UNIFIED MENTOR

YOUR SKILL, SUCCESS & JOURNEY

UNIFIED MENTOR INTERNSHIP PROJECT REPORT 2024

Web Development Internship project on
Student Teacher Booking Appointment Webapp
Done by
S. Vikas

TABLE OF CONTENTS

PREFACE

During my internship at Unified Mentor, from June 5th to July 5th, I had the opportunity to dive deep into the field of web development. This period was marked by significant learning and practical application of modern web technologies. Under the guidance of robust resources, I was able to enhance my skills in HTML, CSS, JavaScript, and backend development, working on real-world projects that demanded both creativity and technical prowess.

My internship focused on developing and improving web applications, ensuring they were not only functional but also user-friendly and aesthetically pleasing. The experience provided me with a holistic view of the web development lifecycle, from conceptualization to final deployment, and instilled a deeper appreciation for the intricacies involved in creating robust web solutions.

Throughout my time at Unified Mentor, I was exposed to various aspects of web development, including front-end design, user interface (UI) and user experience (UX) optimization, and back-end programming. I worked on multiple projects that required me to integrate front-end interfaces with backend databases, ensuring seamless user interactions and efficient data handling.

One of the key projects I contributed to was the development of a student-teacher appointment booking system. This project involved creating a responsive and interactive user interface, implementing secure authentication mechanisms, and developing a robust backend using Python and Firebase. Working on this project allowed me to apply my skills in a meaningful way and see the direct impact of my work on user experience and system functionality.

INTRODUCTION

The Student-Teacher Appointment Booking System is an innovative web application designed to streamline the scheduling process between students and teachers. This project aims to provide an efficient, user-friendly platform that facilitates appointment management, communication, and administrative oversight. Developed using a combination of modern web technologies, the system leverages Python for the backend, HTML and CSS for the frontend, and Firebase for the database and authentication.

Project Objectives:

The primary objectives of this project are:

- Simplify Appointment Scheduling: Enable students to easily book appointments with teachers based on availability.
- Enhance Communication: Provide a platform for messaging between students and teachers.
- Administrative Management: Allow admins to manage student registrations and teacher records efficiently.
- Role-Based Access: Ensure secure access to different functionalities based on user roles (student, teacher, admin).

Key Features:

- Login and Registration
- Role-Based Dashboards
- Booking Request
- Booking Approval and Cancellation
- Direct Communication
- Message Management
- Student Registration Requests
- Student and Teacher Management

TASK DETAILS

Project Title: Student Teacher Booking Appointment

Technologies: HTML, CSS, JS and Firebase

Domain: Web Development (Education)

Project Difficulties Level: Easy

Problem Statement:

Booking appointment systems, either online or through traditional queueing systems, are now popular. Several businesses, such as scheduling an appointment, employ various Web-based appointment systems for their patients, which improve the efficiency of the appointment process, reducing patient wait times and increasing the total number of patients treated. This research proposes a web-based appointment booking system that allows students and lecturers to be aware of their appointment time regardless of where they are by using the web or mobile devices. By connecting to the Internet, students and instructors can easily access the system. It also permits students to send any message, including the appointment's purpose and timing.

System Modules:

- Admin:
    - Add Teacher
        - Name, Department, subject, etc
    - Update/Delete Teacher
    - Approve Registration Student

- Teacher:

  - Login
  - Schedule Appointment
  - Approve/cancel Appointment
  - View Messages
  - View All Appointment
  - Logout

- Student:
  - Register
  - Login
  - Search Teacher
  - Book Appointment
  - Send Message

Database:

You are supposed to use FireBase.

Logging:

Logging is a must for every action performed by your code, use the JavaScript or python logging library for this.

Deployment:

You can host your model in the cloud platform, edge devices, or maybe local, but with a proper justification of your system design.

Optimization of solutions:

Try to optimize your solution on code level, architecture level, and mention all of these things in your final submission.

