Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre

A Project Report
on
“SolveIt”

[COMP 206]

(For partial fulfillment of 2nd Year/1st Semester in Computer Engineering)

Submitted by

Aryan Mahato(30)
AryanThagunna (53)
Pukar Adhikari (03)
Sakar Maharjan(29)

Submitted to

Mr. Suman Shrestha

Department of Computer Science and Engineering

Submission Date:5thAugust,2025

Bona fide Certificate

This project work on

“SolveIt”

is the bona fide work of

“Aryan Mahato,Aryan Thagunna,Pukar Adhikari,Sakar Maharjan”

who carried out the project work under my supervision.

Project Supervisor

____

Dr. Sushil Shrestha
Associate Professor
Department of Computer Science and Engineering

Date:

Acknowledgement

We sincerely thank everyone who contributed to the successful development and completion of our SolveIt website. This project has been a collaborative effort, and we are deeply appreciative of all the support we received throughout this journey.

First and foremost, we would like to acknowledge our supervisor, Dr. Sushil Shrestha, for his continual support, guidance, and invaluable insights during the project. His expertise and encouragement were instrumental in helping us navigate challenges and make informed decisions throughout the development process.

We also extend our heartfelt thanks to our colleagues and peers for their collaboration, idea sharing, technical troubleshooting, and emotional support. Additionally, we are grateful to our family and friends for their unwavering motivation and encouragement, which helped us stay focused and dedicated. This project would not have been possible without the collective effort and support of all these individuals, and we truly value their contributions..

Abstract

SolveIt is a web-based Q&A platform developed specifically for students of Kathmandu University to foster academic collaboration and peer-to-peer learning. Inspired by platforms like Quora, Reddit, and Stack Overflow, SolveIt provides a dedicated space where students can ask questions, share knowledge, and engage in meaningful discussions. The platform features personalized feeds, tagging, and file upload capabilities to support detailed academic queries. It is built using ReactJS for the frontend with Tailwind CSS for styling, and Flask with SQLite on the backend. User registration is required to participate, ensuring a secure and focused learning environment. Although the platform is fully developed, user testing and feedback collection are ongoing to evaluate performance and improve usability. SolveIt demonstrates strong potential as a supportive academic tool tailored to the needs of Kathmandu University students.

Keywords: ReactJS, Tailwind CSS, Flask, SQLite, Q&A Platform

Table of Contents

List of Figure

Acronyms/Abbreviations

CSS: Cascading Style Sheets, a framework used for styling the frontend.

GUI: Graphical User Interface, a part of the project planning phase.

HDD: Hard Disk Drive, a recommended hardware specification.

OTP: One-time Password, a method used for user authentication.

PDF: Portable Document Format, a file type that can be uploaded to the platform.

Q&A: Question and Answer, a term used to describe the platform's primary function.

SPA: Single-Page Application,

SSD: Solid-State Drive, a recommended hardware specification.

Chapter 1 Introduction

1.1 Background

With the rapid growth of online learning and digital education, platforms that facilitate knowledge sharing and collaborative problem-solving have become increasingly important (Anderson et al., 2013). Over the past decade, educational communities such as Stack Overflow, Quora, and Reddit have significantly influenced how users seek and share information. These platforms have introduced features like voting systems, tagging, and user reputation tracking to improve content quality and engagement (Mamykina et al., 2011; Tausczik & Pennebaker, 2011). In parallel, advancements in web technologies have enabled more interactive, responsive, and user-friendly applications, often built with frameworks like React and Flask (Grinberg, 2018).

Despite these developments, existing platforms face several limitations. Many are not tailored specifically for academic or educational institutions, making them overwhelming for students due to the presence of unrelated or non-academic content. File sharing, which is critical for students to exchange diagrams, notes, or code files, is often either unsupported or poorly implemented. Moreover, most popular platforms lack mechanisms to verify user authenticity, which can result in spam, misinformation, or irrelevant answers (Zhang et al., 2017). These issues highlight a gap in the current ecosystem for a dedicated, moderated, and feature-rich platform focused solely on educational problem-solving.

1.2 Objectives

The objective of this project are:

- To develop a web-based Q&A platform tailored to the academic needs of Kathmandu University students.

- To enable question and answer submissions with file attachments for effective academic support (e.g., code, diagrams, PDFs).
- To foster a community-driven environment through features like voting and tagging to improve the relevance and quality of content.

1.3 Motivation and Significance

As university students, we often encountered situations where we had doubts related to course content, assignments, or exam preparation, but lacked a centralized, academic-focused platform to seek and receive help from peers. Existing solutions like messaging apps and social media groups are typically disorganized, lack proper categorization, and fail to preserve academic discussions for future use. Platforms such as Google Classroom [classroom.google.com], while useful for content distribution, are instructor-driven and do not encourage open, peer-to-peer problem-solving. This gap in accessible, student-centered academic support motivated us to build SolveIt, a platform designed specifically for students to ask course-specific questions, get reliable answers from peers, and collaborate in a structured academic environment.

