# IT5016. Software Development Fundamentals Assessment 2. Software Project Dhruvraj Desai - 20231815

# PLANNING:

1. Objectives Defined:

Embarked on the journey to develop a dynamic ticket management system, aimed at streamlining assistance requests within the organization's workflow.

1. Stakeholder Identification:

Recognized the importance of engaging stakeholders early on, ensuring their needs and perspectives were integrated into the project's foundation.

1. Resource Allocation:

Gathered an arsenal of resources, including support from peers and tutors, to fuel the project's inception and ensure a robust start.

1. Goal Establishment:

Set crystal-clear objectives and milestones, akin to charting a roadmap, to guide the project's trajectory and measure progress effectively.

1. Project Management Familiarization:

I delved into the realm of project management principles, equipping myself with tools to orchestrate a seamless execution of the project's vision.

# REQUIREMENT ANALYSIS:

1. Stakeholder Engagement:

Engaged in rich dialogues with stakeholders, unraveling their diverse needs and aspirations, and laying the groundwork for a user-centric solution.

1. Needs Prioritization:

Delved deep into the intricacies of user needs, meticulously prioritizing them to ensure that the system resonated with the core requirements of its users.

1. Collaboration and Guidance:

Forged alliances with classmates and mentors, fostering a collaborative environment that nurtured innovative solutions and enriched perspectives.

1. Task Translation:

Transformed stakeholder requirements into actionable tasks, weaving a coherent narrative that would drive the system's development journey forward.

1. User Story Definition:

Crafted vivid user stories and use cases, painting a vivid picture of the system's functionality and user interactions, guiding its development trajectory.

# DESIGN STAGE:

1. Architecture Design:

Architected a robust and scalable system infrastructure, laying a sturdy foundation for seamless expansion and enhanced performance.

1. User Interface Conceptualization:

Envisioned an intuitive and visually appealing user interface, where user experience was paramount, ensuring ease of navigation and interaction.

1. Security and Database Considerations:

Fortified the system's defenses with state-of-the-art security measures, while meticulously crafting a database architecture that upheld data integrity and accessibility.

1. Resource Utilization:

Tapped into a treasure trove of online resources and peer wisdom, leveraging them to fuel the creative process and surmount design challenges.

1. Alignment with Objectives:

Ensured unwavering alignment of the design with project goals and stakeholder aspirations, forging ahead with purpose and conviction.

# DEVELOPMENT STAGE:

1. Coding Challenges:

Confronted coding challenges head-on, each line of code a testament to perseverance and a step forward in the coding journey.

1. Debugging and Troubleshooting:

Embraced the debugging process as an opportunity for growth, navigating through challenges with resilience and determination.

1. Peer and Tutor Support:

Found solace and guidance in the camaraderie of peers and the wisdom of tutors, their support serving as a beacon in moments of uncertainty.

1. Incremental Progress:

Celebrated each milestone achieved, recognizing that every line of code written was a brick in the grand edifice of the system's development.

1. Skill Expansion:

Embraced the development stage as a crucible for skill refinement, emerging as a more adept and confident coder, primed for the challenges ahead.

# SDLC (Software Development Life Cycle):



1. Planning:

We mapped out our journey, setting clear goals and gathering the necessary resources for our adventure. Like planning a road trip, we charted our course with defined objectives and strategies to engage stakeholders along the way.

1. Requirement Analysis:

We put on our detective hats and dove deep into understanding what our users truly needed. Just like solving a mystery, we translated these needs into actionable tasks and stories to guide our development path.

1. Design:

With a vision in mind, we sketched out the blueprint for our creation. It was like designing a dream home, ensuring every aspect from architecture to user interface aligned perfectly with our goals and aspirations.

1. Development:

Armed with our plans, we set to work building our masterpiece. Each line of code was like adding a brick to our castle, overcoming challenges with the help of our peers and mentors along the way.

1. Testing:

Before unveiling our creation to the world, we meticulously checked every nook and cranny for flaws. It was like polishing a gem, ensuring our system sparkled with perfection before it reached the hands of our users.

1. Deployment:

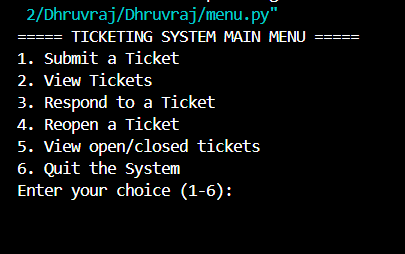
Finally, it was time to launch our creation into the world. Like releasing a bird from its cage, we set our system free, ensuring a smooth transition from development to deployment and keeping a watchful eye to ensure its success in the wild.