PROPOSED DESIGN/MODEL

The proposed solution for the Student-Teacher Appointment Booking System is a robust, scalable web application designed to streamline appointment scheduling and communication. The system utilizes a three-tier architecture consisting of a responsive frontend, a secure backend, and a cloud-based database. The frontend, developed with HTML, CSS, and JavaScript, provides a user-friendly interface for students, teachers, and admins. The backend, powered by Flask (Python), handles the application logic, including user authentication, role management, and data processing. Firebase Firestore serves as the database, offering real-time data synchronization and secure storage of user information, appointments, and messages. The integration of Firebase Authentication ensures secure access and role-based permissions, enhancing the system's security and usability. This design ensures seamless interaction between components, real-time updates, and a scalable foundation for future enhancements.

Technologies Used:

- Frontend:
  - HTML and CSS: For creating a responsive and visually appealing user interface with easy navigation and content priority sorting throughout the interface
  - JavaScript: For dynamic content updates, user interactions, dashboard logic, application alerts, displaying data that has been fetched and browser storage

- Backend:
  - Python (Flask): For handling server-side logic, routing, and interactions with the database and also renders the code from HTML, CSS and JS.

- Database:
    - Firebase Firestore: For storing user data, appointment details, and messages.
    - Firebase Authentication: For managing user authentication and secure access.
    - Firebase Service: Having private firebase datastore key within a JSON file to ensure security and privacy.

System Architecture:

The system architecture consists of three primary components:

- Frontend: The user-facing part of the application, developed using HTML, CSS, and JavaScript. It communicates with the backend through API calls.
- Backend: The server-side logic implemented using Flask (Python). It handles requests from the frontend, processes data, and communicates with Firebase.
- Database: Firebase Firestore serves as the database, storing all necessary information such as user details, appointments, and messages.

User Flow:

- Initial Screen: Users are presented with a sign-in window with an option to register.
- Registration Request: Students submit a registration request, which is reviewed by an admin.
- Role-Based Dashboard: Upon successful login, users are directed to their respective dashboards:
- Students: Can book appointments and send messages.
- Teachers: Can view, approve, cancel, or delete appointments and manage messages.
- Admins: Can approve registration requests and manage both student and teacher records.

SOLUTIONS DESIGN

Low Level Diagram:

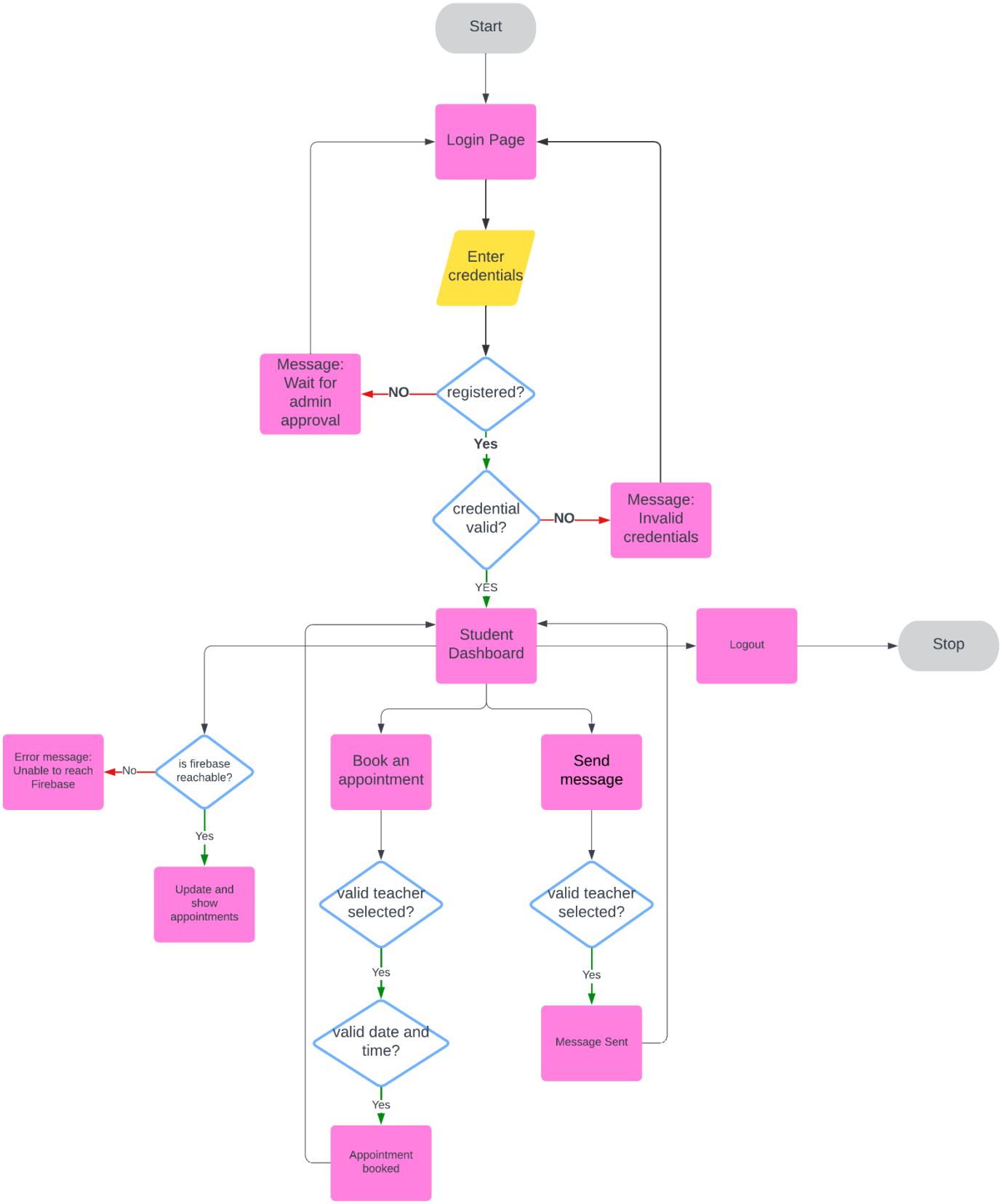
<table>
<tr><td colspan="2">User</td><td colspan="2">Appointment</td></tr>
<tr><td>- Email
- Password
- Role</td><td>+ login()
+ register()</td><td>- appointmentID
- teacherID
- studentID
- date
- time
- status</td><td>+ approve()
+ cancel()
+ delete()</td></tr>
<tr><td colspan="2">Message</td><td colspan="2">Admin</td></tr>
<tr><td>- messageID
- teacherID
- studentID
- content</td><td>+ delete()</td><td>- adminID
- name</td><td>+ approveStudent()
+ manageTeachers()</td></tr>
</table>

Sequence Diagram:

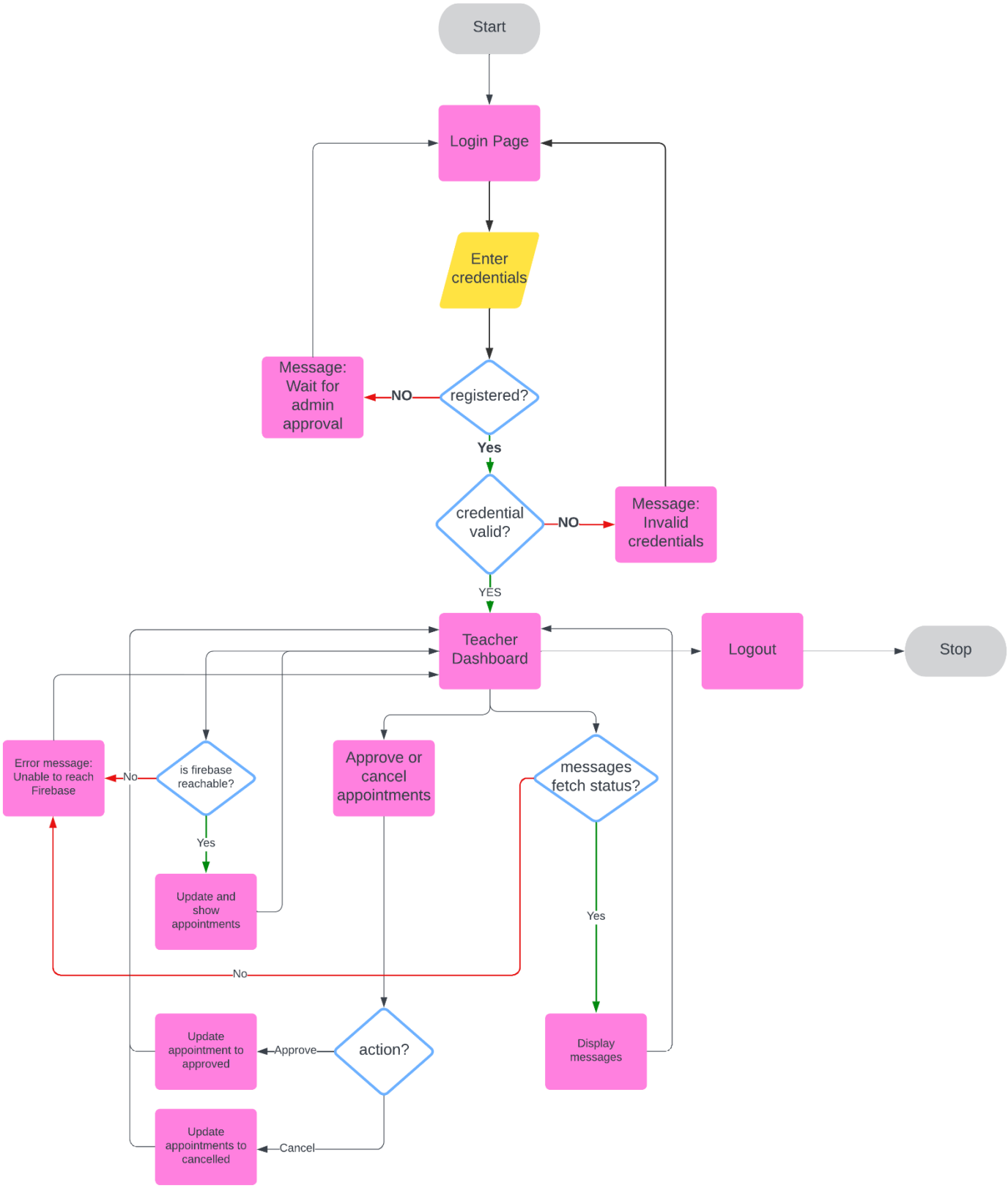
- User ->| POST /login |-> Flask Route ->| Validate Credentials |-> Firebase Auth
- Teacher ->| Click Approve |-> JS Function ->| Call Firestore API |-> Update Status