SolveIt addresses key limitations of current communication tools by introducing a dedicated Q&A platform tailored for university-level academic interaction. Unlike general platforms like Quora or Reddit, SolveIt focuses on course-based tagging, structured discussions, and features such as file uploads (e.g., PDFs, images, notes), OTP-based user verification, and a minimal, learning-oriented interface. The platform enables upvoting to highlight helpful answers and builds a searchable repository of academic content. By promoting faster doubt resolution and encouraging peer learning, SolveIt enhances the overall educational experience and supports a culture of collaborative learning within the university community.

Chapter 2 Related Works

Several online platforms have been developed to support collaborative learning, peer assistance, and knowledge sharing among students. Among them, Piazza is a classroom-oriented question-and-answer tool widely adopted in academic institutions. It allows students to post course-related questions and receive responses from instructors or peers, while supporting anonymity and threaded discussions. However, its use is typically restricted to specific courses and institutional logins, limiting accessibility and hindering long-term knowledge retention outside the classroom environment (Anderson, Huttenlocher, Kleinberg, & Leskovec, 2013).

Google Classroom is another educational platform primarily designed for managing assignments and facilitating communication between students and teachers. While useful for administrative tasks, it lacks a focus on peer-to-peer academic problem-solving and does not offer public Q&A, topic tagging, or rich file-sharing capabilities needed for collaborative learning (Johnson & Lee, 2020).

General-purpose Q&A platforms such as Quora and Reddit also allow users to post academic questions. However, these platforms are often too broad, not tailored to university-specific curriculum, and lack contextual accuracy in responses. Moreover, they do not provide structured features such as file uploads, academic verification, or subject-based tagging that are essential for educational environments (Martinez & Nguyen, 2018).

To address these limitations, SolveIt is designed as a focused academic Q&A platform for university students. It allows verified users to ask course-specific questions, apply subject-based tags, and share relevant files like notes or diagrams. Unlike classroom-restricted platforms, SolveIt creates an open yet structured academic space that encourages cross-departmental and inter-semester collaboration. By facilitating reusable, searchable, and community-driven discussions, SolveIt aims to fill the gap left by existing platforms and promote a culture of continuous academic engagement.

Chapter 3 Design and Implementation

The development of SolveIt, our university-specific Q&A platform, was driven by a combination of technical requirements, user-focused functionality, and practical development constraints. Our final tech stack consisted of:

- **Frontend**: React.js
- **Backend**: Flask (Python)
- **Database**: SQLite

Why We Chose This Tech Stack?

Initially, as suggested by our supervisor, we considered building the platform using C++ and the Qt framework. However, this approach quickly proved unfeasible due to several limitations:

- **Web Deployment Challenges**: Qt is primarily designed for desktop applications and does not natively support web-based deployment, which was a core requirement for our platform.
- **Lack of Web-Friendly Features**: C++ lacks built-in support for critical web development functionalities such as routing, HTTP request handling, templating, and user session management.
- **Time and Learning Constraints**: Implementing a web platform in C++ would require significant time to either write low-level networking code from scratch or integrate complex third-party libraries.

In contrast, the Flask–React–SQLite stack offered a significantly smoother development experience:

- **Flask** provides a lightweight and flexible backend framework with built-in routing and easy database handling, ideal for rapid prototyping and REST API creation.
- **React** allows us to build a dynamic, component-based frontend with responsive design, ensuring a smooth user experience across devices.

- **SQLite**, being serverless and lightweight, was a perfect choice for quick local development and a small-scale user base like ours.

By leveraging this stack, we were able to focus more on user functionality and interface design rather than spending excessive effort on setting up core infrastructure.

After settling on the tech stack, we first invested time in learning the tools. Development followed a modular approach, where we tackled core components one at a time. Our design was centered around simplicity and user accessibility:

- A top navigation bar featuring quick actions like "Ask Question"
- A search bar for fast content discovery
- A personalized feed showing questions relevant to the user’s selected tags
- A sidebar for intuitive navigation to sections like Home, Questions, About Us, and Chat
- A profile system for managing user details and interests

While the shift from C++ to a full web stack required a steep learning curve, it ultimately allowed us to build a reliable and functional platform.

