TalentElite-Advanced Employee Management System

23/24: LZPMPC005M: Programming Clinic

Arthur and Roll Number: Vincent 39226719

on May 26, 2024

1 Introduction

This is a corporate employee management system that includes login authentication, employee information display, logging, email mass mailing, and database backup functionalities. The project employs various front-end and back-end technologies to achieve a comprehensive and user-friendly enterprise management solution.

For futher informations please visit TalentElite:Github, and if you have suggestions and any comments please do not hesitate to contact at y.cheng22@lancaster.ac.uk.

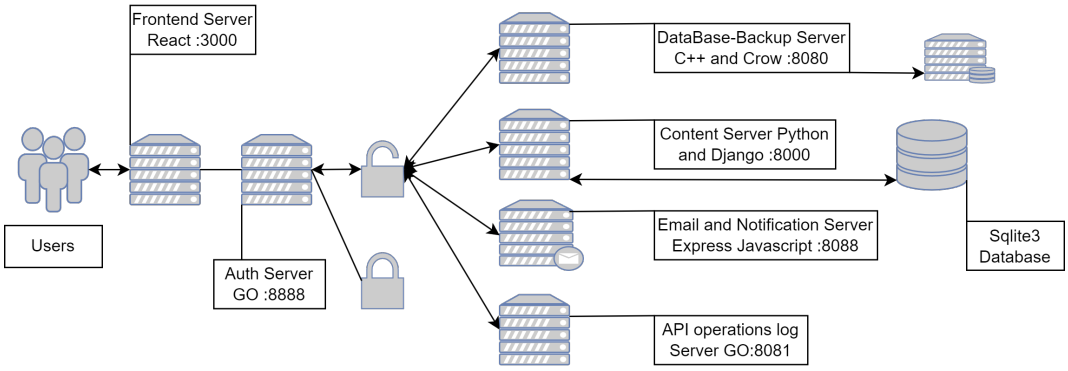

Figure 1: System structure.

2 Back-end Technologies

2.1 Go (Golang) Back-end

- Handles user login authentication, generates and returns JWT tokens.
- Provides logging API, recording all accesses to the Django API.
- Main technologies: gorilla/mux for routing, golang-jwt/jwt for JWT token generation and verification.

2.2 Django Back-end

- Manages corporate employee information (CRUD operations).
- Provides API endpoints for front-end access, including employee information and department information.
- Used for logging and backup functionalities.
- Main technologies: Django framework, Django REST framework, JWT authentication.

2.3 C++ Back-end (Crow)

- Handles the database backup functionality.
- Periodically backs up the Django project's database and stores the backup files in a specific directory.
- Main technologies: C++ language, Crow library (for simple HTTP server), file operations.

2.4 Express.js Back-end

- Handles the email mass mailing functionality.
- Sends emails to all employees via an SMTP server.
- Main technologies: Node.js, Express framework, nodemailer library.

3 Front-end Technologies

3.1 React Front-end

- Used to build the user interface, displaying login page, employee information pages, etc.
- Communicates with the back-end via Axios, sending and receiving data.
- Main technologies: React framework, react-router-dom for routing management, Axios for HTTP requests.

3.2 Semi Design UI Library

- Used to beautify the front-end interface, providing a set of modern UI components.
- Main technologies: @douyinfe/semi-ui library, @douyinfe/semi-icons library.

4 Functionality Summary

4.1 User Login

- The front-end submits the username and password through a login form.
- The Go back-end verifies user information and generates a JWT token.
- Upon successful login, the front-end stores the JWT token in localStorage and redirects to the employee information page.

4.2 Employee Information Management

- The front-end requests employee information from the Django back-end API via Axios and displays it.
- Supports CRUD operations for employee information.

4.3 Logging

- The Go back-end provides logging functionality, recording all accesses to the Django API.
- The front-end retrieves and displays log data via Axios requests.

4.4 Email Mass Mailing

- The Express.js back-end handles the email mass mailing functionality.
- Sends emails to all employees via an SMTP server.

4.5 Database Backup

- The C++ back-end uses the Crow library to periodically back up the Django project's database.
- Backup files are stored in a specified directory.

