-card--applicant` - Applicant profile card
- `jr-card--company` - Company profile card
- `jr-card--job-offer` - Job offer card
- `jr-card--small` - Compact size
- `jr-card--medium` - Default size
- `jr-card--large` - Large size

4.5.2 Buttons

```
{# Primary button #}
<button class="jr-button jr-button--primary">
    {{ __('Submit') }}
</button>

{# Secondary button #}
<a href="{{ url }}" class="jr-button jr-button--secondary">
    {{ __('Cancel') }}
</a>

{# Text button #}
<button class="jr-button jr-button--text">
    {{ __('Learn More') }}
</button>

{# Large button #}
<button class="jr-button jr-button--primary jr-button--large">
    {{ __('Sign Up') }}
</button>

{# Loading state #}
<button class="jr-button jr-button--primary jr-button--loading" disabled>
    {{ __('Saving...') }}
</button>
```

Button Modifiers:

- `jr-button--primary` - Primary action (filled)
- `jr-button--secondary` - Secondary action (outlined)
- `jr-button--text` - Text-only button
- `jr-button--large` - Large size
- `jr-button--loading` - Loading state

4.5.3 Forms

```
<div class="jr-form-page">
    <div class="jr-form-page__header">
        <h1 class="jr-form-page__title">{{ title }}</h1>
        <p class="jr-form-page__description">{{ description }}</p>
    </div>

    <div class="jr-form-page__content">
        <form class="jr-form" method="post">
            <fieldset class="jr-form__fieldset">
                <legend class="jr-form__legend">{{ section_title }}</legend>

                <div class="jr-form__row">
                    <div class="jr-form__group">
                        <label class="jr-form__label" for="field">
                            {{ __('Field Label') }}
                            <span class="jr-form__required">*</span>
                        </label>
                        <input type="text"
                               id="field"
                               name="field"
                               class="jr-form__input"
                               required>
                        <p class="jr-form__help">{{ __('Help text here') }}</p>
                    </div>
                </div>
            </fieldset>

            <div class="jr-form__actions">
                <button type="submit" class="jr-button jr-button--primary">
                    {{ __('Save') }}
                </button>
            </div>
        </form>
    </div>
</div>
```

Form Elements:

- `.jr-form__input` - Text inputs
- `.jr-form__select` - Select dropdowns
- `.jr-form__textarea` - Textareas
- `.jr-form__checkbox` - Checkbox inputs
- `.jr-form__radio` - Radio inputs
- `.jr-form__label` - Labels
- `.jr-form__help` - Help text
- `.jr-form__error` - Error messages
- `.jr-form__required` - Required indicator

4.5.4 Badges

```
{# Status badges #}
<span class="jr-badge jr-badge--success">{{ __('Available now') }}</span>
```

```
<span class="jr-badge jr-badge--warning">{{ __('Pending') }}</span>
<span class="jr-badge jr-badge--error">{{ __('Rejected') }}</span>
<span class="jr-badge jr-badge--info">{{ __('In Review') }}</span>

{# Skill tags #}
<span class="jr-skill-tag">JavaScript</span>
<span class="jr-skill-tag jr-skill-tag--more">+5</span>
```

4.5.5 Match Score

```
<span class="jr-match-score jr-match-score--high">92%</span>   {# >= 80% #}
<span class="jr-match-score jr-match-score--medium">67%</span> {# >= 60% #}
<span class="jr-match-score">45%</span>                         {# < 60% #}
```

4.5.6 Avatars

```
{# With image #}
<img src="{{ avatar_url }}" alt="{{ name }}" class="jr-avatar">

{# Placeholder with initials #}
<div class="jr-avatar jr-avatar--placeholder">
    {{ first_initial }}{{ last_initial }}
</div>
```

4.5.7 Certificates

Certificate components are used on the profile edit form and public profile display:

```
{# Edit form - certificate card #}
<div class="jr-certificate-card">
    <div class="jr-certificate-card__header">
        <div class="jr-certificate-card__preview">
            <img class="jr-certificate-card__thumb" src="{{ thumbnail }}" alt="">
        </div>
        <button type="button" class="jr-certificate-card__remove">&times;</button>
    </div>
    <div class="jr-certificate-card__body">
        <input type="text" class="jr-form__input" placeholder="{{ __('Certificate
Title') }}">
        <input type="text" class="jr-form__input" placeholder="{{ __('Issuer') }}">
    </div>
</div>

{# Public profile - certificate display #}
<div class="jr-certificates-grid">
    <div class="jr-certificate-display">
        <div class="jr-certificate-display__icon">
            <span class="jr-certificate-card__icon jr-certificate-card__icon--
pdf">PDF</span>
        </div>
```

```
        <div class="jr-certificate-display__info">
            <p class="jr-certificate-display__title">{{ title }}</p>
            <p class="jr-certificate-display__issuer">{{ issuer }}</p>
            <p class="jr-certificate-display__dates">{{ issued }} – {{ expiry }}</p>
            <div class="jr-certificate-display__related">
                <span class="jr-certificate-display__tag">Related Entry</span>
            </div>
            <a href="{{ file_url }}" class="jr-certificate-
display__download">{{ __('Download') }}</a>
        </div>
    </div>
</div>
```

Certificate Upload Zone:

- `.jr-certificate-dropzone` - Dashed border upload area with drag-and-drop
- `.jr-certificate-dropzone--active` - Active drag state
- `.jr-certificate-upload-status--success/--error/--info` - Status messages

4.5.8 Notifications

```
<div class="jr-notification jr-notification--success">
    {{ message }}
    <button type="button" class="jr-notification__close">&times;</button>
</div>

<div class="jr-notification jr-notification--error">...</div>
<div class="jr-notification jr-notification--warning">...</div>
<div class="jr-notification jr-notification--info">...</div>
```

4.6 Twig Reference

4.6.1 Available Functions

WordPress Core Functions

Function	Description	Example
`__('text')`	Translate text	`{{ __('Save Changes') }}`
`_n(single, plural, n)`	Plural translation	`{{ _n('item', 'items', count) }}`
`sprintf(format, ...)`	String formatting	`{{ sprintf(__('Hello, %s'), name) }}`
`esc_html(value)`	HTML escape	`{{ esc_html(title) }}`
`esc_attr(value)`	Attribute escape	`{{ esc_attr(value) }}`
`esc_url(url)`	URL escape	`{{ esc_url(link) }}`
`wp_nonce_field(action)`	CSRF nonce field	`{{ wp_nonce_field('jr_save')|raw }}`
`home_url(path)`	Get home URL	`{{ home_url('/contact/') }}`
`admin_url(path)`	Get admin URL	`{{ admin_url('admin-ajax.php') }}`
`get_permalink(id)`	Get post permalink	`{{ get_permalink(post_id) }}`
`is_user_logged_in()`	Check login status	`{% if is_user_logged_in() %}...{% endif %}`
`current_user_can(cap)`	Check capability	`{% if current_user_can('edit_post') %}...{% endif %}`
`get_current_user_id()`	Get current user ID	`{{ get_current_user_id() }}`
`get_avatar_url(id)`	Get user avatar URL	`{{ get_avatar_url(user_id) }}`
`get_the_post_thumbnail_url(id)`	Featured image URL	`{{ get_the_post_thumbnail_url(post_id) }}`
`has_post_thumbnail(id)`	Check featured image	`{% if has_post_thumbnail(post_id) %}...{% endif %}`
`get_post_meta(id, key, single)`	Get post meta	`{{ get_post_meta(post_id, '_jr_applicant_email', true) }}`
`get_the_title(id)`	Get post title	`{{ get_the_title(company_id) }}`
`date_i18n(format, timestamp)`	Format date	`{{ date_i18n('F j, Y', timestamp) }}`
`human_time_diff(from, to)`	Human readable time	`{{ human_time_diff(created_at) }} ago`
`number_format_i18n(num, dec)`	Format number	`{{ number_format_i18n(1234.56, 2) }}`

JobRoom-Specific Functions

Function	Description	Example
`jr_get_cantons()`	Swiss cantons array	`{{ jr_get_cantons() [canton_code] }}`
`jr_get_countries()`	Countries array	`{{ jr_get_countries() [country_code] }}`
`jr_get_employment_types()`	Employment types	`{{ jr_get_employment_types() [type] }}`
`jr_get_remote_options()`	Remote work options	`{{ jr_get_remote_options() [preference] }}`
`jr_get_availability_options()`	Availability options	`{{ jr_get_availability_options() [status] }}`
`jr_get_languages()`	Languages array	`{{ jr_get_languages() [lang_code] }}`
`jr_get_proficiency_levels()`	Skill proficiency levels	`{{ jr_get_proficiency_levels() [level] }}`
`jr_get_benefits_list()`	Company benefits	`{{ jr_get_benefits_list() [benefit_key] }}`
`jr_get_size_categories()`	Company size options	`{{ jr_get_size_categories() [size] }}`
`jr_get_experience_levels()`	Experience levels	`{{ jr_get_experience_levels() [level] }}`
`jr_get_importance_levels()`	Skill importance levels	`{{ jr_get_importance_levels() [level] }}`
`jr_get_work_permit_types()`	Work permit types	`{{ jr_get_work_permit_types() [permit] }}`
`jr_is_applicant()`	Check if user is applicant	`{% if jr_is_applicant() %}...{% endif %}`
`jr_is_company()`	Check if user is company	`{% if jr_is_company() %}...{% endif %}`
`jr_applicant_location(id)`	Formatted location	`{{ jr_applicant_location(post_id) }}`
`jr_company_location(id)`	Formatted location	`{{ jr_company_location(post_id) }}`
`jr_social_links(id)`	Get social media links	`{% for link in jr_social_links(post_id) %}...{% endfor %}`
`jr_share_links(id)`	Get share button links	`{% for link in jr_share_links(post_id) %}...{% endfor %}`

4.6.2 Available Filters

Filter	Description	Example
`esc_html`	HTML escape	`{{ title|esc_html }}`
`esc_attr`	Attribute escape	`{{ value|esc_attr }}`
`esc_url`	URL escape	`{{ link|esc_url }}`
`esc_textarea`	Textarea escape	`{{ text|esc_textarea }}`
`wp_kses_post`	Allow safe HTML	`{{ content|wp_kses_post }}`
`wpautop`	Add paragraphs	`{{ text|wpautop }}`
`wptexturize`	Smart quotes	`{{ text|wptexturize }}`
`wp_trim_words`	Trim to word count	`{{ text|wp_trim_words(20) }}`
`date_i18n`	Format date	`{{ date|date_i18n('F j, Y') }}`
`human_time_diff`	Time ago	`{{ date|human_time_diff }}`
`number_format_i18n`	Format number	`{{ number|number_format_i18n(2) }}`
`translate`	Translate string	`{{ 'text'|translate }}`
`currency`	Format currency	`{{ amount|currency('CHF') }}`
`workload`	Format workload range	`{{ min|workload(max) }}`
`in_array`	Check array membership	`{% if item|in_array(list) %}...{% endif %}`

4.6.3 Global Variables

These variables are available in all templates:

Variable	Description	Example
`site_name`	Site title	`{{ site_name }}`
`site_url`	Home URL	`{{ site_url }}`
`ajax_url`	Admin AJAX URL	`{{ ajax_url }}`
`plugin_url`	Plugin URL	`{{ plugin_url }}`
`plugin_version`	Plugin version	`{{ plugin_version }}`
`is_rtl`	RTL text direction	`{% if is_rtl %}...{% endif %}`
`current_url`	Current page URL	`{{ current_url }}`