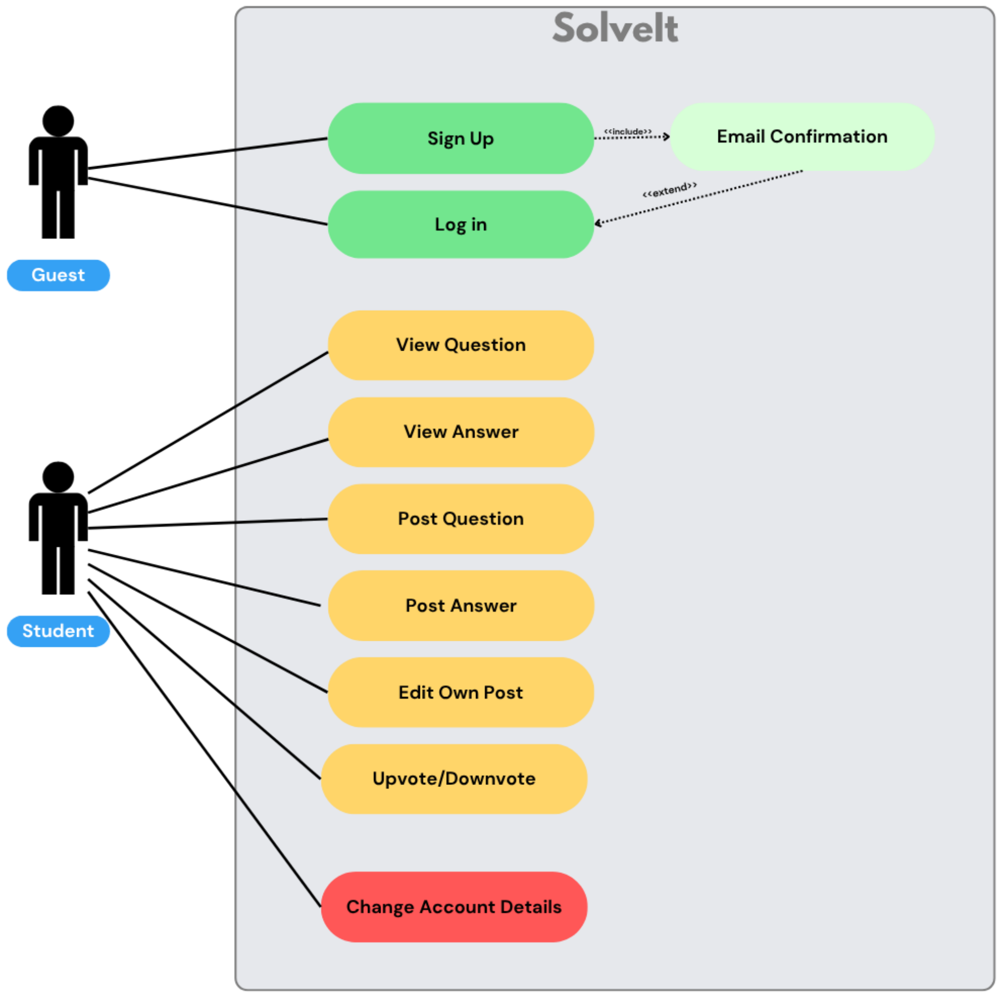


Figure 1: Use Case Diagram

3.1 System Requirement Specifications

3.1.1 Software Specifications

Frontend Technologies

- **React.js**
  React.js is a JavaScript library used to build fast and interactive user interfaces. Its component-based structure allowed us to modularize UI elements like the

question feed, navigation bar, and profile pages. This structure improved code reusability and helped maintain consistent design throughout the app.

- **Tailwind CSS**
  Tailwind CSS is a utility-first CSS framework that allows us to design directly within our React components using pre-defined classes. This eliminated the need for separate CSS files and accelerated our styling process. It ensured visual consistency across components like buttons, cards, and forms.

Backend Technologies

- **Flask**
  Flask is a micro web framework written in Python that provides the core server-side functionality for SolveIt. We used it to:
  - Handle HTTP requests from the React frontend
  - Define API routes for actions like asking questions, submitting answers, and retrieving user feeds
  - Manage user authentication, including login, registration, and OTP-based verification
  - Support file uploads for additional study resources
  - Its lightweight nature, flexibility, and minimal boilerplate made it ideal for a student-level project with limited time and resources.
- **SQLite**
  SQLite is a lightweight, serverless SQL database that stores data in a single local file. We used it to manage structured application data such as:
  - User accounts and credentials
  - Questions and their associated tags
  - Answers, timestamps, and user interactions
  - SQLite was chosen for its zero-configuration setup and seamless integration with Flask via SQLAlchemy, allowing us to efficiently test and develop without setting up a full database server.

3.1.2 Hardware Specifications

To develop and run SolveIt, given below are the minimum and recommended hardware specifications:

Minimum Hardware Requirements ***(suitable for development and basic local testing)***

- **Processor:**
  - Dual-core CPU (e.g., Intel i3 or AMD Ryzen 3)
  - Adequate for running the Flask backend and React development server simultaneously during basic testing.
- **Memory (RAM):**
  - 4 GB
  - Sufficient for lightweight development, though limited for multitasking (e.g., running browser, code editor, and server at once).
- **Storage:**
  - 5 GB free HDD/SSD space
  - Enough for source code, dependencies (Node modules, Python packages), database files, and uploaded content during testing.