1. Maintenance:

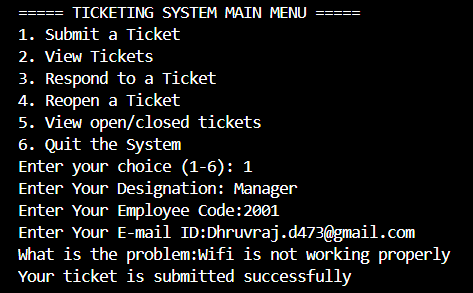
Embraced the responsibility of post-deployment maintenance, addressing user feedback, implementing updates, and ensuring continuous improvement and adaptability.

# TESTING:

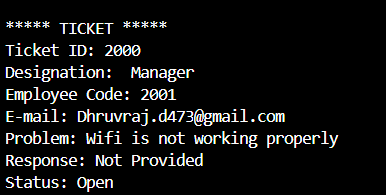
1. Display Main Menu: User have to choose the given options to create the tickets.



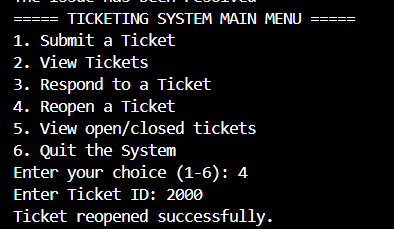
1. Creating Ticket: There are the details that user have to fill while creating the new tickets.



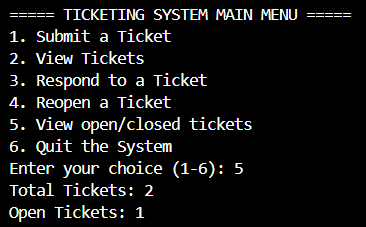
1. View Ticket: Now, If the user wants to see the ticket that they have submitted they can select the options View Ticket.



1. Reopen Ticket: Once the problem is solved and in future if it’s happen again then the user don’t have to submit the ticket again. They just have to reopen the ticket by entering the ticket Id that they have used before.



1. View open/closed Tickets: This will show how many tickets are open or closed in the system.



# Explanation of Code:

1. ticket.py:

This file contains the definition of the `Ticket` class.

- The `Ticket` class represents a ticket for assistance requests in the ticketing system.

- It has attributes such as `ticket\_id`, `designation`, `employee\_code`, `email\_address`, `problem`, `response`, and `status`.

- The constructor `\_\_init\_\_` initializes a new Ticket object with provided attributes. It also sets default values for `response` and `status`.

- The `process\_problem` method checks if the problem description contains certain keywords (in this case, "change" and "password"). If it does, it generates a new password for the employee based on their employee code and designation, and updates the response accordingly.

- The `generate\_new\_password` method generates a new password based on the employee's code and designation.

2. system.py:

This file contains the definition of the `TicketingSystem` class, which manages the tickets in the system. Here's what it does:

- The `TicketingSystem` class initializes with empty lists to store tickets and counts for open and total tickets.

- It has methods like `submit\_ticket`, `display\_tickets`, `response\_ticket`, `reopen\_ticket`, and `status` to perform various operations on tickets.

- `submit\_ticket` method allows users to submit a new ticket by providing necessary details like designation, employee code, email, and problem description. It then creates a new `Ticket` object and adds it to the list of tickets.

- `display\_tickets` method displays all the tickets along with their details.

- `response\_ticket` method allows users to respond to a ticket by providing a response. It updates the ticket's status to "Closed" and decreases the count of open tickets.

- `reopen\_ticket` method allows users to reopen a closed ticket. It updates the ticket's status to "Reopened" and increases the count of open tickets.

- `status` method displays the total number of tickets and the number of open tickets.

3. menu.py:

This file contains the main menu and the entry point of the program. It imports the `Ticket` class from `ticket.py` and the `TicketingSystem` class from `system.py`, and provides a menu-driven interface to interact with the ticketing system. Depending on the user's choice, it calls the corresponding method of the `TicketingSystem` object.

Overall, these files together form a simple ticketing system where users can submit tickets, view tickets, respond to tickets, reopen closed tickets, and view the status of tickets.

# References:

(Greeksforgreeks, n.d.)

# References

*Greeksforgreeks*. (n.d.). Retrieved from Greeksforgreeks: https://www.geeksforgeeks.org/software-development-life-cycle-sdlc/