4.7 Template Overriding

4.7.1 Creating Theme Overrides

1. Create a `wp-jobroom` directory in your theme:

```
your-theme/
└── wp-jobroom/
    └── profiles/
        └── applicant/
            └── card.html.twig
```

2. Copy the original template from the plugin:

```
cp wp-content/plugins/wp-jobroom/templates/profiles/applicant/card.html.twig \
   wp-content/themes/your-theme/wp-jobroom/profiles/applicant/card.html.twig
```

3. Modify the copied template as needed.

4.7.2 Partial Overrides

You can override just the blocks you need by extending the original:

```
{# your-theme/wp-jobroom/profiles/applicant/single.html.twig #}

{% extends '@plugin/templates/profiles/applicant/single.html.twig' %}

{% block article_footer %}
    {# Your custom footer content #}
    <div class="custom-cta">
        <h3>{{ __('Interested?') }}</h3>
        <a href="{{ contact_url }}" class="jr-button jr-button--primary">
            {{ __('Contact this applicant') }}
        </a>
    </div>
{% endblock %}
```

4.7.3 Override Priority

1. Child theme: `wp-content/themes/child-theme/wp-jobroom/`
2. Parent theme: `wp-content/themes/parent-theme/wp-jobroom/`
3. Plugin: `wp-content/plugins/wp-jobroom/templates/`

4.8 Responsive Design

4.8.1 Breakpoints

The plugin uses these standard breakpoints:

```
/* Mobile first approach */

/* Small devices (phones, 576px and up) */
@media (min-width: 576px) { }

/* Medium devices (tablets, 768px and up) */
@media (min-width: 768px) { }

/* Large devices (desktops, 992px and up) */
@media (min-width: 992px) { }

/* Extra large devices (large desktops, 1200px and up) */
@media (min-width: 1200px) { }
```

4.8.2 Grid System

Cards use CSS Grid for responsive layouts:

```
.jr-card-grid {
    display: grid;
    grid-template-columns: repeat(auto-fill, minmax(280px, 1fr));
    gap: var(--jr-spacing-lg);
}

/* Tablet - 2 columns */
@media (min-width: 768px) {
    .jr-card-grid--2 {
        grid-template-columns: repeat(2, 1fr);
    }
}

/* Desktop - 3 columns */
@media (min-width: 992px) {
    .jr-card-grid--3 {
        grid-template-columns: repeat(3, 1fr);
    }
}
```

4.8.3 Responsive Images

Always use `loading="lazy"` for images below the fold:

```
<img src="{{ image_url }}"
     alt="{{ alt_text|esc_attr }}"
```

```
     class="jr-card__image"
     loading="lazy"
     decoding="async">
```

4.9 Accessibility Guidelines

4.9.1 WCAG 2.1 Compliance

All templates should follow WCAG 2.1 Level AA guidelines:

Color Contrast

- Text must have a contrast ratio of at least 4.5:1
- Large text (18pt+) must have at least 3:1
- Use the CSS custom properties which are designed for accessibility

Keyboard Navigation

```
{# All interactive elements must be keyboard accessible #}
<button type="button"
        class="jr-button"
        tabindex="0"
        role="button">
    {{ __('Click me') }}
</button>

{# Skip links for main content #}
<a href="#main-content" class="jr-skip-link">
    {{ __('Skip to main content') }}
</a>
```

ARIA Labels

```
{# Provide context for screen readers #}
<nav aria-label="{{ __('Pagination') }}">
    <a href="{{ prev_url }}" aria-label="{{ __('Previous page') }}">
        &laquo;
    </a>
</nav>

{# Form field associations #}
<label for="email">{{ __('Email') }}</label>
<input type="email" id="email" name="email" aria-describedby="email-help">
<p id="email-help" class="jr-form__help">{{ __('We will never share your email.') }}
</p>
```

Focus Indicators

```
/* Visible focus states */
.jr-button:focus {
    outline: 2px solid var(--jr-primary-color);
    outline-offset: 2px;
}

/* Never remove focus indicators */
*:focus {
    outline: 2px solid var(--jr-primary-color);
    outline-offset: 2px;
}
```

Screen Reader Text

```
{# Visually hidden but accessible to screen readers #}
<span class="jr-sr-only">{{ __('Opens in new tab') }}</span>
```

```
.jr-sr-only {
    position: absolute;
    width: 1px;
    height: 1px;
    padding: 0;
    margin: -1px;
    overflow: hidden;
    clip: rect(0, 0, 0, 0);
    white-space: nowrap;
    border: 0;
}
```

4.10 Best Practices

4.10.1 Template Best Practices

1. **Always escape output**:

   ```
   {{ title|esc_html }}
   {{ url|esc_url }}
   {{ attribute|esc_attr }}
   ```

2. **Use translation functions**:

   ```
   {{ __('Static text') }}
   {{ sprintf(__('Hello, %s'), name|esc_html) }}
   ```

3. **Check for empty values**:

```
{% if skills is defined and skills|length > 0 %}
    {# render skills #}
{% endif %}
```

4. **Use semantic HTML**:

```
<article class="jr-card">
    <header class="jr-card__header">...</header>
    <main class="jr-card__body">...</main>
    <footer class="jr-card__footer">...</footer>
</article>
```

5. **Provide fallbacks**:

```
{{ title|default(__('Untitled')) }}
{{ avatar|default(plugin_url ~ 'assets/img/default-avatar.png') }}
```

4.10.2 CSS Best Practices

1. **Use CSS custom properties** for theming
2. **Follow BEM naming** for maintainability
3. **Avoid** `!important` - use specificity instead
4. **Mobile-first** responsive design
5. **Test in multiple browsers**

4.10.3 Performance Best Practices

1. **Lazy load images** below the fold
2. **Minimize template complexity**
3. **Use** `include` **for reusable components**
4. **Avoid deep nesting** (max 3 levels)
5. **Cache expensive operations**

4.10.4 Security Best Practices

1. **Never output raw user input** without escaping
2. **Use** `|raw` **sparingly** - only for pre-sanitized content
3. **Always include nonces** in forms
4. **Validate data** before display

4.11 Resources

4.11.1 Further Reading

- Twig Documentation
- WordPress Template Hierarchy
- BEM Methodology

- WCAG 2.1 Guidelines

4.11.2 Related Documentation

- Developer Guide - Technical implementation details
- Admin Guide - Plugin configuration
- User Guide - End-user documentation

---

5 Docker Deployment Guide

5.1 Prerequisites

- Docker Engine 24.0+
- Docker Compose v2.20+
- Git (for cloning the repository)

5.2 Service Architecture

The stack consists of 11 services organized into 3 profiles:

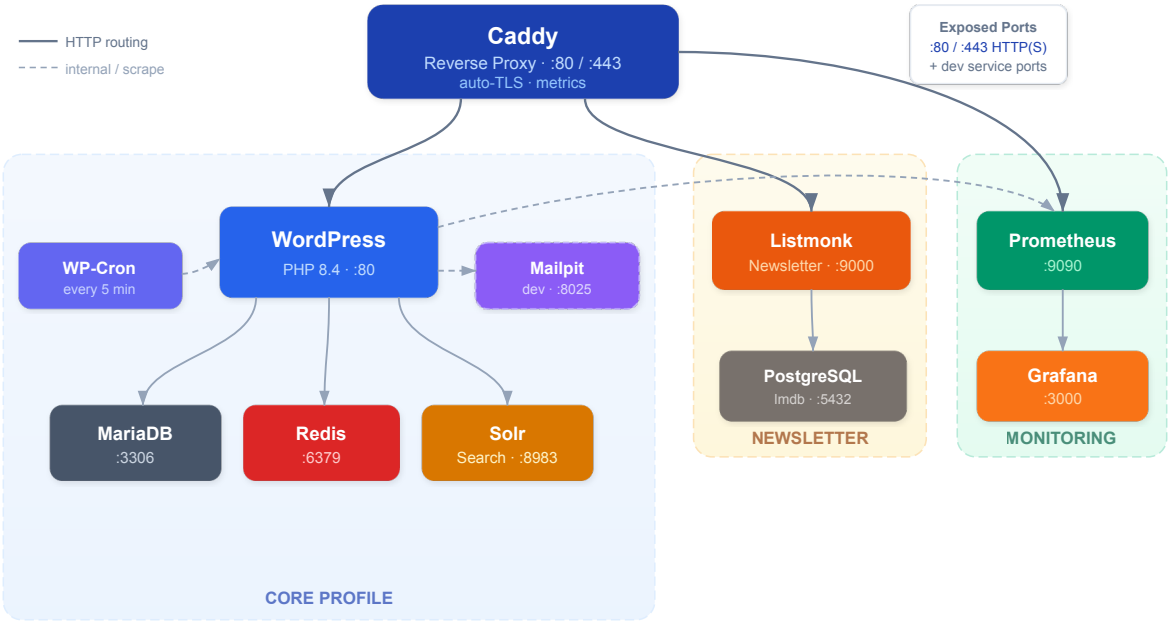

Figure 2: Service Architecture

5.2.1 Service Profiles

Services are grouped into profiles for selective startup:

Profile	Services	Description
`core`	caddy, wordpress, db, redis, search, cron	WordPress stack with reverse proxy, search, and WP-Cron
`newsletter`	listmonk, lmdb	Newsletter management with Listmonk
`monitoring`	prometheus, grafana	Prometheus metrics collection and Grafana dashboards
`fullstack`	All of the above	Convenience profile to start the entire stack at once

Usage examples:

```
# Core services only
docker compose --profile core up -d

# Core + Newsletter
docker compose --profile core --profile newsletter up -d

# Core + Monitoring
docker compose --profile core --profile monitoring up -d

# Everything (explicit profiles)
docker compose --profile core --profile newsletter --profile monitoring up -d

# Everything (shorthand via fullstack profile)
docker compose --profile fullstack up -d
```

> **Note**: In development, `compose.override.yaml` is applied automatically. The `mailer` service (Mailpit) is included in both `core` and `newsletter` profiles for email testing.