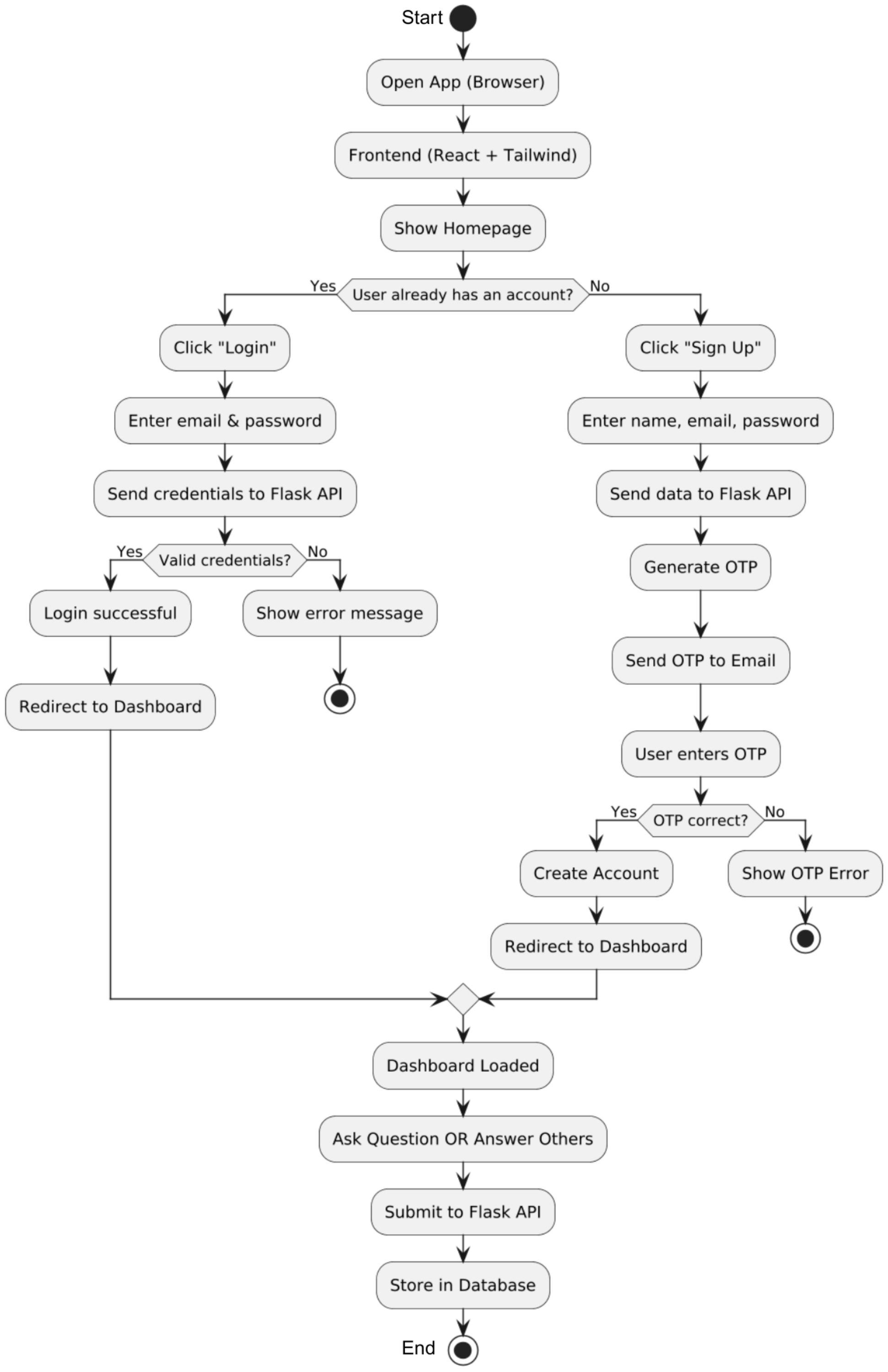


Figure 2: Flowchart

Chapter 4 Discussion on the achievements

The development of SolveIt has been a journey defined by rapid adaptation, collaboration, and meaningful achievement. From its inception, the project aimed to localize the core ideas behind platforms like Stack Overflow, Piazza, and Brainly bringing course-specific, academic peer support to the context of our university. While these platforms offer powerful solutions, they often lack institution-specific focus, filtering, or course tagging tailored to individual curricula. SolveIt was developed to bridge that gap.

Initially, we attempted to build the platform using C++ with the Qt framework, but this approach lacked the flexibility needed for web deployment and modern UI design. Recognizing the challenges ahead, we transitioned to a more suitable and scalable stack: ReactJS for the frontend, Tailwind CSS for responsive design, and Flask with SQLite for backend functionality and data persistence.

