

PIE - návrh řešení

Roman Pejs

October 15, 2025

1 Forma řešení

Projekt bude rozdělen na client-server architekturu. Server bude poskytovat REST API pro komunikaci s klientem.

2 Rozdělení do modulů

- **Core module** - základní funkce a datové struktury
- **UI module** - uživatelské rozhraní
- **Storage module** - správa dat a připojení
- **Auth module** - autentizace a autorizace uživatelů

3 Moduly detailně

3.1 Core module

- Obsahuje datové struktury (doménové) viz Github
- Services - logika aplikace
- Obsahuje Interface pro komunikaci s databází

3.2 RestApi module

- Komunikace s uživatelem (vybrané API)
- Zpracování požadavků a odpovědí
- Mapování DTO objektů ↔ doménové objekty (např. pomocí MapStruct)

3.3 Storage module

- Implementace Interface pro komunikaci s databází
- MongoDB připojení a operace
- Přidání databázových modelů a (domain ↔ db model) mapování

3.4 Auth module ??

- Security filter chain pro zabezpečení API
- Kontrola JWT tokenů
- Správa uživatelů a rolí

Poznámka: Auth modul může být integrován přímo do RestApi modulu a core, pokud je to jednodušší.

4 Flow aplikace

- Uživatelský požadavek přijde na UI modul
- Security filter chain z Auth modulu ověří požadavek + token + uživatele
- UI modul zavolá Core modul (services) pro zpracování logiky
- Core modul může komunikovat se Storage modulem pro přístup k datům
- Výsledek je vrácen zpět UI modulu, který odpoví uživateli

5 Navrh REST API

6 REST API Endpoints

Base Information

Base URL: /api/v1

Format: application/json

Authentication: Bearer JWT (Authorization: Bearer <token>)

Authentication

POST	/auth/register	– Register new user (CUSTOMER or TRANSLATOR)
POST	/auth/login	– Login user, returns access + refresh token
POST	/auth/refresh	– Refresh access token
POST	/auth/logout	– Logout, invalidate refresh token
GET	/me	– Get current logged-in user info

Users

GET	/users	– (ADMIN) List all users (filters: page, perPage)
GET	/users/{id}	– (ADMIN or SELF) Get user details
PATCH	/users/{id}	– (ADMIN or SELF) Update user info
DELETE	/users/{id}	– (ADMIN) Delete user

Translator Languages

GET	/users/{id}/languages	– (ADMIN or SELF) Get translator's languages
PUT	/users/{id}/languages	– (ADMIN or SELF) Replace all languages
POST	/users/{id}/languages	– (ADMIN or SELF) Add a new language
DELETE	/users/{id}/languages/{languageCode}	– (ADMIN or SELF) Remove a language

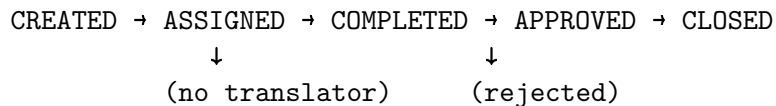
Projects

POST	/projects	
– (CUSTOMER)	Create new project (multipart/form-data)	
GET	/projects	
– (CUSTOMER/TRANSLATOR/ADMIN)	List projects	
GET	/projects/{id}	
–	Get project details	
GET	/projects/{id}/original	
–	Download original file	
GET	/projects/{id}/translation	
–	Download translated file	
PUT	/projects/{id}/translation	
– (TRANSLATOR)	Upload translated file (state	COMPLETED)
POST	/projects/{id}/approval	
– (CUSTOMER)	Approve translation (state	APPROVED)
POST	/projects/{id}/rejection	
– (CUSTOMER)	Reject translation + feedback	
POST	/projects/{id}/assignment	
– (ADMIN)	Assign translator manually	
POST	/projects/{id}/messages	
– (ADMIN)	Send message to customer/translator	
POST	/projects/{id}/close	
– (ADMIN)	Close project (state	CLOSED)

Feedback

GET	/projects/{id}/feedback	(CUSTOMER/TRANSLATOR/ADMIN) Get feedback
PUT	/projects/{id}/feedback	(CUSTOMER) Add or update feedback
GET	/feedbacks	(ADMIN) List all feedbacks (for review)

Project State Flow



↓
CLOSED

↓
CLOSED (after admin review)

Common Query Parameters

```
?state=ASSIGNED|COMPLETED|...  
?languageCode=es  
?hasFeedback=true  
?sort=-createdAt  
?page[size]=20&page[number]=1
```