5.2.2 Service Table

Service	Image	Default Port	Profile	Purpose
`caddy`	Custom Dockerfile	80, 443	core	Reverse proxy with auto-TLS and Prometheus metrics
`wordpress`	Custom Dockerfile	80 (9080 dev)	core	WordPress with PHP extensions, Redis, WP-CLI
`db`	`mariadb:latest`	3306	core	WordPress database
`redis`	`redis:alpine`	6379	core	Object cache with persistent storage
`search`	`solr:latest`	8983	core	Full-text search engine (pre-creates `jobroom` core)
`cron`	`whefter/cron`	—	core	WP-Cron trigger (curl `wp-cron.php` every 5 minutes)
`listmonk`	`listmonk/listmonk:latest`	9000	newsletter	Newsletter management
`lmdb`	`postgres:17-alpine`	5432	newsletter	Listmonk database
`mailer`	`axllent/mailpit:latest`	1025, 8025	core (dev)	Mail catcher (development only)
`prometheus`	Custom Dockerfile	9090	monitoring	Metrics collection with WordPress and Caddy scraping
`grafana`	Custom Dockerfile	3000	monitoring	Dashboard visualization with pre-configured JobRoom dashboard

5.2.3 Network Segmentation

The stack uses four isolated Docker networks to minimize the attack surface:

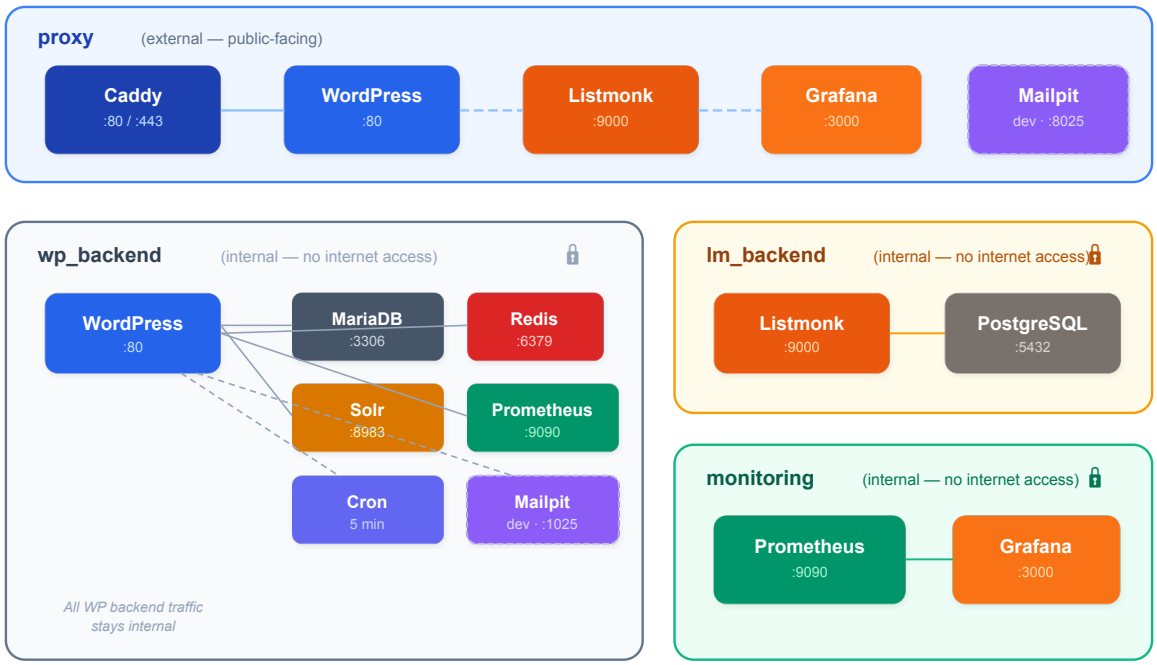

Figure 3: Network Segmentation

Network	Internal	Services	Purpose
`proxy`	No	Caddy, WordPress, Listmonk, Grafana, Mailpit (dev)	Public-facing reverse proxy communication
`wp_backend`	Yes	WordPress, MariaDB, Redis, Solr, Cron, Prometheus	WordPress backend services, isolated from internet
`lm_backend`	Yes	Listmonk, PostgreSQL	Listmonk backend, isolated from internet
`monitoring`	Yes	Prometheus, Grafana	Monitoring services, isolated from internet

Internal networks (`internal: true`) cannot reach the internet and cannot be reached from outside the Docker host. Only the `proxy` network allows external communication, which is limited to the Caddy reverse proxy.

5.2.4 Caddy Reverse Proxy

Caddy serves as the single entry point for all HTTP/HTTPS traffic. In the Docker context, it:

- **Terminates TLS** – In production, Caddy automatically provisions Let's Encrypt certificates for all configured domains. In development, it generates self-signed certificates for `localhost` .
- **Routes traffic** – Each service has its own domain/port configuration via environment variables
- **Adds security headers** – A reusable `(security_headers)` snippet is imported by all server blocks

- **Exposes Prometheus metrics** – The Caddy admin API on port 2019 serves `/metrics` for Prometheus scraping
- **Provides zero-downtime reloads** – Configuration changes are applied without dropping connections

Caddyfile Architecture

```
{
    admin 0.0.0.0:2019          # Prometheus metrics endpoint
    servers {
        metrics                  # Per-server request metrics
    }
}

(security_headers) {
    header {
        X-Content-Type-Options "nosniff"
        X-Frame-Options "SAMEORIGIN"
        Referrer-Policy "strict-origin-when-cross-origin"
        Permissions-Policy "camera=(), microphone=(), geolocation=(), payment=()"
        Strict-Transport-Security "max-age=31536000; includeSubDomains; preload"
        X-XSS-Protection "0"
        -Server
    }
}
```

The `(security_headers)` snippet is a Caddy named snippet – it's imported by each server block via `import security_headers` to ensure consistent header policy across all services.

Service Routing

Domain Variable	Target Service	Default (Dev)
`WP_SERVER_NAME`	`wordpress:80`	`localhost`
`LISTMONK_SERVER_NAME`	`listmonk:9000`	`localhost:9443`
`GRAFANA_SERVER_NAME`	`grafana:3000`	`localhost:6443`

Each domain is defined using Caddy's environment variable syntax `{$VAR:default}`:

```
{$WP_SERVER_NAME:localhost} {
    import security_headers
    reverse_proxy wordpress:80
}
```

In development, Caddy listens on localhost with self-signed TLS. In production, set real domain names and Caddy auto-provisions certificates via ACME/Let's Encrypt.

Development vs Production

Development (`compose.override.yaml`):

- The Caddyfile is bind-mounted for live editing
- Additional ports exposed for Listmonk (9443), Grafana (6443)

- Self-signed TLS for localhost

Production (`compose.prod.yaml`):

- Caddyfile is baked into the `wp-jobroom-proxy` Docker image
- Only ports 80 and 443 are exposed
- Let's Encrypt auto-TLS for real domain names
- Caddy data volume persists certificates

Adding Custom Caddy Configuration

Examples of common customizations:

Rate limiting (requires `caddy-ratelimit` plugin or custom build):

```
{$WP_SERVER_NAME:localhost} {
    import security_headers

    @login path /wp-login.php /login/
    rate_limit @login {
        zone login_zone {
            key {remote_host}
            events 10
            window 1m
        }
    }

    reverse_proxy wordpress:80
}
```

IP allowlisting for admin:

```
{$WP_SERVER_NAME:localhost} {
    import security_headers

    @admin path /wp-admin/*
    handle @admin {
        @blocked not remote_ip 10.0.0.0/8 192.168.0.0/16
        respond @blocked 403
    }

    reverse_proxy wordpress:80
}
```

Static file caching:

```
{$WP_SERVER_NAME:localhost} {
    import security_headers

    @static path *.css *.js *.png *.jpg *.gif *.svg *.woff2
    header @static Cache-Control "public, max-age=2592000"

    reverse_proxy wordpress:80
}
```

5.2.5 Docker Image Build Targets

The multi-stage `docker/Dockerfile` provides 6 build targets:

Target	Base Image	Purpose
`base`	`wordpress:php8.4`	Shared foundation with PHP extensions, WP-CLI, plugins
`development`	`base`	Adds Xdebug and Debug Bar plugin
`production`	`base`	Bakes in plugin, themes, and compiled translations
`proxy`	`caddy:latest`	Bakes in Caddyfile configuration (reverse proxy)
`prometheus`	`prom/prometheus:latest`	Bakes in Prometheus scrape configuration
`grafana`	`grafana/grafana:latest`	Bakes in Grafana provisioning and JobRoom dashboard

Base image includes:

- PHP extensions: redis, apcu, imagick, zip
- WP-CLI
- Plugins: wp-prometheus, smtp-mailer, redis-cache

Development image additionally includes:

- Xdebug (disabled by default, controlled via `XDEBUG_MODE`)
- Debug Bar plugin

5.3 Quick Start: Development

1. **Clone the repository:**

   ```
   git clone https://src.bundespruefstelle.ch/magdev/wp-jobroom.git
   cd wp-jobroom
   git submodule update --init --recursive
   composer install
   ```

2. **Create the environment file:**

   ```
   cp .env-dist .env
   ```

   Edit `.env` and change all `CHANGE_ME` values. At minimum:

   - `WP_DB_PASSWORD` and `WP_DB_ROOT_PASSWORD`
   - `LISTMONK_DB_PASSWORD`
   - All `WP_*_KEY` and `WP_*_SALT` values (generate at https://api.wordpress.org/secret-key/1.1/salt/)

3. **Start the stack:**

```
# Core services only (WordPress, database, Redis, Solr, Caddy, Mailpit)
docker compose --profile core up -d

# With newsletter support
docker compose --profile core --profile newsletter up -d

# With monitoring
docker compose --profile core --profile monitoring up -d

# Everything at once
docker compose --profile fullstack up -d
```

   This uses `compose.yaml` + `compose.override.yaml` (development defaults).

4. **Complete WordPress installation:**

   Open https://localhost in your browser and follow the WordPress setup wizard.

5. **Apply Solr schema** (after WordPress is installed):

   Navigate to **JobRoom > Settings > Integrations > Solr** and click **Apply Schema**, then **Reindex All**.

6. **Initialize Listmonk** (if using newsletter profile, first time only):

   Listmonk auto-initializes on startup via its `--install --idempotent` command. Access the admin UI at https://localhost:9443.

5.3.1 Development URLs

Service	URL
WordPress	https://localhost or http://localhost:9080
Listmonk	https://localhost:9443
Grafana	https://localhost:6443 or http://localhost:3000
Prometheus	http://localhost:9090 (internal only, not exposed via Caddy)
Mailpit UI	http://localhost:8025
Solr Admin	http://localhost:8983

5.3.2 Xdebug

Xdebug is installed in the development image but disabled by default. To enable step debugging:

```
# In .env
XDEBUG_MODE=debug
XDEBUG_PORT=9003
```

Then restart the WordPress container:

```
docker compose restart wordpress
```

Configure your IDE to listen on port 9003 (default). The `client_host` is set to `host.docker.internal`.

5.3.3 Development Overrides

The `compose.override.yaml` file provides development-specific configuration:

- **WordPress**: Builds `development` target (overrides base `production`), mounts source code and theme directories as bind-mounts for live editing
- **Caddy**: Mounts `docker/Caddyfile` as bind-mount for live configuration changes, exposes Listmonk and Grafana ports
- **Cron**: Container named `jobroom-cron`, `restart: unless-stopped` (base uses `restart: always`)
- **Prometheus**: Mounts `docker/monitoring/prometheus/prometheus.yml` as bind-mount for live config changes
- **Grafana**: Mounts provisioning configs and dashboard JSON as bind-mounts for live updates
- **Mailer**: Adds Mailpit service for email testing (profile: `core + newsletter`)
- **All services**: Container names prefixed with `jobroom-`, `restart: unless-stopped`

5.4 Quick Start: Production

1. **Create the environment file** with production values:

   ```
   cp .env-dist .env
   ```

   Required production values:

   - `WP_SERVER_NAME` - Your domain (e.g., `jobroom.example.com`)
   - `LISTMONK_SERVER_NAME` - Newsletter domain (e.g., `newsletter.example.com`)
   - `GRAFANA_SERVER_NAME` - Grafana domain (e.g., `grafana.example.com`)
   - `SMTP_HOST`, `SMTP_PORT`, `SMTP_FROM` - Real SMTP server
   - `WP_PROMETHEUS_AUTH_TOKEN` - Bearer token for Prometheus scraping
   - `GRAFANA_ADMIN_PASSWORD` - Grafana admin password
   - All database passwords and WordPress security keys

2. **Start with production configuration:**

   ```
   docker compose -f compose.yaml -f compose.prod.yaml --profile core --profile
   monitoring up -d
   ```

   Caddy automatically provisions Let's Encrypt TLS certificates for your domains.