This transition required us to learn these technologies from scratch. While platforms like Stack Overflow and Piazza are developed by experienced engineering teams over several years, we were able to build a working prototype in a matter of weeks despite having no prior experience with React or Flask.

Functionally, we implemented key features found in leading Q&A platforms:

- An “Ask Question” interface, accessible from the navigation bar similar to Stack Overflow's prominent posting system.
- A personalized feed based on tags and course preferences, echoing the customization of Brainly and Piazza's course-specific threads.
- User profiles where students can manage their information and select tags based on subjects or university courses.

- A clean sidebar navigation with intuitive access to main sections like Home, Questions, About Us, and Chat, designed for ease of use, especially on mobile devices.

Unlike larger platforms that serve general or global audiences, SolveIt focuses narrowly on the academic needs of students within our university. This local focus enabled us to include custom features like:

- University course-specific tagging
- Lightweight architecture that supports local hosting or campus-wide deployment

While SolveIt may not yet match the feature depth or scalability of global platforms, our achievement lies in creating a functional, university-tailored platform under tight time constraints, limited resources, and a steep technical learning curve.

4.1 Features:

- **Ask Question section** in the navigation bar
- **Personalized newsfeed** displaying questions based on user-selected tags
- **User profile pages** with tag preferences and basic information
- **Upvote and downvote functionality** to promote helpful questions and answers
- **Sidebar navigation** including links to:Home, Questions, About Us, Chat

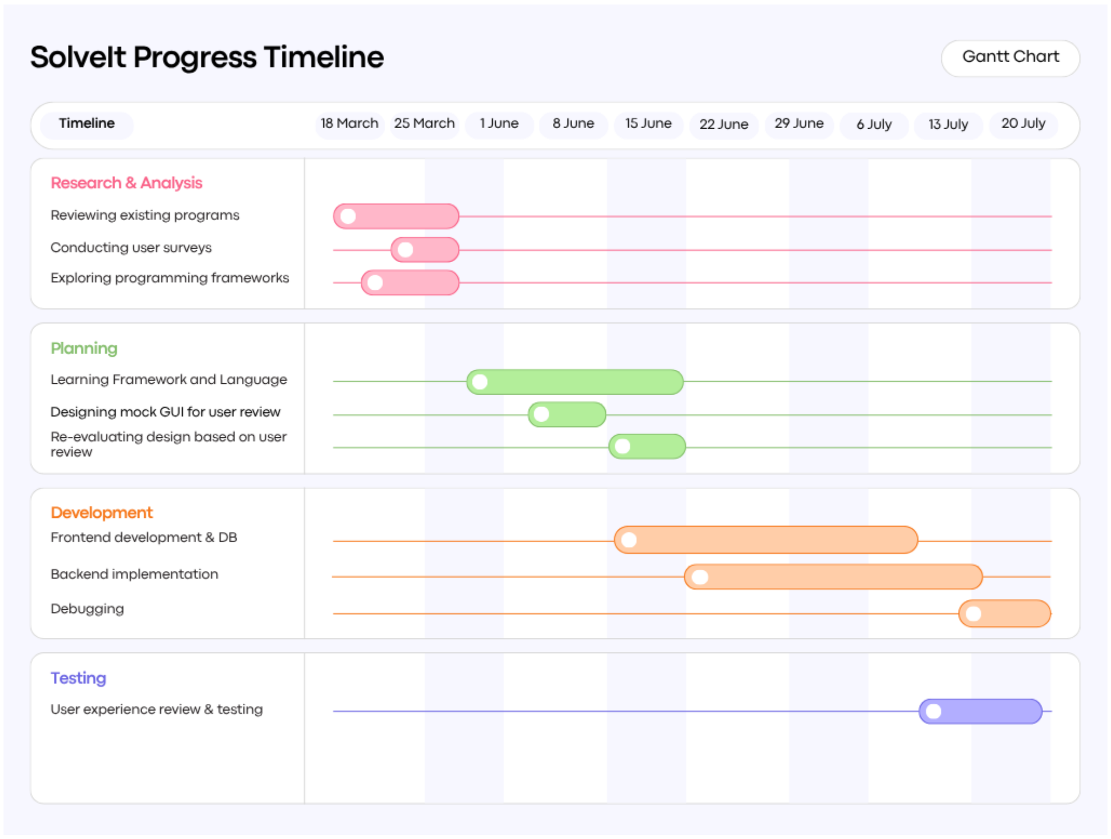


Figure 3: Gantt Chart

Chapter 5 Conclusion and Recommendation

The development of SolveIt represents a focused attempt to localize the collaborative knowledge-sharing model of global platforms like Stack Overflow and Brainly to the academic environment of our university. Our primary goal was to build a functional, student-centric Q&A platform that enables subject-specific discussions, content voting, and file sharing among peers. These core features were successfully implemented and are fully operational in the current version of the platform.

