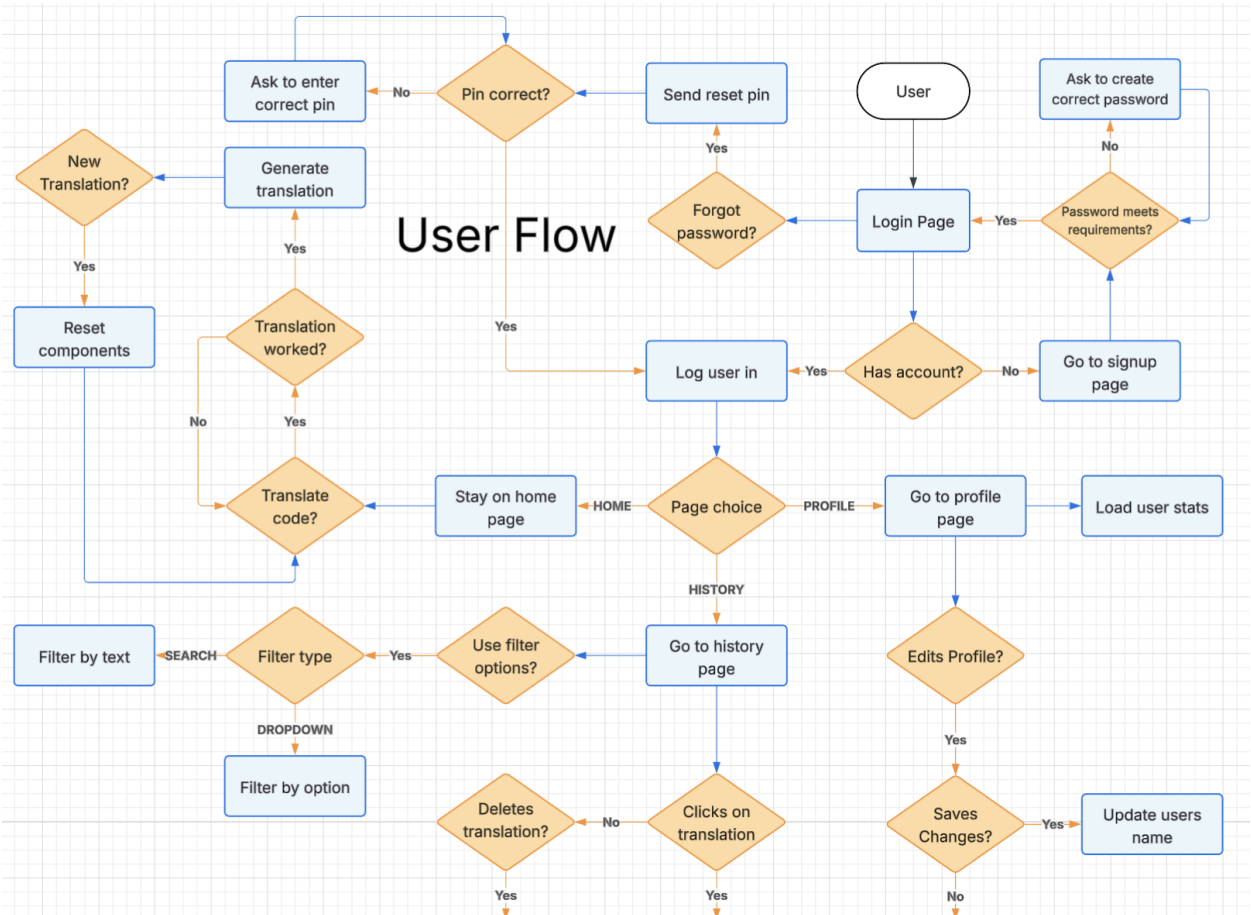
Final Design

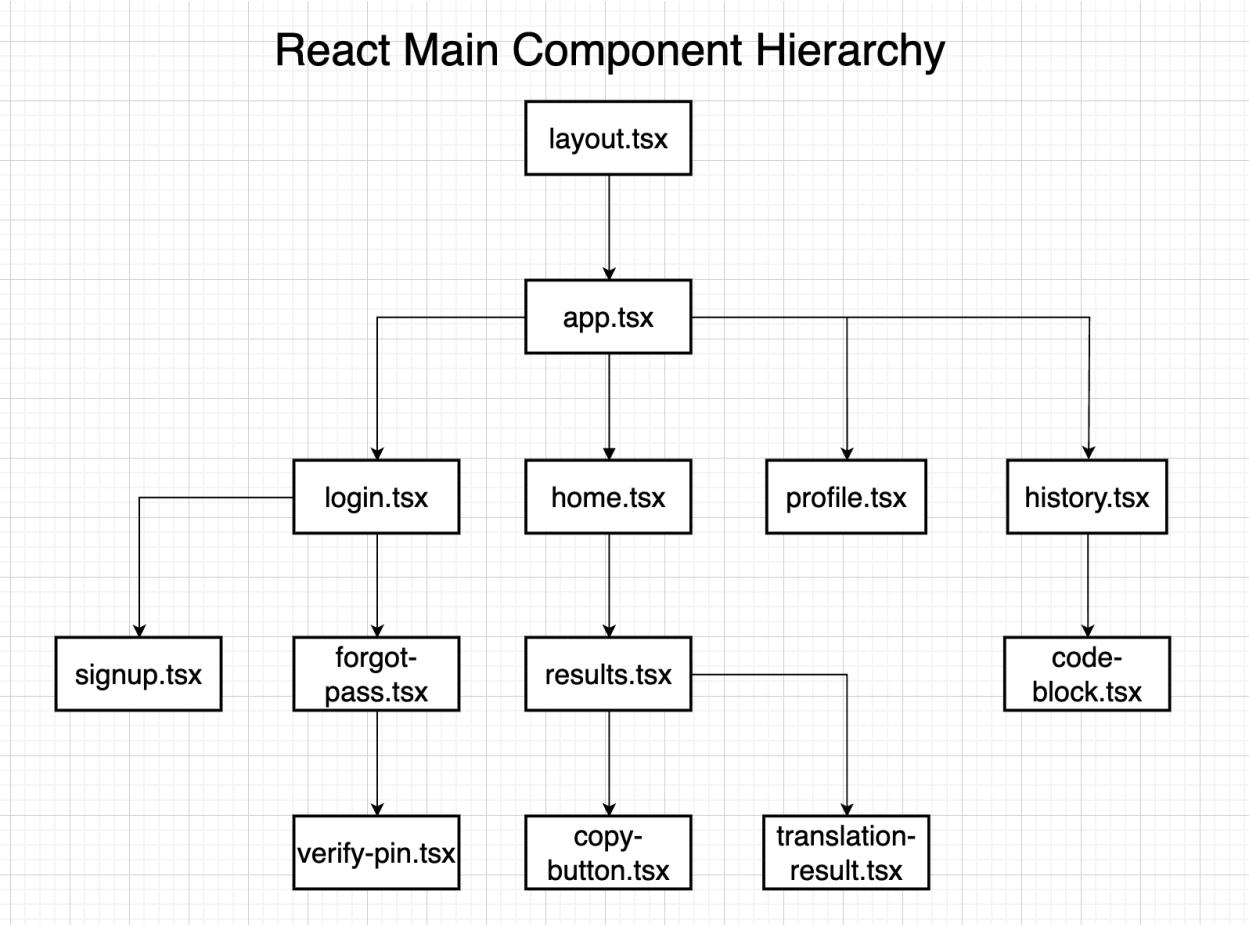
System Architecture

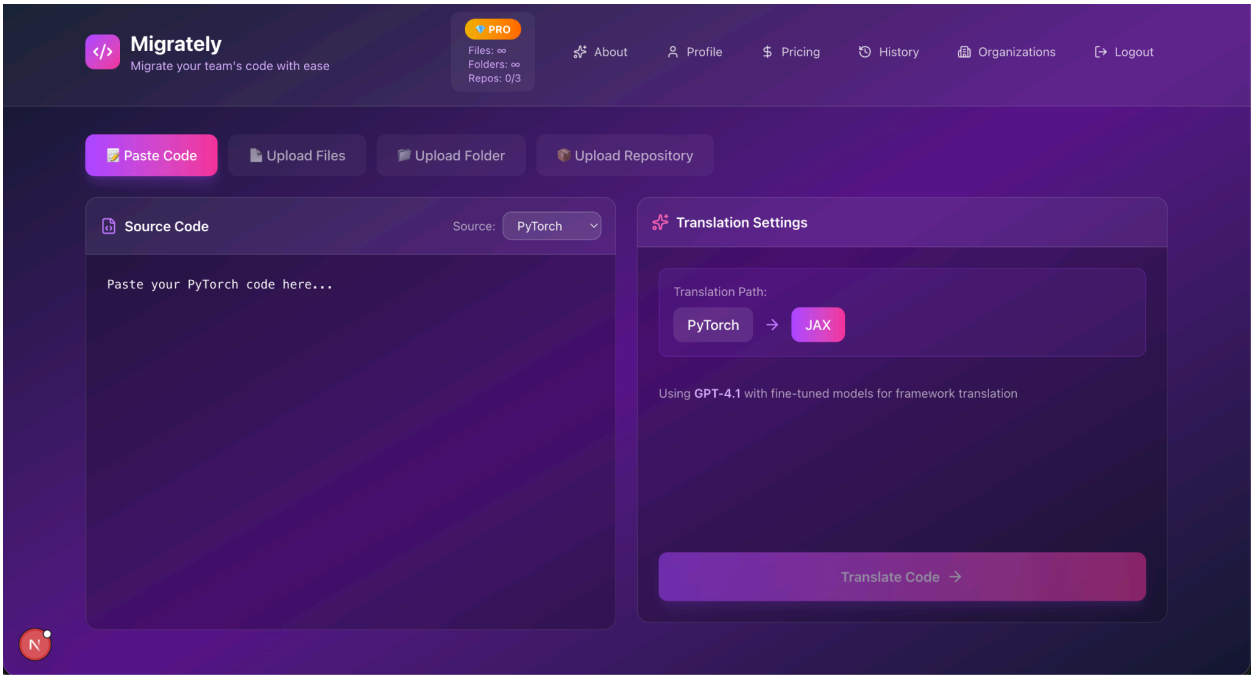
The application includes a NextJS with React frontend FastAPI with SQLAlchemy backend, and Google Cloud SQL with MySQL database for storage. AI services come from OpenAI's GPT models via the OpenAI API key system. The frontend and backend are deployed using Cloud Build/Cloud Run. Application secrets are held in Cloud Secret Manager. REST API protocol is used to transfer data from the backend to the frontend and vice versa.

