Abhishek Aryal - Technical Screen Submission

Link to repository

https://github.com/ary57/uiowaTechnicalScreening/tree/main

The README.md on the repository includes the series of steps to run the application.

Estimated and Actual Hours

(See TimeTracking.md for more details.)

Estimated Hours

- Frontend 4 hours.
- Backend: 4 hours
- Git, writeup + misc: 1 hour
- Total: 9 hour.

Actual Hours

- Frontend 4 hours.
  - Initial design - 1 hour
  - Development - 3 hours
- Backend - 2.5 hours.
  - API - 1.5 hours
  - Database - 1 hour
- Misc - 1hr
  - git, writeups, etc.
- Total: 7.5 hours

Reason for choosing this tech stack

Frontend: React, Bootstrap, CSS, React-Bootstrap

Backend: Python, Flask, SQLite3

In my current role, I am working with my team on migration from a traditional static HTML, Bootstrap, and JavaScript website to a more modern architecture. These frontend technologies form the basis of our selected stack, and I work with them extensively on a daily basis. Among the available options, I am most proficient with this set of tools.

Our web application also uses Python with Flask for API development. Similar to the frontend, I rely on these technologies every day, which made them a natural choice for this project as well.

For the database, I selected SQLite based on my previous experience using it in Python. Given the relatively limited scope of the application, I found it to be a practical and efficient solution.

I understand that the position emphasizes experience with .NET and Angular. While I have not yet used these frameworks in a professional setting, I am eager to expand my skill set. My background with related technologies provides a solid foundation that would support a smooth and efficient transition to these tools.

Assumptions, Problems, Highlights/Additions

Assumptions

- Security: I assumed responsibility for the security of the SQL query by ensuring it was properly parameterized when inserting data into the database. This approach helps protect against SQL injection and aligns with best practices for database interactions.
- User Interface: Since the application is part of the University of Iowa's website, I referred to the university's brand manual to ensure consistency with its visual identity. This included using the correct font, color palette, and official logo.
    - Website: https://brand.uiowa.edu/

Highlights / Personal Additions

- HawkID Field: I included a HawkID field as a way to uniquely identify each employee when submitting reimbursement requests. This ensures accurate record-keeping and traceability.
- Disabled Submit Button: To create a more intuitive user experience, I implemented logic to disable the submit button until all required fields are completed. This helps guide users through the form and prevents incomplete submissions.
- Local Storage of Receipts: When a user submits a receipt, the application stores the image or PDF in two places: as a blob in the database and locally in the server/receipts folder.
    - This feature was initially added to verify that files were being correctly transmitted from the frontend to the backend, but I decided to retain it for submission purposes.
    - The receipt naming convention follows this format: `hawkID` + `date` + `amount` + `uuid`, where the UUID ensures filename uniqueness and prevents any potential overwrites.

Problems

- File Transfer Between Client and Server: Prior to this project, I had not worked with transferring files between the client and server. To get started, I referred to a few online posts / articles that provided an example for sending files from a React frontend to a Flask backend. This served as a useful starting point for my experimentation and implementation.
    - https://stackoverflow.com/questions/75289192/error-sending-pdf-from-reactjs-to-flask-app
    - https://medium.com/@mp.hladun/how-to-upload-a-file-from-react-to-flask-89cafa000ba1

Comments

- Initial Design: To establish a starting point for the application, I used Figma to create an initial design layout. This was as a helpful reference during the early

stages of development. While the final implementation evolved from the original mockup, it provided a solid foundation to build from.

- The initial concept is available as an image in the `concept` folder.