5 UML diagram

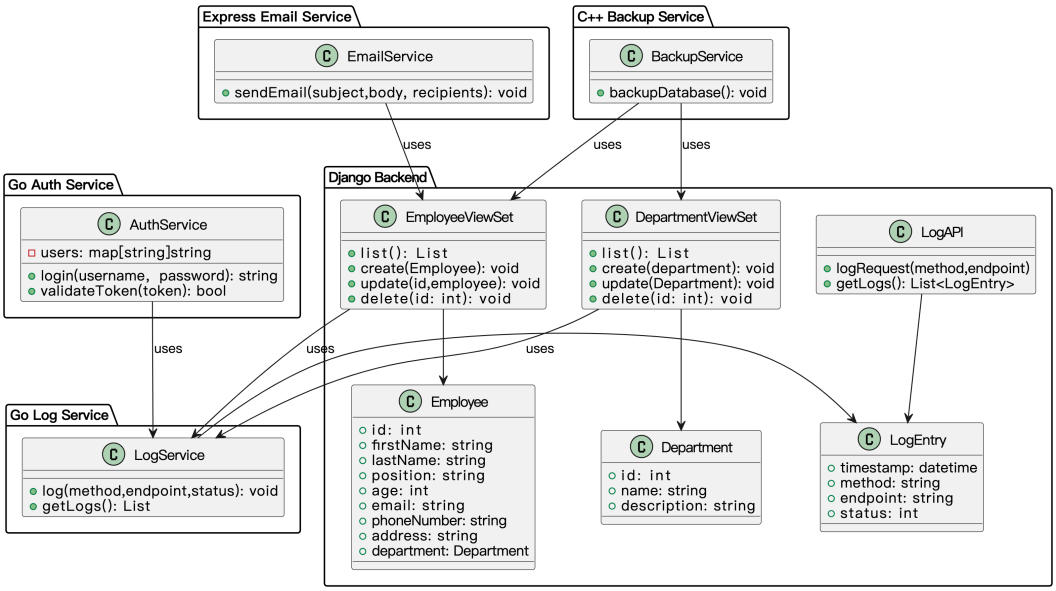

Figure 2: Class Diagram.

