Technical Documentation

Generated at 2025-09-03 08:12 UTC

Overview

System Overview

This repository contains a Ruby on Rails application for an Online Learning Management System (LMS).

- Application name: `PortalSystem`
- Ruby: 3.1.2
- Rails: 7.0.8.4 (config defaults set to 5.2 in `config/application.rb`)
- Database: PostgreSQL
- App server: Puma
- Admin UI: ActiveAdmin
- Auth: Devise (students, admin_users)
- Authorization: CanCanCan
- Background jobs: Sidekiq
- PDFs: wicked_pdf + wkhtmltopdf, Prawn
- Frontend: Sprockets pipeline, Turbolinks, jQuery, Bootstrap 4.6

Key directories:

- `app/controllers/` — REST controllers (students, courses, registrations, payments, reports, etc.)
- `app/models/` — ActiveRecord models (e.g., `Student`, `SemesterRegistration`, `Course`, `Invoice`, `MakeupExam`)
- `app/views/` — ERB views for all features
- `app/admin/` — ActiveAdmin resources
- `app/services/` — service objects for domain tasks
- `config/routes.rb` — comprehensive routing configuration

Major features (non-exhaustive):

- Student authentication and profile management
- Semester registration and course add/drop
- Assessments and grade reporting (including online approval and PDF exports)
- Document requests and payments
- Make-up exams, readmissions, transfers, exemptions
- Academic calendar, notices, class and exam schedules
- Admin dashboards and batch operations (ActiveAdmin)

Setup

Setup Guide

This app is a Ruby on Rails monolith.

- Ruby: 3.1.2 (see `.ruby-version`)
- Bundler: 2.4.x (see `Gemfile.lock`)
- Rails: 7.0.8.4
- Node/Yarn: only needed for asset tooling if you add Node-based tools; current app uses Sprockets and does not require Node.
- Database: PostgreSQL 12+
- Redis: required for Sidekiq (background jobs)
- wkhtmltopdf: required for HTML-to-PDF via `wicked_pdf`

1) Clone and bootstrap

```
bundle install
bin/rails db:setup   # creates, loads schema, seeds if any
```

If you get pg compile issues, ensure PostgreSQL client headers are installed and `pg_config` is on PATH.

2) Environment variables

Create `.env` (dotenv-rails) and set:

- `DB_USERNAME`, `DB_PASSWORD`, `DB_HOST`, `DB_PORT` — see `config/database.yml`
- `PORTAL_SYSTEM_DATABASE_PASSWORD` — production `database.yml`
- Any additional secrets used by integrations (e.g., Moodle if configured)

See `docs/configuration.md` for details.

3) Run the app

```
# Start Rails server
bin/rails s

# In another terminal, start Sidekiq (if jobs are used)
bundle exec sidekiq
```

Visit http://localhost:3000

- Public root: `pages#home`
- Admin: `/admin` (ActiveAdmin)

4) wkhtmltopdf

Install wkhtmltopdf 0.12.6+ compatible with your OS to support `wicked_pdf`.

5) Seeds and access

Check `db/seeds.rb` (if present) for initial users. For ActiveAdmin, create an admin user via console if none exists:

```
AdminUser.create!(email: "admin@example.com", password: "password", password_confirmation: "password")
```

6) Tests

```
bundle exec rspec
```

See `docs/testing.md`.

Configuration

Configuration

Environment variables

- Database (see `config/database.yml`)
  - `DB_USERNAME`
  - `DB_PASSWORD`
  - `DB_HOST`
  - `DB_PORT`
  - Production: `PORTAL_SYSTEM_DATABASE_PASSWORD` if using user `portal_system`
- Rails
  - `RAILS_MAX_THREADS` (optional)
  - `RAILS_ENV` (`development`/`test`/`production`)
- Redis
  - `REDIS_URL` (if not default, for Sidekiq)
- PDF
  - `WKHTMLTOPDF_BINARY` (optional if not on PATH)
- Integrations
  - Moodle: environment variables as required by `moodle_rb` (e.g., endpoint URL, token). Consult integration code if used.

Store local values in `.env` (dotenv-rails), never commit secrets.

Rails configuration

- `config/application.rb`
  - `config.load_defaults 5.2` with Rails 7 gems — legacy defaults retained
  - Asset pipeline via Sprockets; precompiles ActiveAdmin and font assets

Credentials and secrets

Consider using `rails credentials:edit` for production secrets or environment-based secret management.

Operations

Operations

Processes

- Web: Puma (via `rails s` or your process manager)
- Background: Sidekiq (`bundle exec sidekiq`)

Health

- App boots with root at `pages#home`
- Admin at `/admin`

Jobs

- Sidekiq required if any async jobs are enqueued (check `app/jobs/`)
- Configure Redis via `REDIS_URL`

PDFs

- `wicked_pdf` uses wkhtmltopdf for HTML-to-PDF
- `prawn` used for programmatic PDFs (e.g., grade reports)

Common Rake tasks

```
bin/rails db:migrate
bin/rails db:seed
bin/rails assets:precompile RAILS_ENV=production
```

Deployment

- Capistrano included (see gems: `capistrano`, `capistrano-rails`, `capistrano-passenger`, `capistrano-rbenv`, `capistrano-rails-db`)
- Ensure:
  - System Ruby or rbenv with Ruby 3.1.2
  - PostgreSQL, Redis available
  - wkhtmltopdf installed

Logs

- Rails logs under `log/`
- Sidekiq logs depend on your process manager; by default to STDOUT

Background schedules

- `whenever` gem present for cron-based schedules. Check `config/schedule.rb` (if present) for jobs.