SolveIt allows students to:

- Post and answer course-related questions
- Upvote and downvote contributions based on helpfulness
- Filter content using subject tags
- Upload relevant documents and resources

These capabilities promote peer-to-peer learning and help foster a more collaborative academic community within our institution.

While several advanced features such as a real-time chat or messaging remain pending, this was a deliberate choice. Our focus remained on delivering a stable and meaningful user experience within the time constraints of an academic semester. These omitted features represent clear opportunities for future expansion and enhancement of the platform.

5.1 Limitations

- Restricted User Base: Currently accessible only to students of Kathmandu University, limiting its reach and impact.
    - Designed exclusively for Kathmandu University students.

    - It confines the platform's potential audience, reducing opportunities for broader engagement and influence.
- Lack of Direct Communication Tools: The absence of chat and messaging restricts user engagement to public threads only.
    - Why are chat and messaging features important? Because they facilitate direct, private, and instant communication between users.
    - How does their absence affect user engagement? It limits interactions to only public threads, which can reduce the depth and immediacy of user conversations.
- No Global Search Functionality: Without a search bar, users must manually browse topics, which may hinder the discoverability of content.
    - Why is a search feature necessary? Because it allows users to quickly find relevant content and navigate the platform efficiently.
    - How does not having a search bar hinder content discoverability? Users have to spend more time scrolling and browsing, which can discourage exploration and reduce the visibility of valuable content.

5.2 Future Enhancement

- Implement a Global Search Bar: Enabling users to search questions, tags, and users efficiently.
- Introduce Chat and Messaging Features: Allowing users to communicate directly and collaborate in real-time.
- Mobile App Version: A mobile-friendly version can further increase accessibility and ease of use.

References

SAnderson, A., Huttenlocher, D., Kleinberg, J., & Leskovec, J. (2013). *Steering user behavior with badges*. Proceedings of the 22nd International Conference on World Wide Web. https://doi.org/10.1145/2488388.2488398

Grinberg, M. (2018). *Flask Web Development: Developing Web Applications with Python*. O'Reilly Media, Inc.

Mamykina, L., Manoim, B., Mittal, M., Hripcsak, G., & Hartmann, B. (2011). *Design Lessons from the Fastest Q&A Site in the West*. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. https://doi.org/10.1145/1978942.1979095

Tausczik, Y. R., & Pennebaker, J. W. (2011). *Predicting the perceived quality of online mathematics contributions from users' reputations*. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. https://doi.org/10.1145/1978942.1979063

Stack Overflow. (n.d.). *About Stack Overflow*. https://stackoverflow.com/company

Flask. (n.d.). *Flask Documentation (2.x)*. https://flask.palletsprojects.com/

SQLAlchemy. (n.d.). *The Database Toolkit for Python*. https://www.sqlalchemy.org/