3. **Initialize services** (first time only):

   ```
   docker compose -f compose.yaml -f compose.prod.yaml exec search solr create -c
   jobroom
   ```

   Then apply the Solr schema from the WordPress admin.

5.5 Monitoring

5.5.1 Prometheus

Prometheus scrapes metrics from three sources:

Job	Target	Metrics Path	Auth
`wordpress`	`wordpress:80`	`/metrics/`	Bearer token (`WP_PROMETHEUS_AUTH_TOKEN`)
`caddy`	`caddy:2019`	`/metrics`	None (internal network)
`docker`	`host.docker.internal:9323`	`/metrics`	None (host network)

The Prometheus configuration is at `docker/monitoring/prometheus/prometheus.yml`.

WordPress metrics require:

1. The WP Prometheus plugin to be activated
2. Prometheus metrics enabled in **JobRoom > Settings > Integrations > Prometheus**
3. A Bearer auth token configured in both `.env` (`WP_PROMETHEUS_AUTH_TOKEN`) and the WP Prometheus plugin settings

Caddy metrics are enabled via the global options in `docker/Caddyfile`:

```
{
    admin 0.0.0.0:2019
    servers {
        metrics
    }
}
```

The Caddy admin API (port 2019) is only accessible from within the `wp_backend` Docker network — not exposed to the host.

Docker daemon metrics require `host.docker.internal` to resolve to the host machine. The Prometheus service includes `extra_hosts: host.docker.internal=host-gateway` for this purpose. You also need to enable the Docker daemon's built-in Prometheus endpoint. Copy `docker/monitoring/prometheus/docker-daemon.json` to your Docker daemon config:

```
# Linux: /etc/docker/daemon.json
# macOS: ~/.docker/daemon.json
sudo cp docker/monitoring/prometheus/docker-daemon.json /etc/docker/daemon.json
sudo systemctl restart docker
```

> **Note**: Docker daemon metrics are optional. Prometheus will log a scrape error for the `docker` job if the daemon endpoint is not enabled, but this does not affect other scrape targets.

5.5.2 Grafana

Grafana is pre-configured with:

- **Prometheus datasource**: Auto-configured to connect to `http://prometheus:9090`
- **JobRoom dashboard**: 59 panels across 11 sections, auto-provisioned from `data/grafana-jobroom-dashboard.json`

Default credentials: `admin` / value of `GRAFANA_ADMIN_PASSWORD` (default: `admin`).

Grafana is accessible via:

- **Development**: http://localhost:3000 (direct) or https://localhost:6443 (via Caddy)
- **Production**: Via Caddy reverse proxy at the configured `GRAFANA_SERVER_NAME` domain

Prometheus is only accessible from internal Docker networks (`wp_backend` and `monitoring`). It is not exposed via the Caddy reverse proxy. In development, port 9090 is mapped to the host for debugging:

- **Development**: http://localhost:9090 (direct port mapping)
- **Production**: Not publicly accessible — only reachable by Grafana and WordPress within the Docker network

Dashboard Sections

Section	Panels	Metrics
Overview	6	Applicants, companies, jobs, applications, daily activity
Applications & Matching	5	Status breakdown, match scores, position types
Subscriptions	3	Tier distribution, status, MRR
Communication & Activity	4	Messages, threads, saved searches
Taxonomy & Configuration	5	Skills, occupations, industries, threshold
Search Engine (Solr)	9	Solr status, documents, completeness, pending sync
Users & Verification	4	User counts by role, email verification
Security & Compliance	8	MFA, cookie consent, newsletter, API keys
Revenue & Invoices	3	Invoice status, total revenue, pending
Job Offer Lifecycle	3	Expired, expiring soon, no deadline
Data Quality	2	Test data detection and breakdown

5.6 Environment Variables

5.6.1 Docker Images

Variable	Default	Description
`WP_IMAGE`	`hub.bundespruefstelle.ch/wp-jobroom`	Base Docker image name (suffixed with `-proxy`, `-prometheus`, `-grafana` for other images)
`WP_TAG`	`latest`	Docker image tag (used for all images)

5.6.2 WordPress

Variable	Default	Description
`PHP_VERSION`	`8.4`	PHP version for the Docker image
`WP_PORT`	`9080`	WordPress direct port (dev only)
`WP_DEBUG`	`1` (dev) / `0` (prod)	Enable WordPress debug mode
`WP_DEBUG_LOG`	`1` (dev) / `0` (prod)	Enable debug logging

5.6.3 Database

Variable	Default	Required	Description
`WP_DB_NAME`	`wordpress`		Database name
`WP_DB_USER`	`wordpress`		Database user
`WP_DB_PASSWORD`		Yes	Database password
`WP_DB_ROOT_PASSWORD`		Yes	MariaDB root password

5.6.4 Security Keys

All keys are **required**. Generate at https://api.wordpress.org/secret-key/1.1/salt/.

`WP_AUTH_KEY`, `WP_SECURE_AUTH_KEY`, `WP_LOGGED_IN_KEY`, `WP_NONCE_KEY`, `WP_AUTH_SALT`, `WP_SECURE_AUTH_SALT`, `WP_LOGGED_IN_SALT`, `WP_NONCE_SALT`

5.6.5 License

Variable	Default	Description
`WP_JOBROOM_LICENSE_SERVER_URL`		License server base URL
`WP_JOBROOM_LICENSE_SERVER_SECRET`		Shared HMAC secret for secure license server communication (optional)
`WP_JOBROOM_LICENSE_KEY`		License key to activate the plugin

All three variables are **optional**. When set, they take precedence over the corresponding values stored in the WordPress database (Settings > License tab). The admin UI fields become read-only for any setting configured via environment variable.

This is the recommended approach for production deployments because it:

- Keeps secrets out of the database (no plaintext license key in `wp_options`)
- Makes deployments reproducible (license config travels with the environment, not the database)
- Prevents accidental changes via the admin UI

Example `.env` configuration:

```
WP_JOBROOM_LICENSE_SERVER_URL=https://shop.example.com
WP_JOBROOM_LICENSE_SERVER_SECRET=your-hmac-shared-secret
WP_JOBROOM_LICENSE_KEY=XXXX-XXXX-XXXX-XXXX
```

When only some variables are set (e.g., `WP_JOBROOM_LICENSE_SERVER_URL` without the key), the remaining values are read from the database as usual.

5.6.6 Caddy

Variable	Default	Description
`CADDY_HTTP_PORT`	`80`	HTTP port
`CADDY_HTTPS_PORT`	`443`	HTTPS port
`CADDY_LISTMONK_PORT`	`9443`	Listmonk HTTPS port (dev only)
`CADDY_GRAFANA_PORT`	`6443`	Grafana HTTPS port (dev only)
`WP_SERVER_NAME`	`localhost`	WordPress domain
`LISTMONK_SERVER_NAME`	`localhost:9443`	Listmonk address (prod: use a domain name)
`GRAFANA_SERVER_NAME`	`localhost:6443`	Grafana address (prod: use a domain name)

5.6.7 SMTP

Variable	Default	Description
`SMTP_HOST`	`mailer`	SMTP server hostname
`SMTP_PORT`	`1025`	SMTP server port
`SMTP_FROM`	`wordpress@localhost`	Sender email address
`SMTP_FROM_NAME`	`WP JobRoom`	Sender display name

5.6.8 Listmonk

Variable	Default	Required	Description
`LISTMONK_DB_NAME`	`listmonk`		PostgreSQL database name
`LISTMONK_DB_USER`	`listmonk`		PostgreSQL user
`LISTMONK_DB_PASSWORD`		Yes	PostgreSQL password

5.6.9 Monitoring

Variable	Default	Description
`WP_PROMETHEUS_AUTH_TOKEN`		Bearer token for Prometheus to scrape WordPress metrics
`GRAFANA_ADMIN_PASSWORD`	`admin`	Grafana admin password
`GRAFANA_PORT`	`3000`	Grafana direct port (dev only)
`PROMETHEUS_PORT`	`9090`	Prometheus direct port (dev only)

5.6.10 Solr

Variable	Default	Description
`SOLR_PORT`	`8983`	Solr admin port (dev only)

5.6.11 Xdebug (Development Only)

Variable	Default	Description
`XDEBUG_MODE`	`off`	Xdebug mode (`off` , `debug` , `coverage` , `profile`)
`XDEBUG_PORT`	`9003`	Xdebug client port

5.6.12 Mailpit (Development Only)

Variable	Default	Description
`MAILPIT_UI_PORT`	`8025`	Mailpit web UI port

5.7 Common Tasks

5.7.1 WP-CLI Commands

```
# List installed plugins
docker compose exec wordpress wp plugin list --allow-root

# Activate WP JobRoom
docker compose exec wordpress wp plugin activate wp-jobroom --allow-root

# Activate wp-prometheus
docker compose exec wordpress wp plugin activate wp-prometheus --allow-root

# Flush rewrite rules
docker compose exec wordpress wp rewrite flush --allow-root

# Create admin user
docker compose exec wordpress wp user create admin admin@example.com --
role=administrator --user_pass=admin --allow-root

# Export database
docker compose exec db mariadb-dump -u wordpress -p wordpress > backup.sql
```

5.7.2 WP-Cron

The `cron` service triggers WordPress's scheduled task runner every 5 minutes, replacing unreliable visitor-triggered cron. This ensures time-sensitive tasks always run on schedule:

- Newsletter digest emails (daily / weekly)
- SLA breach detection and alerts
- IMAP email polling (every 5 minutes)
- Matching notification digests

```
# Check cron service logs
docker compose logs -f cron

# Trigger WP-Cron manually (for debugging)
docker compose exec wordpress wp cron event run --due-now --allow-root

# List scheduled WP-Cron events
docker compose exec wordpress wp cron event list --allow-root
```

5.7.3 Redis

```
# Check Redis connection
docker compose exec redis redis-cli ping

# Monitor Redis activity
docker compose exec redis redis-cli monitor

# Flush Redis cache
docker compose exec redis redis-cli flushall
```

5.7.4 Solr

```
# Create core (auto-created on container start via solr-precreate)
docker compose exec search solr create -c jobroom

# Delete core
docker compose exec search solr delete -c jobroom

# Check core status
docker compose exec search curl -s http://localhost:8983/solr/admin/cores?action=
STATUS
```

5.7.5 Listmonk

```
# Initialize database (first time — auto-runs on container start)
docker compose exec listmonk ./listmonk --install

# Upgrade database (after version update)
docker compose exec listmonk ./listmonk --upgrade
```

5.7.6 Monitoring

```
# Check Prometheus targets
curl http://localhost:9090/api/v1/targets

# Check Prometheus is scraping WordPress
curl http://localhost:9090/api/v1/query?query=jobroom_applicants_total

# Check Caddy metrics directly
curl http://localhost:2019/metrics

# Grafana API health
curl http://localhost:3000/api/health
```