Frontend

The frontend is designed using NextJS and React. It is a web application available at https://code-translation-frontend-283805296028.us-central1.run.app/. Typescript is utilized for communication with the backend. Below is a depiction of some of the user flow possible for our application, a react component diagram and the home screen of our application:




React Main Component Hierarchy
layout.tsx
app.tsx
login.tsx
home.tsx
profile.tsx
history.tsx
signup.tsx
forgot-pass.tsx
results.tsx
code-block.tsx
verify-pin.tsx
copy-button.tsx
translation-result.tsx


Migrately
Migrate your team's code with ease
PRO
Files: ∞
Folders: ∞
Repos: 0/3
About
Profile
Pricing
History
Organizations
Logout
Paste Code
Upload Files
Upload Folder
Upload Repository
Source Code
Source:
PyTorch
Paste your PyTorch code here...
Translation Settings
Translation Path:
PyTorch
JAX
Using GPT-4.1 with fine-tuned models for framework translation
Translate Code

Backend

The backend is designed using FastAPI and SQLAlchemy. The url for the backend is https://code-translation-api-283805296028.us-central1.run.app/. FastAPI is used for communication with the frontend and SQLAlchemy is used for communication with the database. Below is a table of APIs our application employs:

Auth

Endpoint	Method	Description
/login	POST	Logs a user in
/request-password-reset	POST	Requests a password reset for current user
/verify-reset-pin	POST	Verifies the password reset token
/reset-password	POST	Resets the user password with the given token

Users

Endpoint	Method	Description
/register	POST	Registers a user in the system
/me	PUT	Updates a user's info
/me	POST	Gets the current user
/	GET	Get all users
/me/rate-limit	GET	Gets current user's rate limit info
/me/tier	PUT	Gets current user's tier
/me	DELETE	Deletes the current user's account

Translations

Endpoint	Method	Description
/batch/{batch_id}	GET	Get all translations by batch
/{translation_id}	DELETE	Deletes a translation
/	DELETE	Deletes all translations for the current user
/stats	GET	Gets stats for the current user
/addTranslation	POST	Adds a translation to the db
/	GET	Get translation history for current user

Generate

Endpoint	Method	Description
/	POST	Generate code using OpenAI API
/batch	POST	Translate multiple files at once
/repo	POST	Clones repo, finds relevant files, and translates them all

Organizations

Endpoint	Method	Description
/	POST	Creates a new organization
/	GET	Lists all organizations where the user is a member
/{org_id}	GET	Gets an organization

/{org_id}	PUT	Updates an organization
/{org_id}	DELETE	Deletes an organization
/{org_id}/members	GET	Get all members of an organization
/{org_id}/members/{member_id}	PUT	Updates a member's role within an organization
/{org_id}/members/{member_id}	DELETE	Deletes a member from an organization
/{org_id}/transfer	POST	Transfer ownership to another member of the organization
/{org_id}/invites	GET	List all active invite links
/{org_id}/invite-links	POST	Creates a new invite link for the organization
/{org_id}/invite-links/{invite_id}	DELETE	Deletes an invite link
/{org_id}/invites	POST	Invite a member to the organization

Projects

Endpoint	Method	Description
/	POST	Creates a new project
/org/{org_id}	GET	Verifies user is a member of an organization
/{project_id}	GET	Gets a project
/{project_id}	PUT	Updates a project
/{project_id}	DELETE	Deletes a project
/{project_id}/translations	GET	Gets all translations for a specific project
/{project_id}/stats	GET	Get translation stats for this

