Software Engineering Project Presentation: **Support Ticketing System**

Synergy_4APD: Adhil Ahmed, Afnan Ahmad, Preetodeep Dev

About Us - Team Synergy_4APD

- **Adhil Ahmed** is a student from Vellore, Tamil Nadu. He is a final year Automobile Engineering student at Anna University, MIT Campus. He has been fascinated by technology since 8th grade. In his free time, he like to read thrillers, go for a bicycle ride, and watch Formula 1 (#MV33).

- **Afnan Ahmad** is pursuing BCA at IGNOU, New Delhi alongside the BS in Data Science and Applications from IITM. His interests include algorithms and artificial intelligence.

- **Preetodeep Dev** is pursuing B.Tech (CSE) at Assam University alongside this BS in Data Science and Applications from IITM. He is currently in awe of the work this team as put in collectively! :)

Support Ticketing System - Core Project Requirements

- User management - registration, login, logout, promotion, demotion, etc.
- Ticket management - creation, updation (addition of messages), resolution, deletion, etc.
- Article/FAQ management - creation, updation (addition of comments), deletion, viewing, etc.
- Notification and escalation system.
- Other requirements as per the uploaded user stories

Technologies Used - Development

- As was asked in the problem statement, the entire project was built from the ground-up while using version control software (git).
- Bash was used extensively for version control using git, and also for installing the necessary packages (via pip, or otherwise).

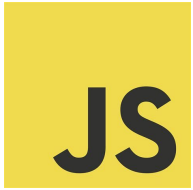
- Visual Studio Code was the primary IDE used for developing the app. It is quite powerful and the extensions therein were very useful.
- We've also used tools such as Swagger OAS editor, Trello, Postman, Google Workspace apps, etc.

- Mozilla Firefox, Apple Safari, and Google Chrome were the browsers used throughout the development process
- Browser features such as developer console and extensions were used extensively

- DB Browser for SQLite was used for inserting, deleting, and editing the data in the SQLite database at the backend during the development and testing process

Technologies Used - Frontend

- The frontend was built using Flask - Jinja2 templating
- Features of Jinja2 such as dynamic webpage generation, extensions, template inheritance, etc. have been used extensively.



- Bootstrap was used throughout the entire app, for styling purposes
- Various Bootstrap elements, such as buttons, tables, cards, etc. have been used

Technologies Used - Backend

- Python-based frameworks were used to develop the backend
- Various Python libraries, such as uuid, datetime, humanize, etc. were used

- As mentioned in the problem statement, the Flask web framework was used for the backend server
- Various Flask sub-components, such as Flask-Security-Too (for the authentication mechanisms), Flask-RESTful (for the API), Flask-CORS (for enabling cross-origin requests), etc. have been used

Flask-Security

- SQLite was used as the database for storing the app data at the backend
- Various tables corresponding to the users, tickets, articles (FAQs), etc. were created

- Flask-SQLAlchemy was used as the ORM to enable the interactions between the app at the backend Flask API and the SQLite database
- The various data models were made using the appropriate classes in SQLAlchemy

Application Working Demonstration - Some Typical Process Flows

Workflow Demonstration Done In The Video

Challenges We Faced - Lessons for the Future

- **Selecting the appropriate data structure for the application given the various contexts** was a major chunk of the debugging/testing that had to be done. Often, we've seen that it is easier to store data in a particular structure, but easier to read from another structure. **The tradeoff has to be thought about carefully**.
- **Strict time deadlines:** The strict milestone deadlines meant that we were mainly hopping from milestone deadline to milestone deadline, without actually getting much time to think about the entire project in an organic manner.
- **Initial coordination** and getting into a proper workflow was slightly more difficult than we initially thought.
- **Prior unfamiliarity with git in practical usage** was a major hurdle that we had to overcome. We always found ourselves hesitating while trying to resolve merge conflicts and approve pull requests.

Beyond the Project - Next Steps

- **Addition of the email and chat/SMS notification** features, over and above the notifications implemented in the API/database
- **Automatic support staff assignment** to tickets with large number of upvotes or to tickets facing starvation
- **Ticket merging** option for similar tickets, accessible to support staff
- **Moderation controls** for support staff and administrators
- **Student-initiated reminder/alert** in case their ticket hasn’t been responded to in a long time

- Introduce **localization features** and allow users to set their locales
- Implementation of a **VueJS frontend**. The app should ideally be an SPA.
- Creation of a **mobile app based on the existing API**
- Implement a **CI/CD workflow**
- A **more comprehensive test suite**
- **Caching within the overall stack** - the app should perform well even at scale
- Improve **accessibility features** of the app, such as screen-reader support

Acknowledgements and Thanks

1. **Course faculties**: Prof. Shridhar Iyer and Dr. Prajish Prasad, for their excellent lecture videos and their effective teaching of the software engineering course.
2. **Course instructors**: Mr. Arup Chattopadhyay, Mr. Ankur Parmar, Mr. Abhishek Rajput; for the numerous live sessions they conducted explaining the requirements of the project and their constant guidance and support.
3. **As a team, all three of us thank each other.** We feel that as a team, we have all been able to work together quite effectively and with great cohesion. We've covered for each other and were able to have a much better outcome collectively.
4. The internet at large: **StackOverflow; Mozilla Developer Network; documentation of Python, Flask, Jinja2, etc.**
5. **Linus Torvalds, Richard Stallman, and Guido van Rossum**; for obvious reasons! :)

Thank you!