5.7.7 Container Management

```
# View all service logs
docker compose logs -f
```

```
# View specific service logs
docker compose logs -f wordpress

# Restart a service
docker compose restart wordpress

# Stop the stack
docker compose down

# Stop and remove volumes (WARNING: deletes all data)
docker compose down -v

# Rebuild all images (after Dockerfile changes)
docker compose build

# Rebuild specific image
docker compose build wordpress

# Check service health
docker compose ps
```

5.8 Customization

5.8.1 Custom PHP Configuration

Create `docker/php-custom.ini` and mount it in `compose.override.yaml`:

```
services:
  wordpress:
    volumes:
      - ./docker/php-custom.ini:/usr/local/etc/php/conf.d/custom.ini:ro
```

5.8.2 Adding WordPress Plugins

Add plugin installation commands to `docker/Dockerfile` in the `base` stage:

```
RUN curl -fsSL -o /tmp/plugin.zip \
        "https://downloads.wordpress.org/plugin/plugin-name.latest-stable.zip" \
    && unzip -q /tmp/plugin.zip -d /usr/src/wordpress/wp-content/plugins/ \
    && rm /tmp/plugin.zip
```

5.8.3 Custom Caddy Configuration

Edit `docker/Caddyfile` for additional routing, rate limiting, or security headers. The Caddyfile includes a reusable `security_headers` snippet:

```
(security_headers) {
    header {
        X-Content-Type-Options nosniff
        X-Frame-Options SAMEORIGIN
        Referrer-Policy strict-origin-when-cross-origin
        X-XSS-Protection "1; mode=block"
        -Server
    }
}
```

5.8.4 Custom Prometheus Scrape Targets

Edit `docker/monitoring/prometheus/prometheus.yml` to add additional scrape targets. In development, the config file is bind-mounted so changes take effect after restarting Prometheus:

```
docker compose restart prometheus
```

5.8.5 Custom Grafana Dashboards

Place additional dashboard JSON files in `docker/monitoring/grafana/provisioning/dashboards/`. In development, the provisioning directory is bind-mounted.

5.9 CI/CD: Docker Image Build

Four Docker images are automatically built and pushed to the container registry when a version tag is pushed:

```
git tag -a v0.18.10 -m "Version 0.18.10"
git push origin v0.18.10
```

The workflow (`.gitea/workflows/docker.yml`) runs:

1. **PHP Lint** - Syntax checking
2. **Unit Tests** - PHPUnit test suite
3. **Code Quality** - Composer validation and security audit
4. **Build and Push** - Build and push all 4 Docker images

5.9.1 Built Images

WordPress (`wp-jobroom`) – WordPress with PHP extensions, plugin, and themes baked in:

- `hub.bundespruefstelle.ch/wp-jobroom:latest`
- `hub.bundespruefstelle.ch/wp-jobroom:0` (major)
- `hub.bundespruefstelle.ch/wp-jobroom:0.18` (minor)
- `hub.bundespruefstelle.ch/wp-jobroom:0.18.10` (exact)

Proxy (`wp-jobroom-proxy`) – Caddy reverse proxy with Caddyfile baked in:

- `hub.bundespruefstelle.ch/wp-jobroom-proxy:{latest,major,minor,exact}`

Prometheus (`wp-jobroom-prometheus`) – Prometheus with scrape config baked in:

- `hub.bundespruefstelle.ch/wp-jobroom-prometheus:{latest,major,minor,exact}`

Grafana (`wp-jobroom-grafana`) – Grafana with provisioning and dashboard baked in:

- `hub.bundespruefstelle.ch/wp-jobroom-grafana:{latest,major,minor,exact}`

5.9.2 Dependency Resolution

The CI/CD workflow automatically resolves the latest versions of dependency projects:

- **wp-prometheus** - WordPress Prometheus plugin
- **wp-bootstrap** - Parent theme
- **wp-jobroom-theme** - Child theme

Versions are resolved via the Gitea API with a three-tier fallback: `/releases/latest` → `/releases?limit=1` (includes pre-releases) → `/tags?limit=1` .

5.9.3 Required Gitea Secrets

Secret	Description
`REGISTRY_USERNAME`	Container registry username
`REGISTRY_PASSWORD`	Container registry password
`SRC_GITEA_TOKEN`	Gitea API token for cross-repo dependency resolution

5.10 Compose File Structure

File	Purpose
`compose.yaml`	Base configuration with all services, build targets, profiles, networks, and volumes
`compose.override.yaml`	Development overrides: bind-mounts, exposed ports, container names, Mailpit, WordPress dev target
`compose.prod.yaml`	Production overrides: pre-built images, SMTP config, required env vars

How they combine:

- **Development**: `compose.yaml` + `compose.override.yaml` (automatic)
- **Production**: `compose.yaml` + `compose.prod.yaml` (explicit `-f` flags)

5.11 Troubleshooting

5.11.1 WordPress shows “Error establishing a database connection”

The MariaDB container may not be ready yet. Check its health:

```
docker compose ps db
docker compose logs db
```

5.11.2 Caddy returns 502 Bad Gateway

WordPress container may still be starting. Check its health:

```
docker compose ps wordpress
docker compose logs wordpress
```

5.11.3 Solr connection fails

Verify the Solr container is running and the core exists:

```
docker compose ps search
docker compose exec search solr status
```

The `jobroom` core is auto-created on container start via `solr-precreate`. If missing:

```
docker compose exec search solr create -c jobroom
```

5.11.4 Listmonk shows database errors

Listmonk auto-initializes on startup. If you see errors, check the PostgreSQL container:

```
docker compose ps lmdb
docker compose logs listmonk
```

5.11.5 Emails not arriving (development)

Check Mailpit UI at http://localhost:8025. All emails sent by WordPress and Listmonk are captured there.

5.11.6 Prometheus shows "DOWN" targets

Check which targets are failing:

```
curl -s http://localhost:9090/api/v1/targets | python3 -m json.tool
```

Common causes:

- **wordpress target DOWN**: WP Prometheus plugin not activated, or `WP_PROMETHEUS_AUTH_TOKEN` mismatch between `.env` and plugin settings
- **caddy target DOWN**: Caddy admin API not accessible on port 2019 (check Caddyfile global options)
- **docker target DOWN**: Docker daemon metrics not enabled (optional, see Docker daemon metrics section)

5.11.7 Grafana shows "No Data"

1. Verify Prometheus is healthy: http://localhost:9090/-/healthy

2. Check Prometheus targets: http://localhost:9090/targets
3. Verify the Grafana datasource: Grafana > Connections > Data sources > Prometheus > Test

5.11.8 Permission issues with plugin bind mount

If WordPress can't write to the plugin directory in development:

```
docker compose exec wordpress chown -R www-data:www-data /var/www/html/wp-content/
plugins/wp-jobroom
```

---

6 Web Server Configuration Guide

6.1 Introduction

This guide covers web server configuration for standalone (non-Docker) installations of WP JobRoom. It addresses the specific requirements of the plugin's custom URL routing, file upload handling, security headers, and optional Solr search integration.

For Docker-based deployments, see the Docker Guide. The Docker stack uses Caddy as a reverse proxy with automatic TLS and is fully pre-configured.

The WP JobRoom plugin registers custom rewrite rules via WordPress's `add_rewrite_rule()` API using the `jr_action` query variable. These rules power all plugin frontend URLs (`/login/` , `/dashboard/` , `/pricing/` , `/register/` , `/search/` , `/messages/` , `/support/` , `/account/` , etc.). Your web server must support URL rewriting so that WordPress can process these requests through `index.php` .

6.2 General Requirements

6.2.1 Software

Requirement	Minimum	Recommended
PHP	8.3.0	8.4.x
WordPress	6.0	Latest
Web Server	Apache 2.4, Nginx 1.18, or Caddy 2.x	Latest stable

6.2.2 PHP Extensions

The following PHP extensions are required or recommended:

Extension	Required	Purpose
`curl`	Yes	REST API calls, Listmonk integration, Payrexx webhooks
`mbstring`	Yes	Multi-byte string handling (i18n, UTF-8)
`xml`	Yes	WordPress core dependency
`zip`	Yes	CV/PDF import, profile export
`gd`	Yes	Image processing (fallback)
`imagick`	Recommended	Avatar image processing (300x300 crop), better quality than GD
`redis`	Recommended	Object cache (Redis-based caching)
`apcu`	Recommended	In-memory opcode/data caching
`intl`	Recommended	Locale-aware formatting

6.2.3 Network

- **HTTPS is strongly recommended.** Required for:
  - Push Notifications (VAPID/Web Push requires secure context)
  - Secure cookie attributes (`Secure` , `SameSite`)
  - HSTS preloading
  - Service Worker registration (requires secure origin)
- **URL rewriting** must be enabled (`mod_rewrite` for Apache, `try_files` for Nginx, built-in for Caddy)
- **Sufficient upload limits** for CV/certificate uploads: at least 10 MB for `upload_max_filesize` and `post_max_size`

6.3 Apache Configuration

6.3.1 .htaccess (WordPress Root)

Place this `.htaccess` in your WordPress root directory. It extends the standard WordPress rewrite rules with security headers and access restrictions for plugin-sensitive files.

```
# ============================================================================
# WordPress Rewrite Rules
# ============================================================================
# BEGIN WordPress
<IfModule mod_rewrite.c>
RewriteEngine On
```

```
RewriteRule .* - [E=HTTP_AUTHORIZATION:%{HTTP:Authorization}]
RewriteBase /
RewriteRule ^index\.php$ - [L]
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule . /index.php [L]
</IfModule>
# END WordPress

# ======================================================================
# Security Headers
# ======================================================================
<IfModule mod_headers.c>
    Header always set X-Content-Type-Options "nosniff"
    Header always set X-Frame-Options "SAMEORIGIN"
    Header always set Referrer-Policy "strict-origin-when-cross-origin"
    Header always set Permissions-Policy "camera=(), microphone=(), geolocation=(),
payment=()"
    Header always set X-XSS-Protection "0"

    # HSTS - only enable after confirming HTTPS works correctly
    # Header always set Strict-Transport-Security "max-age=31536000;
includeSubDomains; preload"

    # Remove server version disclosure
    Header always unset X-Powered-By
</IfModule>

# ======================================================================
# Deny Access to Sensitive Plugin Files
# ======================================================================
<IfModule mod_rewrite.c>
    # Block access to translation source files
    RewriteRule ^wp-content/plugins/wp-jobroom/languages/.*\.(po|pot)$ - [F,L]

    # Block access to documentation and configuration
    RewriteRule ^wp-content/plugins/wp-jobroom/(CLAUDE|PLAN|CHANGELOG|
MARKETING).*\.md$ - [F,L]
    RewriteRule ^wp-content/plugins/wp-jobroom/composer\.(json|lock)$ - [F,L]
    RewriteRule ^wp-content/plugins/wp-jobroom/\.claude/ - [F,L]
    RewriteRule ^wp-content/plugins/wp-jobroom/\.gitea/ - [F,L]

    # Block access to plugin cache directory
    RewriteRule ^wp-content/plugins/wp-jobroom/cache/ - [F,L]

    # Block access to plugin tests
    RewriteRule ^wp-content/plugins/wp-jobroom/tests/ - [F,L]

    # Block access to docs directory
    RewriteRule ^wp-content/plugins/wp-jobroom/docs/ - [F,L]
</IfModule>

# ======================================================================
# Disable Directory Browsing
# ======================================================================
```

```
Options -Indexes

# ======================================================================
# PHP Settings (if using mod_php or CGI)
# ======================================================================
<IfModule mod_php.c>
    php_value upload_max_filesize 10M
    php_value post_max_size 12M
    php_value max_execution_time 120
    php_value memory_limit 256M
</IfModule>

# ======================================================================
# Compression
# ======================================================================
<IfModule mod_deflate.c>
    AddOutputFilterByType DEFLATE text/html text/plain text/css
    AddOutputFilterByType DEFLATE text/javascript application/javascript
application/json
    AddOutputFilterByType DEFLATE application/xml text/xml
    AddOutputFilterByType DEFLATE image/svg+xml
</IfModule>

# ======================================================================
# Browser Caching
# ======================================================================
<IfModule mod_expires.c>
    ExpiresActive On
    ExpiresByType text/css "access plus 1 month"
    ExpiresByType text/javascript "access plus 1 month"
    ExpiresByType application/javascript "access plus 1 month"
    ExpiresByType image/jpeg "access plus 1 year"
    ExpiresByType image/png "access plus 1 year"
    ExpiresByType image/gif "access plus 1 year"
    ExpiresByType image/webp "access plus 1 year"
    ExpiresByType image/svg+xml "access plus 1 year"
    ExpiresByType image/x-icon "access plus 1 year"
    ExpiresByType application/font-woff2 "access plus 1 year"
    ExpiresByType font/woff2 "access plus 1 year"
</IfModule>
```