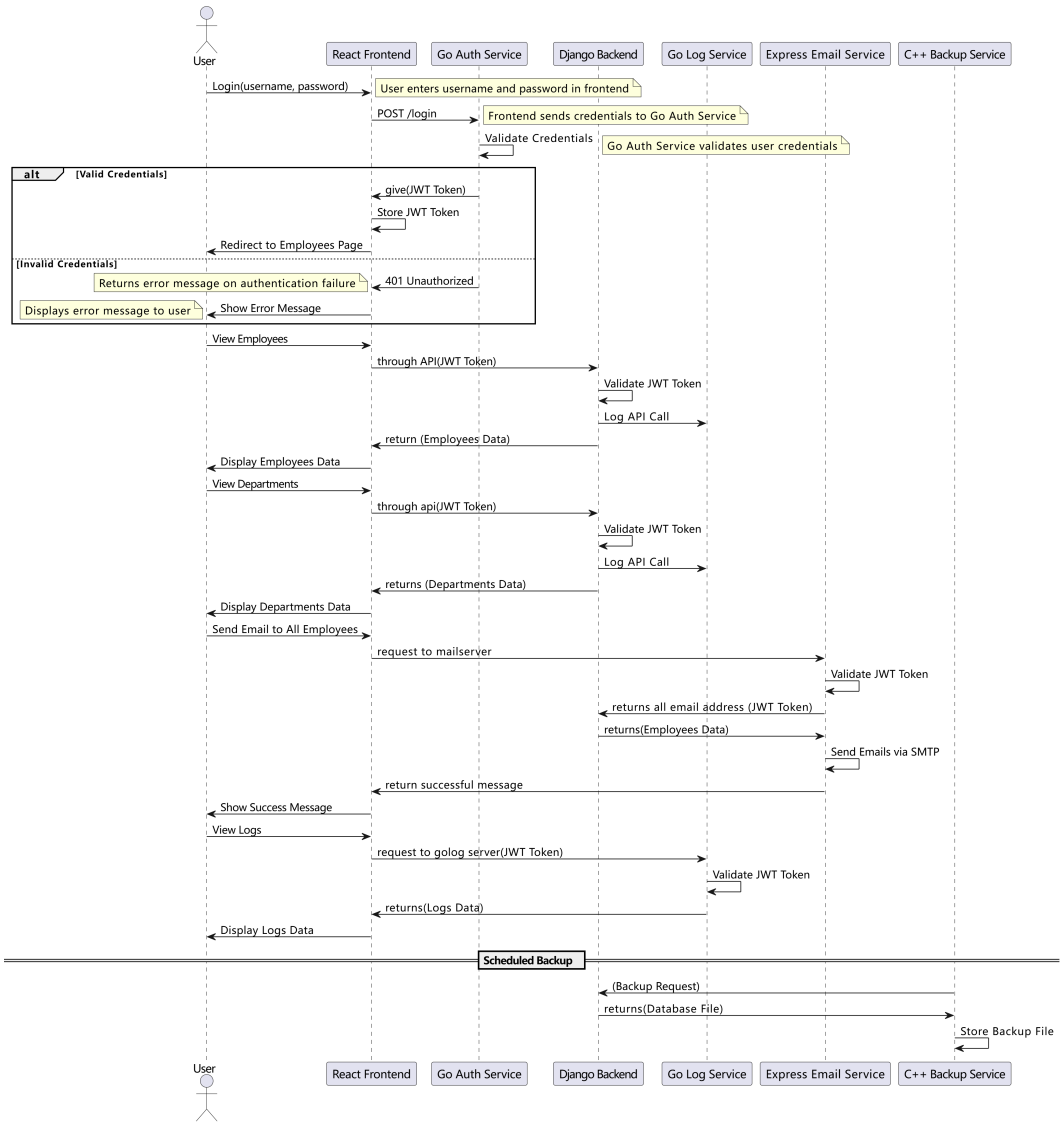

Figure 3: Sequence Diagram.

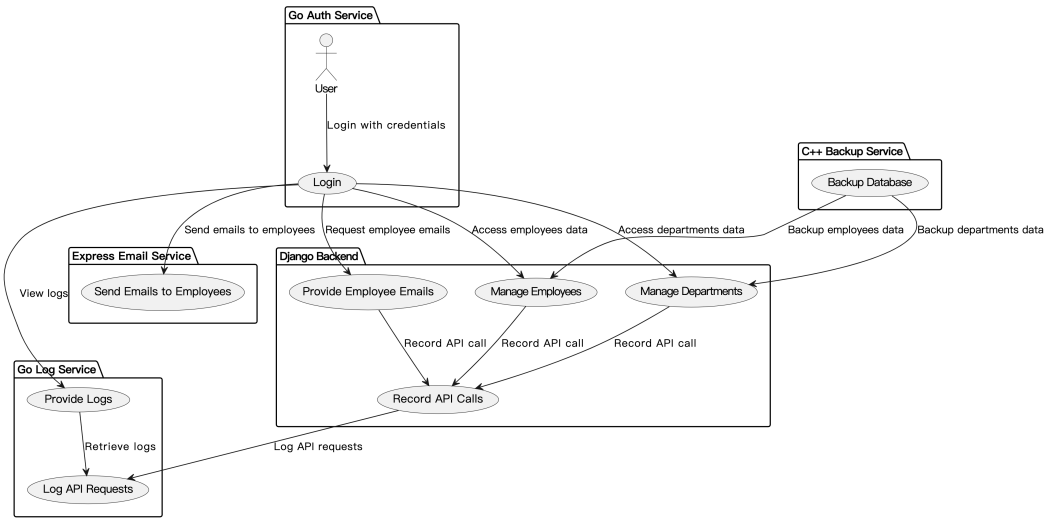

Figure 4: Usecase Diagram.

6 Technology Stack Summary

6.1 Back-end Technology Stack

- Go: Used for user login authentication and logging services.
- Django: Used for corporate employee management and API provision.
- C++ (Crow): Used for database backup services.
- Express.js: Used for email mass mailing services.

6.2 Front-end Technology Stack

- React: Used to build a single-page application.
- React Router: Used for routing management.
- Axios: Used for HTTP communication with the back-end.
- Semi Design: Used to beautify the user interface.

7 File Tree

```
- backend/
  - go-auth/ // Go authentication service
    - main.go
  - go-log/ // Go service and logging API operations service
    - main.go
  - python-django/ // Django backend service
    - manage.py
    - hrm/
      - models.py
      - views.py
      - urls.py
    - settings.py
  - cpp-backup/ // C++ backup service
    - main.cpp
```

```
    - CMakeLists.txt
  - express-email/ // Express.js mass email service and notifications
    - server.js
    - package.json
- frontend/
  - src/
    - components/
      - LoginComponent.js
      - EmployeeListComponent.js
      - LogComponent.js
      - EmailComponent.js
      - DatabaseBackupComponent.js
    - App.js
    - index.js
```