Database Design

The application uses a SQLite database to store employee information data. The specific database design is as follows:

1.The "staff_information" table contains basic information of employees, such as employee ID, age, attrition status, department, education level, job role, and marital status.

2.The "staff_privacy_information" table contains sensitive personal information of employees, such as business travel history, distance from home to work, number of employees in the company, environment and job satisfaction, work-life balance, and performance rating. This information requires strict confidentiality and management.

Front-end design

The application is designed using the Bootstrap front-end framework and includes the following pages:

1.index.html：For displaying the list of employees.

2.detail.html：For displaying employee details.

3.chat_detail.html：For displaying feedback results from chatbots.

Back-end design

The application is designed using Python's Flask framework and includes the following components:

1.Flask applications：Used to process HTTP requests and responses.

2.SQLite Database：Used to store employee information and employee privacy information data.

3.OpenAI Library：For natural language processing and generating intelligent chat logs.

Development

In developing the staff project, we accomplished the following tasks:

Write the front-end code: Designed the employee information display page and chatbot page using Bootstrap front-end framework. Implementing back-end logic: Implemented back-end logic using the Flask framework, including handling HTTP requests and responses, connecting to SQLite databases, and calling the OpenAI API for artificial intelligence interaction. Writing test cases: Used Behave to conduct automated testing and wrote test cases to test the functionality and performance of the application.

Implementation

When deploying the staff project, we accomplished the following tasks.

Installed dependencies: Installed and configured Python dependencies for Flask, Flask Paginate, Behave, Selenium, and Gunicorn using Pyenv and virtual environment management tools. Initialize the database: Initialized the SQLite database using the parse_csy.py script and imported employee information data and employee privacy information data. Start the application: Run the Flask application using Gunicorn, ensuring it runs continuously in the background. Access the application: Access the application URL in a web browser to view employee information and engage in intelligent conversations with the chatbot.

Unit Testing

The project uses Behave and Selenium for automated testing, which can be run using the following commands.

behave

The test cases are located under the features folder.