6.3.2 VirtualHost Configuration

A complete VirtualHost configuration with SSL via Let's Encrypt (certbot).

```
<VirtualHost *:80>
    ServerName jobroom.example.com
    ServerAlias www.jobroom.example.com

    # Redirect all HTTP to HTTPS
    RewriteEngine On
    RewriteRule ^(.*)$ https://%{HTTP_HOST}$1 [R=301,L]
</VirtualHost>
```

```
<VirtualHost *:443>
    ServerName jobroom.example.com
    ServerAlias www.jobroom.example.com

    DocumentRoot /var/www/html

    # SSL via Let's Encrypt (certbot)
    SSLEngine on
    SSLCertificateFile /etc/letsencrypt/live/jobroom.example.com/fullchain.pem
    SSLCertificateKeyFile /etc/letsencrypt/live/jobroom.example.com/privkey.pem
    Include /etc/letsencrypt/options-ssl-apache.conf

    # Minimum TLS 1.2
    SSLProtocol all -SSLv3 -TLSv1 -TLSv1.1

    <Directory /var/www/html>
        AllowOverride All
        Require all granted
    </Directory>

    # Deny direct access to wp-config.php
    <Files wp-config.php>
        Require all denied
    </Files>

    # Solr reverse proxy (optional, if Solr is on a separate host)
    # Restrict to localhost or authenticated requests only
    # <Location /solr/>
    #     ProxyPass http://127.0.0.1:8983/solr/
    #     ProxyPassReverse http://127.0.0.1:8983/solr/
    #     Require ip 127.0.0.1
    # </Location>

    # Logging
    ErrorLog ${APACHE_LOG_DIR}/jobroom-error.log
    CustomLog ${APACHE_LOG_DIR}/jobroom-access.log combined
</VirtualHost>
```

Enable required Apache modules:

```
sudo a2enmod rewrite headers ssl deflate expires proxy proxy_http
sudo systemctl restart apache2
```

Obtain a Let's Encrypt certificate:

```
sudo apt install certbot python3-certbot-apache
sudo certbot --apache -d jobroom.example.com -d www.jobroom.example.com
```

6.3.3 Protecting Uploaded Files

Place this `.htaccess` in `wp-content/uploads/` to prevent PHP execution in the uploads directory:

```
# Deny PHP execution in uploads directory
<FilesMatch "\.ph(p[3457]?|t|tml)$">
    Require all denied
</FilesMatch>

# Deny access to hidden files
<FilesMatch "^\.">
    Require all denied
</FilesMatch>
```

6.4 Nginx Configuration

6.4.1 Server Block Configuration

A complete Nginx server block with SSL, WordPress permalink rewriting, security headers, and PHP-FPM integration.

```
# Upstream PHP-FPM
upstream php-fpm {
    server unix:/run/php/php8.3-fpm.sock;
    # Or for TCP: server 127.0.0.1:9000;
}

# Redirect HTTP to HTTPS
server {
    listen 80;
    listen [::]:80;
    server_name jobroom.example.com www.jobroom.example.com;
    return 301 https://$host$request_uri;
}

# Main server block
server {
    listen 443 ssl http2;
    listen [::]:443 ssl http2;
    server_name jobroom.example.com www.jobroom.example.com;

    root /var/www/html;
    index index.php;

    # ============================================================================
    # SSL Configuration (Let's Encrypt via certbot)
    # ============================================================================
    ssl_certificate /etc/letsencrypt/live/jobroom.example.com/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/jobroom.example.com/privkey.pem;
    include /etc/letsencrypt/options-ssl-nginx.conf;
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem;

    # Minimum TLS 1.2
    ssl_protocols TLSv1.2 TLSv1.3;
```

```
    # ============================================================================
    # Security Headers
    # ============================================================================
    add_header X-Content-Type-Options "nosniff" always;
    add_header X-Frame-Options "SAMEORIGIN" always;
    add_header Referrer-Policy "strict-origin-when-cross-origin" always;
    add_header Permissions-Policy "camera=(), microphone=(), geolocation=(),
payment=()" always;
    add_header X-XSS-Protection "0" always;
    # HSTS - only enable after confirming HTTPS works correctly
    # add_header Strict-Transport-Security "max-age=31536000; includeSubDomains;
preload" always;

    # Hide server version
    server_tokens off;

    # ============================================================================
    # Upload Size
    # ============================================================================
    client_max_body_size 12m;

    # ============================================================================
    # WordPress Permalink Rewriting
    # ============================================================================
    location / {
        try_files $uri $uri/ /index.php?$args;
    }

    # ============================================================================
    # PHP-FPM
    # ============================================================================
    location ~ \.php$ {
        fastcgi_pass php-fpm;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;

        fastcgi_intercept_errors on;
        fastcgi_buffer_size 128k;
        fastcgi_buffers 4 256k;
        fastcgi_busy_buffers_size 256k;
        fastcgi_read_timeout 120s;
    }

    # ============================================================================
    # Deny Access to Sensitive Plugin Files
    # ============================================================================
    # Translation source files
    location ~* /wp-content/plugins/wp-jobroom/languages/.*\.(po|pot)$ {
        deny all;
    }

    # Documentation and configuration files
    location ~* /wp-content/plugins/wp-jobroom/(CLAUDE|PLAN|CHANGELOG|
```

```
MARKETING).*\.md$ {
        deny all;
    }
    location ~* /wp-content/plugins/wp-jobroom/composer\.(json|lock)$ {
        deny all;
    }

    # Development and cache directories
    location ~ /wp-content/plugins/wp-jobroom/(\.claude|\.gitea|cache|tests|docs|
docker)/ {
        deny all;
    }

    # Deny access to wp-config.php
    location = /wp-config.php {
        deny all;
    }

    # Deny access to hidden files and directories
    location ~ /\. {
        deny all;
        access_log off;
        log_not_found off;
    }

    # ============================================================================
    # Deny PHP Execution in Uploads
    # ============================================================================
    location ~ ^/wp-content/uploads/.*\.php$ {
        deny all;
    }

    # ============================================================================
    # Static Asset Caching
    # ============================================================================
    location ~* \.(css|js)$ {
        expires 30d;
        add_header Cache-Control "public, immutable";
        access_log off;
    }

    location ~* \.(jpg|jpeg|png|gif|webp|ico|svg|woff|woff2|ttf|eot)$ {
        expires 1y;
        add_header Cache-Control "public, immutable";
        access_log off;
    }

    # ============================================================================
    # Gzip Compression
    # ============================================================================
    gzip on;
    gzip_vary on;
    gzip_proxied any;
    gzip_comp_level 6;
    gzip_min_length 1024;
```

```
    gzip_types
        text/plain
        text/css
        text/javascript
        application/javascript
        application/json
        application/xml
        text/xml
        image/svg+xml;

    # ============================================================================
    # Rate Limiting for Login and AJAX
    # ============================================================================
    # Define rate limit zones in the http {} block of nginx.conf:
    #   limit_req_zone $binary_remote_addr zone=login:10m rate=5r/m;
    #   limit_req_zone $binary_remote_addr zone=ajax:10m rate=30r/m;

    location = /wp-login.php {
        limit_req zone=login burst=3 nodelay;

        fastcgi_pass php-fpm;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
    }

    location = /wp-admin/admin-ajax.php {
        limit_req zone=ajax burst=20 nodelay;

        fastcgi_pass php-fpm;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
    }

    # ============================================================================
    # Logging
    # ============================================================================
    access_log /var/log/nginx/jobroom-access.log;
    error_log /var/log/nginx/jobroom-error.log;
}
```

Add the rate limit zones to your main `nginx.conf` inside the `http {}` block:

```
http {
    # Rate limiting zones for WP JobRoom
    limit_req_zone $binary_remote_addr zone=login:10m rate=5r/m;
    limit_req_zone $binary_remote_addr zone=ajax:10m rate=30r/m;

    # ... other configuration ...
}
```

Obtain a Let's Encrypt certificate:

```
sudo apt install certbot python3-certbot-nginx
sudo certbot --nginx -d jobroom.example.com -d www.jobroom.example.com
```

6.4.2 Solr Reverse Proxy (Optional)

If Apache Solr is running on a separate host and you need to access it through Nginx, add this location block. Restrict access to trusted IP addresses only.

```
# Solr reverse proxy - restrict access
location /solr/ {
    # Allow only from the WordPress server itself
    allow 127.0.0.1;
    allow ::1;
    deny all;

    proxy_pass http://127.0.0.1:8983/solr/;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;

    proxy_read_timeout 60s;
    proxy_connect_timeout 10s;
}
```

6.5 Caddy Configuration (Standalone)

Caddy provides automatic HTTPS via Let's Encrypt with zero configuration. This makes it an excellent choice for standalone WordPress deployments.