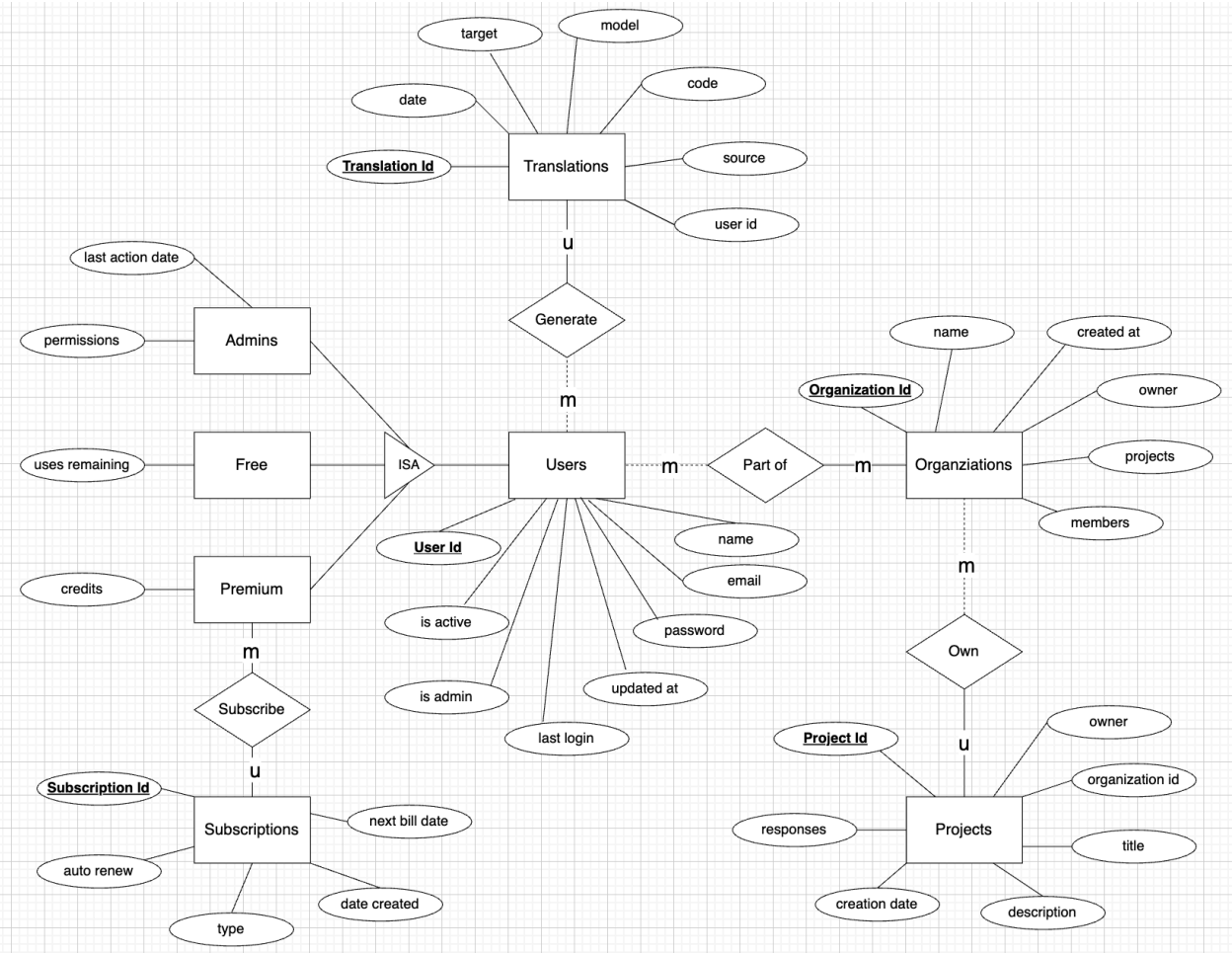
		project

Notifications

Endpoint	Method	Description
/translation-complete	POST	Sends email when a translation completes to the current user

Database

The database is designed using Google Cloud SQL and MySQL. The specific schema we used was code-translator-db. The link to it is here: https://console.cloud.google.com/sql/instances/code-translator-db/overview?project=coms-4020-team-12. All data is stored here for the project. Below is an ER diagram of our database:



Technologies Used

- FastAPI - api services
- Docker - conterization, CI/CD
- MySQL - database schemas
- GCP - storage, database, secrets, deployment, logging
- SQLAlchemy - database migration
- React - modern website design
- Python - backend scripting
- Typescript - communication with backend
- Javascript - frontend
- OpenAI API - access to OpenAI's models
- Gitlab - codebase
- HuggingFace Hub - datasets for research
- Sphinx - API Documentation
- PyPI - for dev tool
- Colab - data analysis
- NextJS - server for frontend