Architecture

Architecture

Layers

- Controllers (`app/controllers/`): RESTful endpoints for students, courses, registrations, payments, reports, etc.
- Models (`app/models/`): ActiveRecord; key domain models include `Student`, `SemesterRegistration`, `Course`, `Invoice`, `Assessment`, `MakeupExam`.
- Views (`app/views/`): ERB templates; PDFs rendered via wicked_pdf or Prawn.
- Admin (`app/admin/`): ActiveAdmin resources for administration.
- Services (`app/services/`): encapsulated domain operations.
- PDFs (`app/pdfs/`): Prawn PDF builders.

Authentication & Authorization

- Devise for `Student` and `AdminUser`.
- CanCanCan (`Ability`) centralizes permissions (`app/models/ability.rb`).

Routing

- See `config/routes.rb` for comprehensive routes including:
  - Course management, assessments, assessment plans, results
  - Semester registrations, add/drop, transfers, readmissions
  - Document requests with payment and receipt upload
  - Reports (grade, student, payment, course assignments)
  - Admin namespaces for registration payments, sections, schedules

Data storage

- PostgreSQL primary data store.
- ActiveStorage enabled with validations; drag-and-drop support included.

Background jobs

- Sidekiq with Redis.

PDFs

- WickedPDF + wkhtmltopdf for HTML-based PDFs.
- Prawn + prawn-table for programmatic PDFs.

Data Model

Data Model Notes

Key models and responsibilities (non-exhaustive):

- `Student` — authentication, profile, enrollments, grades
- `SemesterRegistration` — registration lifecycle, fees, invoices
- `Course`, `CourseModule`, `Section` — curriculum and delivery structure
- `AssessmentPlan`, `Assessment`, `AssessmentResult`, `StudentGrade`, `GradeReport` — evaluation and reporting
- `Invoice`, `Payment`, `PaymentTransaction`, `PaymentMethod` — billing and payments
- `MakeupExam`, `Readmission`, `ExternalTransfer`, `ProgramExemption`, `Exemption` — student lifecycle exceptions
- `AcademicCalendar`, `ClassSchedule`, `ExamSchedule` — timelines and schedules
- `Notice`, `DocumentRequest` — communications and official documents
- `AdminUser`, `Ability` — admin and authorization

See `app/models/` for the full list. Start with:

- `app/models/semester_registration.rb` (complex domain logic)
- `app/models/student.rb`
- `app/models/student_grade.rb`
- `app/models/makeup_exam.rb`

Routes

Routes Highlights

See `config/routes.rb` for full details. Notable endpoints:

- Root: `GET / => pages#home`
- Student auth: Devise routes under `/students`
- Admin: ActiveAdmin under `/admin`
- Semester registration:
  - `GET /new/semester/registration => pages#enrollement`
  - `POST /create/semester/registration => pages#create_semester_registration`
- Assessments:
  - `resources :assessment_plans` nested under `courses`
  - `resources :assessmens` with collection routes: `bulk_create`, `find_course`, `missing_assessments_report`
- Documents:
  - `resources :document_requests` with `payment`, `upload_receipt`, `track_form`, `track`
- Reports:
  - `GET /course_assignments => reports#course_assignments`
  - Multiple student and payment reports under `student_report_*` and `payment_report_*`
- Payments & invoices:
  - `resources :invoices`, `payment_methods`, `payment_transactions`
- Exceptions:
  - `resources :makeup_exams` with `payment`, `verify`
  - `resources :readmissions` with `payment`, `verify`
  - `resources :external_transfers` with `filter_programs`, `payment`

Admin

Admin (ActiveAdmin)

- Access via `/admin`
- Manage key resources (students, courses, registrations, payments, schedules, etc.)
- Gems:
  - `activeadmin`, `activeadmin_addons`, themes (`active_admin_theme`, `active_admin_flat_skin`)
  - Import/export via `active_admin_import`
- Check `app/admin/` for resource definitions and custom actions.

Testing

Testing

- Framework: RSpec (`rspec-rails`)
- Spec directory: `spec/`
- Run all tests:

```
bundle exec rspec
```

Suggested structure (if not already present):

- `spec/models/*_spec.rb`
- `spec/controllers/*_spec.rb`
- `spec/requests/*_spec.rb`
- `spec/features/*_spec.rb`

Use factories (FactoryBot) if added; currently not listed in Gemfile.

Security

Security & Access

- Authentication via Devise; ensure secure password policies in production.
- Authorization via CanCanCan (`Ability`); review rules in `app/models/ability.rb`.
- Admin access via ActiveAdmin at `/admin`.
- Do not commit secrets. Use environment variables and/or Rails credentials.
- Force SSL in production (via environment config) behind a TLS terminator.
- Keep wkhtmltopdf binaries up-to-date and from trusted sources.
- Background job queues should be on private networks; secure Redis with auth if exposed.

Troubleshooting

Troubleshooting

PostgreSQL (pg) compile errors

- Ensure `pg_config` is on PATH
- On macOS with Homebrew: `brew install postgresql` and re-run `bundle install`

wkhtmltopdf not found

- Install wkhtmltopdf and/or set `WKHTMLTOPDF_BINARY`

Redis/Sidekiq connection errors

- Set `REDIS_URL` or start local Redis (`brew services start redis`)

Asset compilation issues

- Precompile: `bin/rails assets:precompile`
- Ensure Sprockets and gems (sass-rails, uglifier) are installed

PDF generation layout issues

- Check CSS specificity and print CSS; wkhtmltopdf uses WebKit rendering