6.5.1 Caddyfile

```
# ================================================================================
# Security Headers Snippet (reusable)
# ================================================================================
(security_headers) {
    header {
        X-Content-Type-Options "nosniff"
        X-Frame-Options "SAMEORIGIN"
        Referrer-Policy "strict-origin-when-cross-origin"
        Permissions-Policy "camera=(), microphone=(), geolocation=(), payment=()"
        Strict-Transport-Security "max-age=31536000; includeSubDomains; preload"
        X-XSS-Protection "0"
        -Server
        -X-Powered-By
    }
}

# ================================================================================
```

```
# WordPress Site
# ================================================================================
jobroom.example.com {
    import security_headers

    root * /var/www/html

    # PHP-FPM
    php_fastcgi unix//run/php/php8.3-fpm.sock {
        # Or for TCP:
        # php_fastcgi 127.0.0.1:9000
    }

    # Static file serving
    file_server

    # Upload size limit
    request_body {
        max_size 12MB
    }

    # ================================================================
    # Deny Access to Sensitive Files
    # ================================================================

    # Block translation source files
    @po_files path /wp-content/plugins/wp-jobroom/languages/*.po /wp-content/
plugins/wp-jobroom/languages/*.pot
    respond @po_files 403

    # Block documentation and configuration
    @sensitive_md path_regexp sensitive /wp-content/plugins/wp-jobroom/(CLAUDE|PLAN|
CHANGELOG|MARKETING).*\.md$
    respond @sensitive_md 403

    @composer_files path /wp-content/plugins/wp-jobroom/composer.json /wp-content/
plugins/wp-jobroom/composer.lock
    respond @composer_files 403

    # Block development and cache directories
    @dev_dirs path /wp-content/plugins/wp-jobroom/.claude/* /wp-content/plugins/wp-
jobroom/.gitea/* /wp-content/plugins/wp-jobroom/cache/* /wp-content/plugins/wp-
jobroom/tests/* /wp-content/plugins/wp-jobroom/docs/* /wp-content/plugins/wp-
jobroom/docker/*
    respond @dev_dirs 403

    # Block wp-config.php
    @wpconfig path /wp-config.php
    respond @wpconfig 403

    # Block hidden files
    @hidden path /.*
    respond @hidden 403

    # Block PHP execution in uploads
```

```
    @uploads_php path_regexp uploads_php ^/wp-content/uploads/.*\.php$
    respond @uploads_php 403

    # ============================================================================
    # Static Asset Caching
    # ============================================================================
    @static path *.css *.js *.jpg *.jpeg *.png *.gif *.webp *.ico *.svg *.woff
*.woff2 *.ttf *.eot
    header @static Cache-Control "public, max-age=2592000, immutable"

    @images path *.jpg *.jpeg *.png *.gif *.webp *.ico *.svg
    header @images Cache-Control "public, max-age=31536000, immutable"

    # ============================================================================
    # Compression (enabled by default in Caddy)
    # ============================================================================
    encode gzip zstd

    # ============================================================================
    # Logging
    # ============================================================================
    log {
        output file /var/log/caddy/jobroom-access.log
        format json
    }
}
```

Caddy handles TLS certificate provisioning automatically. No manual certbot setup is needed. Simply point your domain's DNS to the server and Caddy obtains and renews certificates from Let's Encrypt.

6.5.2 Solr Reverse Proxy (Optional)

Add this block inside the site definition if Solr needs to be proxied:

```
    # Solr reverse proxy - bind to localhost only
    @solr path /solr/*
    handle @solr {
        # Restrict to local connections
        @not_local not remote_ip 127.0.0.1 ::1
        respond @not_local 403

        reverse_proxy 127.0.0.1:8983
    }
```

6.6 Security Headers Reference

All recommended security headers and their purpose:

Header	Value	Purpose
`X-Content-Type-Options`	`nosniff`	Prevents MIME type sniffing. Browsers must use the declared `Content-Type`.
`X-Frame-Options`	`SAMEORIGIN`	Prevents clickjacking by restricting `<iframe>` embedding to same-origin pages.
`Referrer-Policy`	`strict-origin-when-cross-origin`	Sends full URL as referrer for same-origin requests, only the origin for cross-origin.
`Permissions-Policy`	`camera=(), microphone=(), geolocation=(), payment=()`	Disables browser features not used by WP JobRoom.
`Strict-Transport-Security`	`max-age=31536000; includeSubDomains; preload`	Forces HTTPS for 1 year. Only enable after confirming HTTPS works correctly.
`X-XSS-Protection`	`0`	Explicitly disabled. Modern browsers use Content Security Policy instead. The legacy XSS filter can introduce vulnerabilities.

6.6.1 Content Security Policy Considerations

A full Content Security Policy (CSP) is not included in the configurations above because WordPress admin relies heavily on inline scripts and styles, and many plugins inject inline JavaScript. A strict CSP will break the WordPress admin interface.

If you want to implement CSP, start with report-only mode to identify violations before enforcing:

```
Content-Security-Policy-Report-Only: default-src 'self'; script-src 'self' 'unsafe-inline' 'unsafe-eval'; style-src 'self' 'unsafe-inline'; img-src 'self' data: https:; font-src 'self' data:; connect-src 'self'; frame-ancestors 'self'
```

Key considerations for WP JobRoom:

- `'unsafe-inline'` is required for both `script-src` and `style-src` due to WordPress core and plugin inline scripts
- `'unsafe-eval'` may be required for certain WordPress admin features (Gutenberg editor)
- `connect-src` must allow your Solr host if using external Solr
- `img-src` must allow `data:` for inline SVGs and potentially `https://secure.gravatar.com` for Gravatar support

6.7 PHP Configuration Recommendations

6.7.1 php.ini Settings

These settings can be placed in `php.ini`, a drop-in `.ini` file (e.g., `/etc/php/8.3/fpm/conf.d/99-jobroom.ini`), or in `.htaccess` for Apache with `mod_php`.

```
; ============================================================================
; Upload and Request Limits
; ============================================================================
upload_max_filesize = 10M
post_max_size = 12M
max_execution_time = 120
max_input_time = 60
max_input_vars = 3000

; ============================================================================
; Memory
; ============================================================================
memory_limit = 256M

; ============================================================================
; OPcache (recommended for production)
; ============================================================================
opcache.enable = 1
opcache.memory_consumption = 128
opcache.interned_strings_buffer = 16
opcache.max_accelerated_files = 10000
opcache.revalidate_freq = 60
opcache.validate_timestamps = 1
opcache.save_comments = 1

; ============================================================================
; Session
; ============================================================================
session.cookie_httponly = 1
session.cookie_secure = 1
session.cookie_samesite = Lax
session.use_strict_mode = 1

; ============================================================================
; Error Handling (production)
; ============================================================================
display_errors = Off
log_errors = On
error_log = /var/log/php/error.log
error_reporting = E_ALL & ~E_DEPRECATED & ~E_STRICT

; ============================================================================
; Redis Session Handler (optional, if using Redis)
; ============================================================================
; session.save_handler = redis
; session.save_path = "tcp://127.0.0.1:6379"
```

```
; ============================================================================
; APCu (if installed)
; ============================================================================
apc.enabled = 1
apc.shm_size = 64M
apc.ttl = 7200
apc.enable_cli = 0
```

6.7.2 PHP-FPM Pool Configuration

For Nginx and Caddy deployments using PHP-FPM, configure a dedicated pool in `/etc/php/8.3/fpm/pool.d/www.conf`:

```
[www]
user = www-data
group = www-data

; Use a Unix socket for better performance
listen = /run/php/php8.3-fpm.sock
listen.owner = www-data
listen.group = www-data
listen.mode = 0660

; Process management
pm = dynamic
pm.max_children = 20
pm.start_servers = 5
pm.min_spare_servers = 3
pm.max_spare_servers = 10
pm.max_requests = 500

; Timeouts
request_terminate_timeout = 120s

; Logging
slowlog = /var/log/php/fpm-slow.log
request_slowlog_timeout = 10s

; Environment variables (pass WordPress security keys if needed)
clear_env = no
```

6.7.3 Redis Object Cache

If using Redis for WordPress object caching (recommended for production):

1. Install and start Redis:

   ```
   sudo apt install redis-server
   sudo systemctl enable redis-server
   sudo systemctl start redis-server
   ```

2. Install the PHP Redis extension:

```
sudo apt install php8.3-redis
sudo systemctl restart php8.3-fpm
```

3. Add to `wp-config.php`:

```
define('WP_REDIS_HOST', '127.0.0.1');
define('WP_REDIS_PORT', 6379);
define('WP_CACHE', true);
```

4. Install and activate a Redis object cache plugin (e.g., Redis Object Cache by Till Kruss) through the WordPress admin.

6.8 SSL/TLS Recommendations

6.8.1 Certificate Providers

Web Server	Recommended Method	Notes
Apache	certbot with `python3-certbot-apache`	Automatic VirtualHost configuration
Nginx	certbot with `python3-certbot-nginx`	Automatic server block configuration
Caddy	Built-in (automatic)	Zero-configuration ACME; Let's Encrypt by default

6.8.2 TLS Protocol Requirements

- **Minimum:** TLS 1.2
- **Recommended:** TLS 1.3 (where supported by client and server)
- **Disabled:** SSLv3, TLS 1.0, TLS 1.1 (all have known vulnerabilities)

6.8.3 HSTS Preloading

HTTP Strict Transport Security (HSTS) tells browsers to only connect via HTTPS. The preload directive submits your domain to browser HSTS preload lists.

Before enabling HSTS preload:

1. Confirm HTTPS works correctly on all subdomains
2. Confirm all HTTP URLs redirect to HTTPS
3. Start with a short `max-age` (e.g., 300 seconds) and increase gradually
4. Only add `preload` after verifying everything works with a long `max-age`

To submit your domain for preload after enabling the header with `preload`, visit https://hstspreload.org/.

6.8.4 Let's Encrypt Setup (Apache and Nginx)

```
# Install certbot
sudo apt install certbot

# Apache
sudo apt install python3-certbot-apache
sudo certbot --apache -d jobroom.example.com

# Nginx
sudo apt install python3-certbot-nginx
sudo certbot --nginx -d jobroom.example.com

# Verify auto-renewal
sudo certbot renew --dry-run
```

Certbot automatically sets up a cron job or systemd timer for certificate renewal (every 60 days for Let's Encrypt certificates, which are valid for 90 days).

6.8.5 WP-Cron via System Cron

For reliable scheduled task execution (newsletter digests, SLA checks, IMAP polling), disable WordPress's visitor-triggered cron and use a system cron job instead:

1. Add to `wp-config.php`:

   ```
   define('DISABLE_WP_CRON', true);
   ```

2. Add a system cron entry:

   ```
   # Run WP-Cron every 5 minutes
   */5 * * * * curl -fsS -o /dev/null https://jobroom.example.com/wp-cron.php?
   doing_wp_cron
   ```

   Or using WP-CLI:

   ```
   */5 * * * * cd /var/www/html && /usr/local/bin/wp cron event run --due-now --
   quiet 2>/dev/null
   ```

---

7 Testing Guide

7.1 Unit Testing

7.1.1 Overview

Unit tests verify individual classes and methods in isolation using PHPUnit 11.