APPENDIX

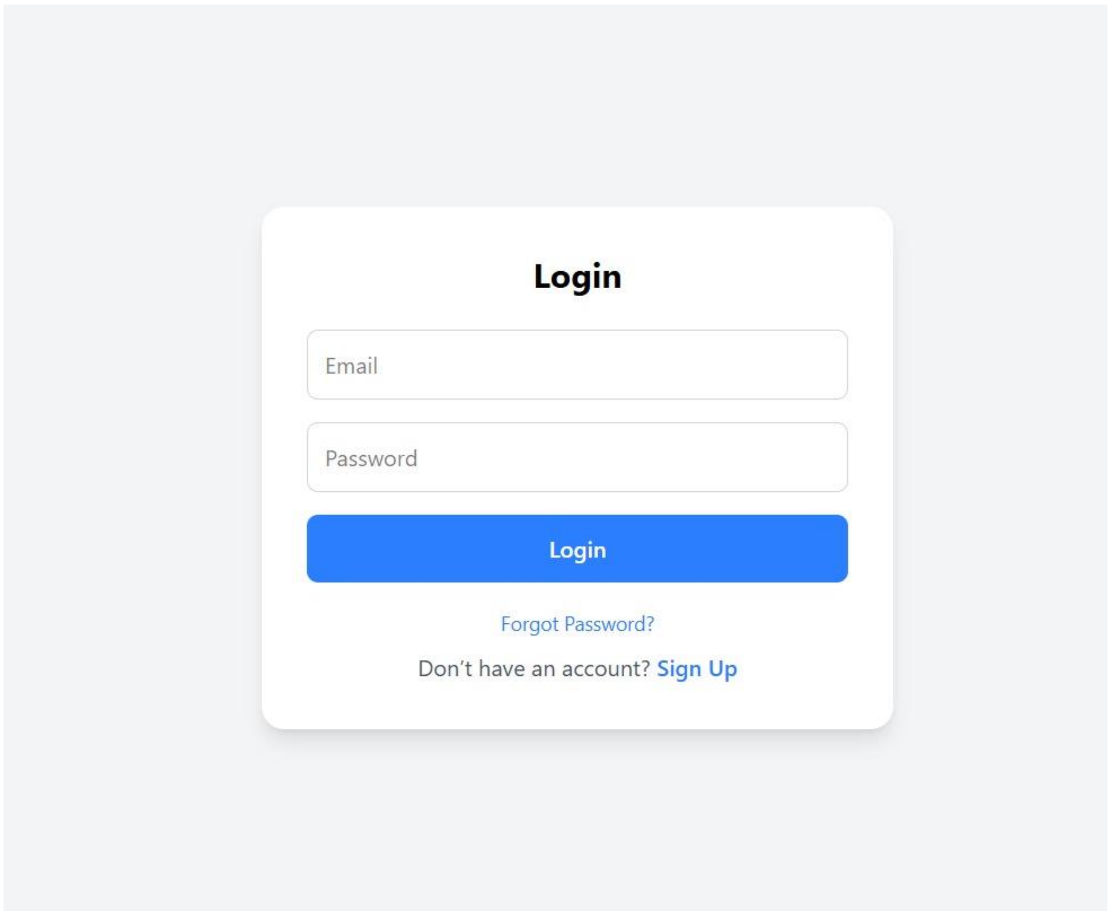
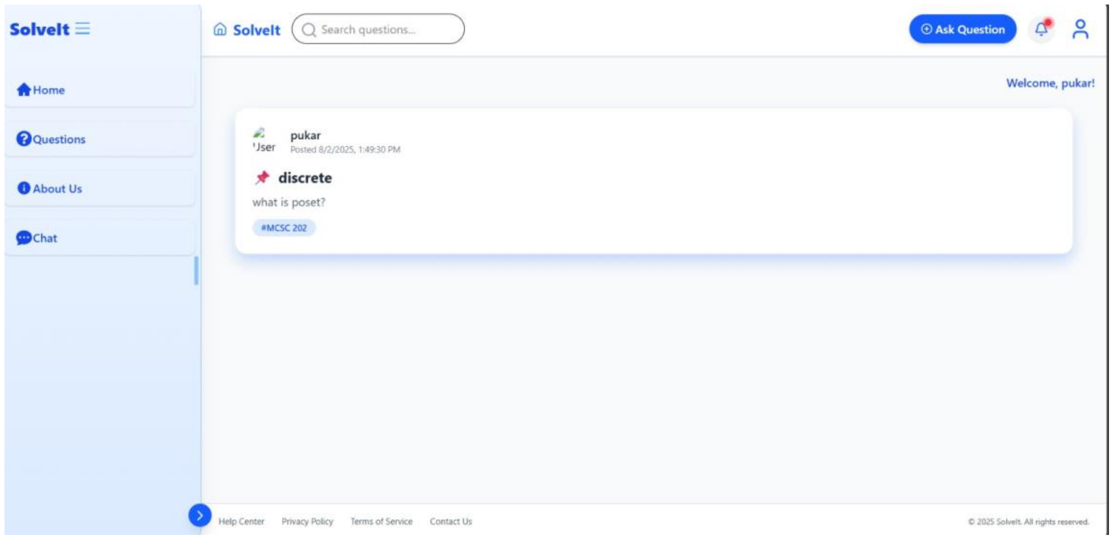


Figure 4: Login Page



pukar
Posted 8/2/2025, 1:49:30 PM

discrete

what is poset?

