PIAE — Dokumentace projektu

Translation Service (FastAPI + MongoDB + Django)

Martin Reich

Abstrakt

Projekt **PIAE** je jednoduchá služba pro správu překladatelských projektů. Backend je implementován ve **FastAPI** a ukládá data do **MongoDB**; soubory ukládá do **GridFS**. Frontend je server-renderovaný **Django** web, který komunikuje s backendem přes HTTP. Projekt obsahuje JWT autentizaci a volitelně OTP (TOTP) přihlášení.

Obsah

1 Přehled a cíle

1.1 Cíl systému

Cílem je umožnit:

- zákazníkovi vytvořit překladatelský projekt (upload souboru + cílový jazyk),
- automaticky přiřadit vhodného překladatele dle jazyků a vytíženosti,
- překladateli stáhnout originál a nahrát překlad,
- zákazníkovi schválit/zamítnout výsledek a poskytnout zpětnou vazbu,
- administrátorovi řešit zpětnou vazbu, posílat zprávy a uzavírat projekty.

1.2 Role

- **CUSTOMER** — vytváří projekty, hodnotí a posílá feedback.
- **TRANSLATOR** — spravuje cílové jazyky, odevzdává překlady.
- **ADMINISTRATOR** — správa feedbacku, komunikace, uzavírání projektů.

2 Architektura

2.1 Komponenty

Projekt je rozdělen do dvou aplikací a infrastruktury:

- **Backend** (`/Backend`): FastAPI + MongoDB (Motor) + GridFS.
- **Frontend** (`/Frontend`): Django UI volající backend přes HTTP.
- **Infrastruktura**: Docker Compose (MongoDB, MailHog, backend, frontend).

2.2 Logická vrstvená architektura backendu

- **API vrstva**: `Backend/app/api/*` — routy, validace, autorizace.
- **Service vrstva**: `Backend/app/services/*` — business logika.
- **Repository vrstva**: `Backend/app/repositories/*` — dotazy do MongoDB.
- **Domain model**: `Backend/app/domain/*` — Pydantic modely a enumy.

2.3 Komunikace

Frontend ukládá JWT do session a posílá ho do backendu v hlavičce:

```
Authorization: Bearer <token>
```

3 Backend

3.1 Technologie a odpovědnosti

Backend je implementovaný ve **FastAPI** a poskytuje REST API. Odpovídá za:

- autentizaci (JWT) a volitelně OTP (TOTP),
- správu uživatelů a rolí (CUSTOMER/TRANSLATOR/ADMINISTRATOR),
- správu projektů (stavový automat, přiřazení překladatele),
- ukládání souborů do GridFS (originál a překlad),
- zasílání notifikací e-mailem (v dev prostředí přes MailHog).

3.2 Vrstevní architektura

- `app/main.py`: inicializace aplikace, CORS, routery.
- `app/api/*`: HTTP rozhraní (FastAPI routy).
- `app/services/*`: orchestrace a business logika (např. tvorba projektu, assignment).
- `app/repositories/*`: perzistence a dotazy do MongoDB.
- `app/db/*`: přístup k MongoDB a GridFS.
- `app/security/*`: JWT, hashování hesel.

3.3 Ukládání souborů (GridFS)

Soubory se neukládají přímo do dokumentu projektu, ale do **GridFS**. Projekt pak nese pouze reference (id) na originální a přeložený soubor.

3.4 Přiřazování překladatele

Při vytváření projektu proběhne automatické přiřazení překladatele podle:

- podporovaných cílových jazyků (TranslatorLanguage),
- aktuální vytíženosti (počtu aktivních projektů na překladatele).

Pokud systém nenajde vhodného překladatele, projekt se uzavře (`CLOSED`) a zákazník je informován e-mailem.

4 Frontend

4.1 Technologie a odpovědnosti

Frontend je server-renderovaná aplikace v **Django**. Zajišťuje:

- UI pro všechny role (obsah se řídí rolí v session),
- ukládání JWT do Django session,
- volání backend API přes interní HTTP klient (`web/backend_client.py`),

- zobrazení seznamů projektů (customer/translator/admin) a detailů,
- upload souborů (vytvoření projektu, odevzdání překladu),
- lokalizaci (EN/CS) přes Django i18n.

4.2 Struktura frontendu

- `web/views.py`: hlavní view funkce pro UI.
- `templates/*`: HTML šablony (base layout, stránky pro projekty, login, languages, ...).
- `static/*`: CSS/JS.
- `locale/*`: překlady (`django.po`/`django.mo`).

4.3 Autentizace ve frontendu

Po úspěšném loginu frontend uloží JWT token do session a přidává ho do požadavků na backend jako hlavičku `Authorization: Bearer ...`.