7.1.2 Running Unit Tests

```
# Install dev dependencies
composer install

# Run all unit tests
composer test:unit

# Run with coverage (requires Xdebug)
XDEBUG_MODE=coverage composer test:coverage

# Run specific test file
./vendor/bin/phpunit tests/Unit/Swiss/PhoneValidatorTest.php

# Run specific test method
./vendor/bin/phpunit --filter test_validates_valid_swiss_phone_numbers
```

7.1.3 Test Structure

```
tests/
├── bootstrap.php              # Test setup and WordPress mocks
├── Unit/
│   ├── Admin/
│   │   └── HelpPageImporterTest.php    # 56 tests - help page import
│   ├── Analytics/
│   │   └── EventTypeTest.php           # 9 tests - event type constants
│   ├── Communication/
│   │   └── ApplicationStatusTest.php   # 30 tests - application workflow
│   ├── Core/
│   │   ├── AssetOptimizerTest.php      # 42 tests - minification, lazy loading
│   │   ├── LocationFormatterTest.php   # 6 tests - location display
│   │   ├── SearchCacheSettingsTest.php # 9 tests - cache settings
│   │   ├── SingletonTraitTest.php      # 4 tests - singleton pattern
│   │   └── StatusWorkflowTraitTest.php # 14 tests - shared workflow trait
│   ├── Debug/
```

```
│   │   └── DebugCollectorTest.php      # 22 tests - debug data collector
│   ├── Marketing/
│   │   └── SocialShareTest.php         # 27 tests - social share URLs
│   ├── Matching/
│   │   ├── CalculatorTest.php          # 28 tests - scoring algorithms
│   │   └── CriteriaTest.php           # 25 tests - criteria configuration
│   ├── Security/
│   │   ├── EncryptionTest.php          # 18 tests - AES-256-GCM encryption
│   │   ├── PasswordPolicyTest.php      # 28 tests - password complexity
│   │   └── TOTPManagerTest.php         # 26 tests - MFA/TOTP (RFC 6238)
│   ├── Solr/
│   │   ├── DocumentBuilderTest.php     # 30 tests - document building
│   │   ├── SolrSchemaTest.php          # 22 tests - schema definitions
│   │   └── SolrSearchAdapterTest.php   # 41 tests - search adapter
│   ├── Subscription/
│   │   ├── PlanManagerTest.php         # 30 tests - pricing, features
│   │   ├── PlanTierTest.php            # 18 tests - tier definitions
│   │   └── SubscriptionStatusTest.php  # 24 tests - subscription workflow
│   └── Swiss/
│       ├── MultiLanguageTest.php       # 22 tests - multi-language content
│       ├── PhoneValidatorTest.php      # 25 tests - Swiss phone validation
│       └── PositionTypeTest.php        # 24 tests - position types
└── Integration/
    └── README.md              # Integration testing guide
```

7.1.4 Test Categories

Category	Classes Tested	Tests
Admin	HelpPageImporter	56
Analytics	EventType	9
Communication	ApplicationStatus	30
Core	AssetOptimizer, StatusWorkflowTrait, SingletonTrait, LocationFormatter, SearchCache	75
Debug	DebugCollector	22
Marketing	SocialShare	27
Matching	Calculator, Criteria	53
Security	Encryption, PasswordPolicy, TOTPManager	72
Solr	DocumentBuilder, SolrSchema, SolrSearchAdapter	93
Subscription	PlanManager, PlanTier, SubscriptionStatus	72
Swiss	PhoneValidator, PositionType, MultiLanguage	71
Total	**36 test files**	**1066 tests, 3035 assertions**

7.1.5 Writing New Tests

```
<?php
namespace Magdev\WpJobroom\Tests\Unit\Example;

use PHPUnit\Framework\Attributes\Test;
use PHPUnit\Framework\Attributes\CoversClass;

#[CoversClass(MyClass::class)]
class MyClassTest extends \WP_JobRoom_TestCase
{
    #[Test]
    public function my_method_does_something(): void
    {
        // Arrange
        $instance = new MyClass();

        // Act
        $result = $instance->doSomething();

        // Assert
```

```
        $this->assertEquals('expected', $result);
    }
}
```

7.2 Integration Testing

7.2.1 Overview

Integration tests verify that multiple components work together with a real WordPress environment.

7.2.2 Setup

1. Install WordPress test library
2. Configure test database
3. Set `WP_TESTS_DIR` environment variable

See tests/Integration/README.md for detailed setup instructions.

7.2.3 Test Scenarios

Workflow	Test Cases
Registration	User creation, email verification, profile linking
Application	Submit, status transitions, notifications
Matching	Score calculation, cache invalidation
Subscription	Payment processing, tier upgrades

7.3 Browser Compatibility Testing

7.3.1 Supported Browsers

Browser	Minimum Version	Status
Chrome	90+	✅ Supported
Firefox	88+	✅ Supported
Safari	14+	✅ Supported
Edge	90+	✅ Supported
Opera	76+	✅ Supported
Internet Explorer	-	❌ Not Supported

7.3.2 Manual Testing Checklist

Registration Flow

- ☐ Applicant registration form loads correctly
- ☐ Company registration form loads correctly
- ☐ Form validation messages display
- ☐ Password strength indicator works
- ☐ Email verification link works

Search & Navigation

- ☐ Search filters work
- ☐ Pagination functions
- ☐ Sort options work
- ☐ Profile cards display correctly

Profile Management

- ☐ Profile editing saves correctly
- ☐ Image uploads work
- ☐ Repeater fields (experience, education) function
- ☐ Privacy settings save

Communication

- ☐ Message inbox loads
- ☐ Compose message works
- ☐ Application form submits
- ☐ Status updates display

Dashboard

- ☐ Analytics charts render
- ☐ Export functionality works
- ☐ Statistics update

7.3.3 Automated Browser Testing

Use Playwright or Cypress for automated browser testing:

```
// Example Playwright test
test('applicant registration', async ({ page }) => {
  await page.goto('/register/applicant/');
  await page.fill('#email', 'test@example.com');
  await page.fill('#password', 'SecureP@ss123');
  await page.click('button[type="submit"]');
  await expect(page).toHaveURL(/verify-email/);
});
```

7.4 Mobile Responsiveness Testing

7.4.1 Breakpoints

Device Class	Breakpoint	Test Focus
Mobile Portrait	< 576px	Navigation, forms, cards
Mobile Landscape	576px - 767px	Two-column layouts
Tablet	768px - 991px	Sidebar visibility
Desktop	992px - 1199px	Full layout
Large Desktop	≥ 1200px	Wide content areas

7.4.2 Testing Tools

1. **Browser DevTools**
   - Chrome DevTools Device Toolbar
   - Firefox Responsive Design Mode
2. **Online Tools**
   - BrowserStack
   - LambdaTest
   - Responsively App

7.4.3 Manual Testing Checklist

Navigation

- ☐ Mobile menu opens/closes
- ☐ All menu items accessible
- ☐ Back navigation works

Forms

- ☐ Input fields are tap-friendly (min 44px)
- ☐ Keyboards don't obscure inputs
- ☐ Form submission works on mobile

Content

- ☐ Images scale correctly
- ☐ Text is readable without zooming
- ☐ Tables scroll horizontally
- ☐ Cards stack on mobile

Touch Interactions

- ☐ Swipe navigation (if applicable)
- ☐ Tap targets are adequately sized
- ☐ No hover-only interactions

7.4.4 Viewport Sizes to Test

```
iPhone SE: 375 x 667
iPhone 12: 390 x 844
iPad: 768 x 1024
iPad Pro: 1024 x 1366
Desktop: 1920 x 1080
```

7.5 Accessibility Testing (WCAG 2.1)

7.5.1 Compliance Level

WP JobRoom targets **WCAG 2.1 Level AA** compliance.

7.5.2 Testing Tools

Tool	Purpose	URL
WAVE	Browser extension for accessibility evaluation	wave.webaim.org
axe DevTools	Automated accessibility testing	deque.com/axe
Lighthouse	Chrome DevTools accessibility audit	Built into Chrome
NVDA	Screen reader testing (Windows)	nvaccess.org
VoiceOver	Screen reader testing (Mac/iOS)	Built into macOS

7.5.3 Manual Testing Checklist

1.1 Text Alternatives

- ☐ All images have alt text
- ☐ Decorative images use `alt=""`
- ☐ Complex images have long descriptions
- ☐ Icons have accessible labels

1.3 Adaptable

- ☐ Proper heading hierarchy (h1 → h2 → h3)
- ☐ Lists use proper markup
- ☐ Tables have headers
- ☐ Forms have proper labels

1.4 Distinguishable

- ☐ Color contrast ratio ≥ 4.5:1 (text)
- ☐ Color contrast ratio ≥ 3:1 (large text)
- ☐ Text can be resized to 200%
- ☐ No loss of content at 320px width

2.1 Keyboard Accessible

- ☐ All functionality available via keyboard
- ☐ No keyboard traps
- ☐ Focus order is logical
- ☐ Focus is visible

2.4 Navigable

- ☐ Skip links present
- ☐ Page titles are descriptive
- ☐ Focus order matches visual order
- ☐ Link purpose is clear

3.1 Readable

- ☐ Page language is set (`<html lang="de-CH">`)
- ☐ Language changes are marked

3.2 Predictable

- ☐ Navigation is consistent
- ☐ Identification is consistent
- ☐ No unexpected context changes

3.3 Input Assistance

- ☐ Error messages are clear
- ☐ Labels are present
- ☐ Error prevention on important actions

4.1 Compatible

- ☐ Valid HTML
- ☐ ARIA attributes used correctly
- ☐ Name, role, value available for custom components

7.5.4 Screen Reader Testing

Test with at least one screen reader:

```
1. Navigate using Tab key
2. Use heading navigation (H key in NVDA)
3. Navigate forms using F key
4. Verify all interactive elements are announced
5. Test form error announcements
```

7.5.5 Color Contrast Testing

Minimum ratios (WCAG AA): - Normal text: 4.5:1 - Large text (18pt+): 3:1 - UI components: 3:1

Use WebAIM Contrast Checker: https://webaim.org/resources/contrastchecker/

7.6 Performance Testing

7.6.1 Metrics to Monitor

Metric	Target	Tool
Time to First Byte	< 200ms	WebPageTest
First Contentful Paint	< 1.8s	Lighthouse
Largest Contentful Paint	< 2.5s	Lighthouse
Time to Interactive	< 3.8s	Lighthouse
Cumulative Layout Shift	< 0.1	Lighthouse

7.6.2 Database Query Testing

```
// Enable query logging
define('SAVEQUERIES', true);

// Check query count
global $wpdb;
echo 'Total queries: ' . count($wpdb->queries);
```

7.6.3 Load Testing

Use tools like: - Apache Bench: `ab -n 1000 -c 10 https://example.com/job-offers/` - k6: https://k6.io/ - Locust: https://locust.io/

7.7 Security Testing

7.7.1 OWASP Top 10 Checklist

- ☐ A01: Broken Access Control
- ☐ A02: Cryptographic Failures
- ☐ A03: Injection
- ☐ A04: Insecure Design
- ☐ A05: Security Misconfiguration
- ☐ A06: Vulnerable Components
- ☐ A07: Authentication Failures
- ☐ A08: Integrity Failures
- ☐ A09: Security Logging
- ☐ A10: SSRF

7.7.2 Security Testing Tools

Tool	Purpose
WPScan	WordPress vulnerability scanner
OWASP ZAP	Web application security scanner
Burp Suite	HTTP proxy and scanner

7.7.3 Manual Security Checks

```
# Check for exposed files
curl -I https://example.com/wp-content/plugins/wp-jobroom/composer.json

# Test CSRF protection (should fail without nonce)
curl -X POST https://example.com/wp-admin/admin-ajax.php \
  -d "action=jr_create_message"

# Test SQL injection
curl "https://example.com/?s=test' OR 1=1--"
```

7.8 CI/CD Integration

Testing is integrated into the Gitea CI/CD pipeline. See `.gitea/workflows/test.yml` for the workflow configuration.

7.8.1 Test Stages

1. **Lint**: PHP syntax checking
2. **Unit Tests**: PHPUnit unit tests
3. **Code Quality**: PHPCS, static analysis
4. **Build**: Verify release package builds

7.8.2 Running CI Tests Locally

```
# Run the same checks as CI
composer validate
composer test:unit
```

7.9 Reporting Issues

When reporting test failures:

1. Include the test name and file

2. Provide the full error message
3. Note the PHP version and environment
4. Include steps to reproduce