#MCSC 202

Figure 5: Home Page

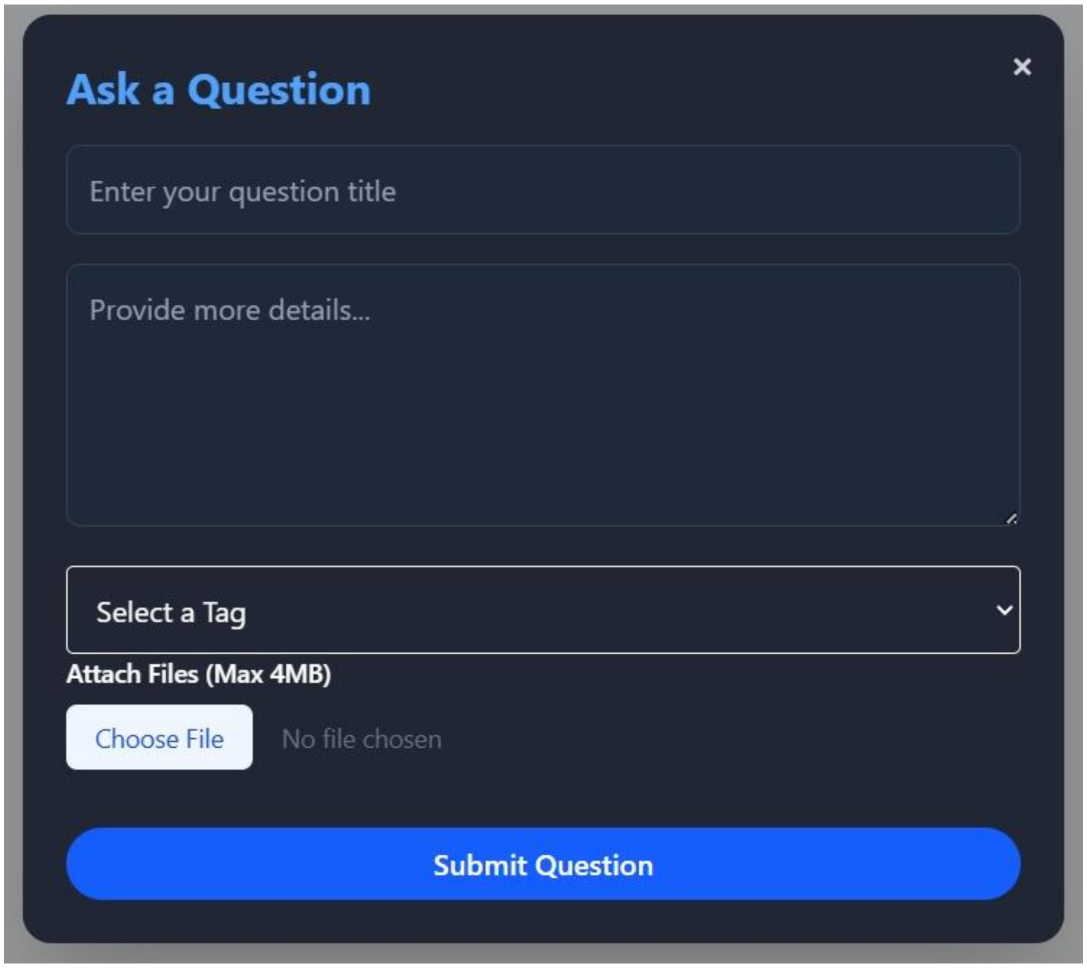


Figure 6: Ask Question Modal

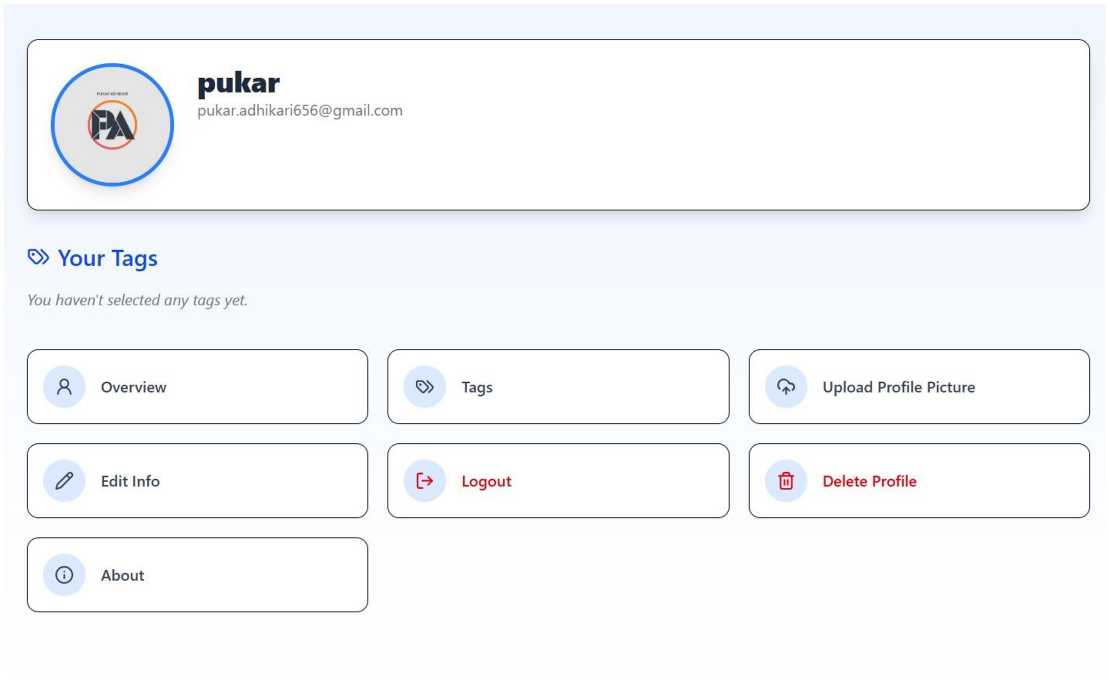


Figure 7: Profile Page