5 Datový model

5.1 MongoDB kolekce

Aplikace používá několik kolekcí a GridFS bucket:

- `users`
- `projects`
- `translator_languages`
- `feedback`
- GridFS bucket `files.files` + `files.chunks`

5.2 Doménové entity

5.2.1 User

Uživatel obsahuje id, jméno, email, roli, heslový hash a volitelně OTP secret.

5.2.2 TranslatorLanguage

Entita mapuje překladatele na cílový jazyk (ISO 639-1). Slouží pro výběr překladatele při vytváření projektů.

5.2.3 Project

Projekt obsahuje vazby na zákazníka/překladatele, cílový jazyk, GridFS id souborů a stavový automat.

5.2.4 Feedback

Feedback je 1:1 k projektu a nese text a čas vytvoření.

6 Stavový automat projektu

Stavy (`ProjectState`):

- CREATED — projekt vytvořen
- ASSIGNED — přiřazen překladateli
- COMPLETED — překlad nahrán
- APPROVED — zákazník schválil
- CLOSED — uzavřen

7 Produkční tok (hlavní use-cases)

7.1 Vytvoření projektu (Customer)

1. Customer odešle `POST /projects` s jazykem a souborem.
2. Backend uloží soubor do GridFS.
3. Backend vytvoří dokument projektu (state=CREATED).
4. Proběhne přiřazení překladatele (Assignment).
5. Projekt se nastaví na ASSIGNED nebo CLOSED.
6. Odešle se email (překladateli nebo zákazníkovi).

7.2 Odevzdání překladu (Translator)

1. Translator stáhne originál `GET /projects/id/original`.
2. Nahraje překlad (endpoint ve `/projects` routách).
3. Projekt přejde typicky do COMPLETED.

7.3 Schválení / zamítnutí (Customer)

1. Customer otevře detail projektu.
2. Provede approve/reject a volitelně/povinně přidá feedback.

7.4 Admin práce se zpětnou vazbou

1. Admin zobrazí seznam projektů se zpětnou vazbou.
2. Může poslat zprávu zákazníkovi/překladateli.
3. Může projekt uzavřít.

8 API specifikace (stručně)

Pozn.: Kompletní seznam endpointů je definován na <backend_uri>:<port>/docs.

8.1 Auth

- `POST /auth/login` — přihlášení heslem, vrací JWT.
- `POST /auth/otp/enable` — vygeneruje OTP secret a provisioning URI.
- `POST /auth/otp/login` — přihlášení TOTP.

8.2 Users

- `POST /users/customers/register`
- `POST /users/translators/register`
- `GET /users/id`
- `GET/POST/DELETE /users/translators/translator_id/languages`

8.3 Projects

- `POST /projects` — vytvoří projekt (Customer).
- `GET /projects` — list dle role.
- `GET /projects/id` — detail.
- `GET /projects/id/original` — download originálu.

8.4 Feedback

- `GET /feedback/projects/project_id` — feedback pro projekt.

9 Bezpečnost

9.1 JWT

Backend generuje JWT token s informací o `user_id` a `role`. Frontend token ukládá do session.

9.2 OTP (TOTP)

OTP je volitelné a navazuje na prvotní password login. Backend generuje per-user secret a ověřuje TOTP kód.

10 Testování

10.1 Backend testy

Testy v `Backend/tests` obsahují:

- unit testy business logiky (např. ProjectService a AssignmentService),
- integrační test repository vrstvy (MongoDB),
- API testy pro validace (např. velikost uploadu, role access).

Spuštění:

```
cd Backend
python -m pytest
```

11 Spuštění a deployment

11.1 Docker Compose

Repo obsahuje `docker-compose.yml` se službami: MongoDB, MailHog, backend a frontend. Stačí v root složce projektu dát:

```
docker compose up -d --build
```

11.2 Exponované porty

- Frontend UI: http://localhost:8001
- Backend API: http://localhost:8000
- MailHog UI: http://localhost:8025
- MongoDB: `mongodb://localhost:27017`

11.3 Konfigurace

11.3.1 Backend (FastAPI)

Klíčové proměnné prostředí (nastavené v Docker Compose):

- `MONGODB_URI`, `MONGODB_DB`
- `MAX_UPLOAD_MB`
- `JWT_SECRET`
- `CORS_ALLOW_ORIGINS`
- `SMTP_HOST`, `SMTP_PORT`, `SMTP_FROM`
- `OTP_MASTER_SECRET`

11.3.2 Frontend (Django)

- `BACKEND_API_BASE_URL` (např. `http://backend:8000` v Dockeru)

12 Závěr

Projekt demonstruje kompletní tok překladatelského projektu od vytvoření až po schválení včetně OTP autentizace, ukládání souborů v GridFS a role-based UI.