SYSTEM ARCHITECTURE

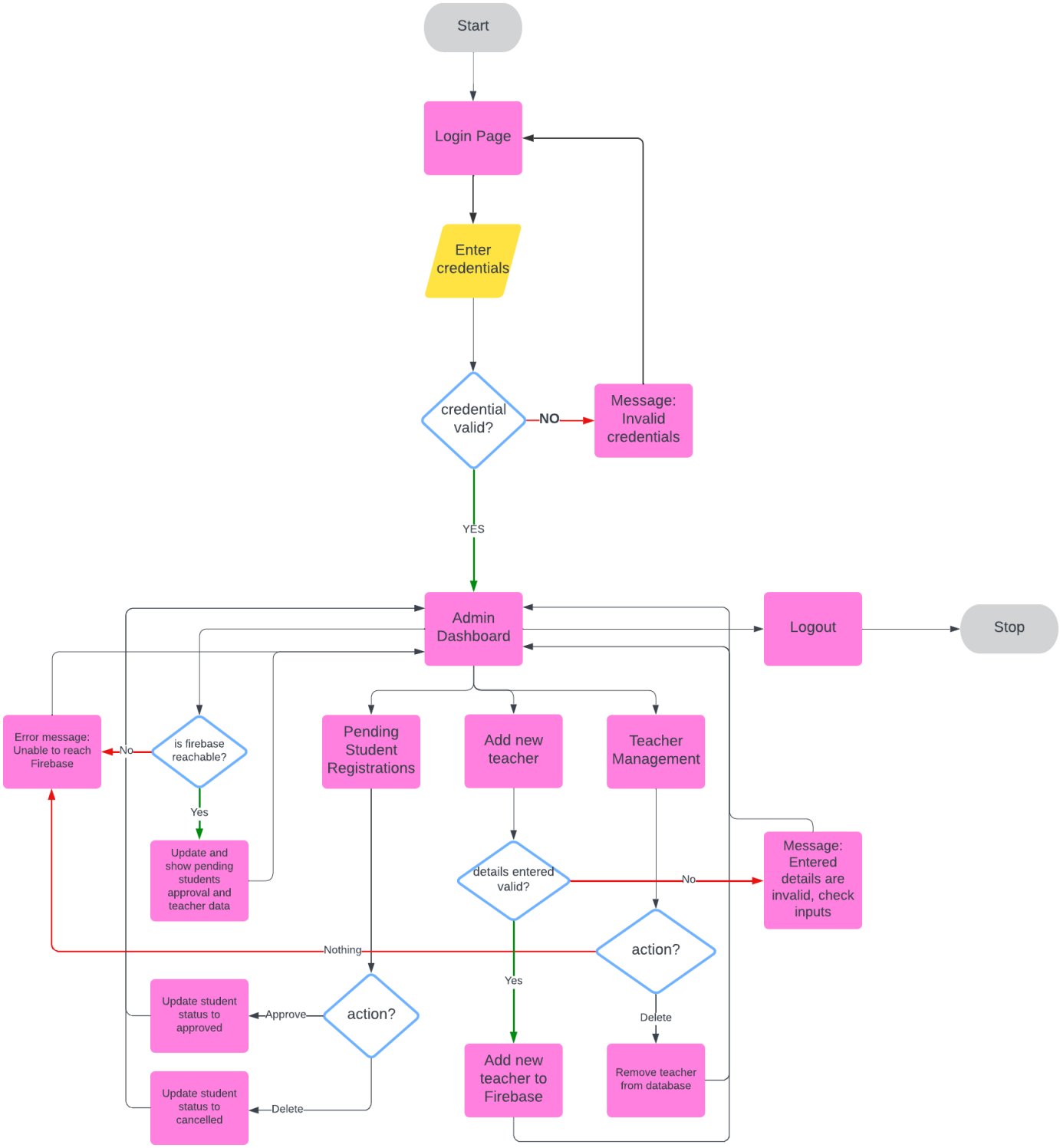
Student:



Teacher:


Start
Login Page
Enter credentials
registered?
NO
Message: Wait for admin approval
Yes
credential valid?
NO
Message: Invalid credentials
YES
Teacher Dashboard
Logout
Stop
Error message: Unable to reach Firebase
No
is firebase reachable?
Yes
Update and show appointments
Approve or cancel appointments
messages fetch status?
Yes
No
Update appointment to approved
Approve
action?
Update appointments to cancelled
Cancel
Display messages

Admin:


Start
Login Page
Enter credentials
credential valid?
NO
Message: Invalid credentials
YES
Admin Dashboard
Logout
Stop
Error message: Unable to reach Firebase
No
is firebase reachable?
Yes
Update and show pending students approval and teacher data
Pending Student Registrations
Add new teacher
Teacher Management
details entered valid?
No
Message: Entered details are invalid, check inputs
Nothing
action?
Yes
Delete
Update student status to approved
Approve
action?
Update student status to cancelled
Delete
Add new teacher to Firebase
Remove teacher from database

PERFORMANCE TEST

Google PageSpeed Insights:

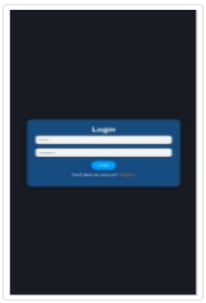
100 Performance | 92 Accessibility | 72 Best Practices | 90 SEO

100

Performance

Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

▲ 0–49 ■ 50–89 ● 90–100



METRICS Expand view

- First Contentful Paint 0.8 s
- Largest Contentful Paint 0.8 s
- Total Blocking Time 0 ms
- Cumulative Layout Shift 0
- Speed Index 1.0 s

100 Performance | 92 Accessibility | 75 Best Practices | 90 SEO

100

Performance

Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.

▲ 0–49 ■ 50–89 ● 90–100

METRICS Expand view

- First Contentful Paint 0.2 s
- Largest Contentful Paint 0.2 s
- Total Blocking Time 0 ms
- Cumulative Layout Shift 0
- Speed Index 0.5 s

Pingdom Website Speed Test:

Your Results:

DOWNLOAD HAR | SHARE RESULT

Performance grade	Page size
A 98	4.4 KB

Load time	Requests
469 ms	3

Content size by content type

CONTENT TYPE	PERCENT	SIZE
{ } CSS	58.44%	2.6 KB
HTML	28.19%	1.2 KB
Error	13.37%	590.0 B
Total	100.00%	4.4 KB

Requests by content type

CONTENT TYPE	PERCENT	REQUESTS
HTML	33.33%	1
{ } CSS	33.33%	1
Error	33.33%	1
Total	100.00%	3

Content size by domain

CONTENT TYPE	PERCENT	SIZE
stabapp.pythonanywhere.com	100.00%	3.6 KB
Total	100.00%	3.6 KB

Requests by domain

CONTENT TYPE	PERCENT	REQUESTS
stabapp.pythonanywhere.com	100.00%	3
Total	100.00%	3

Improve page performance

GRADE	SUGGESTION
B 89	Add Expires headers
A 100	Avoid empty src or href
A 100	Put JavaScript at bottom
A 100	Reduce the number of DOM elements
A 100	Make favicon small and cacheable
A 100	Avoid HTTP 404 (Not Found) error
A 100	Avoid URL redirects

MY LEARNINGS

Working on the Student-Teacher Booking Appointment App has been an enriching journey, offering numerous opportunities to enhance my technical and problem-solving skills. One of the key learnings was mastering the integration of Firebase for authentication and database management. This provided me with a deep understanding of how to handle user data securely and efficiently, and manage real-time updates seamlessly.

Additionally, developing the backend with Python and Flask reinforced my knowledge of server-side scripting, API development, and database interactions. On the frontend, working with HTML, CSS, and JavaScript allowed me to create a responsive and interactive user interface. I focused on modern design principles, ensuring a fluid user experience that adapts well across different devices and screen sizes.

The project also taught me the importance of robust error handling and logging. By implementing detailed logging mechanisms, I was able to track user actions and debug issues more effectively, improving the overall reliability of the application. I also explored performance optimization techniques, such as minimizing load times and ensuring smooth transitions, which are crucial for a positive user experience.

Finally, this project highlighted the significance of good project management practices. From initial planning and requirement gathering to design, implementation, and testing, I learned how to manage my time effectively, collaborate with team members, and adapt to changes and challenges along the way. This comprehensive experience has prepared me to handle more complex projects in the future, with a balanced approach to both technical and managerial aspects.

FUTURE SCOPE

The Student-Teacher Booking Appointment App has significant potential for future development and expansion, paving the way for a more comprehensive and sophisticated educational platform. One key area for future enhancement is the integration of advanced analytics and reporting tools. By incorporating data analytics, the app can provide insights into student engagement, appointment trends, and teacher performance, helping administrators make data-driven decisions to improve the educational experience. Additionally, introducing machine learning algorithms can enable personalized recommendations for students, such as suggesting available time slots based on their past booking behaviour and preferences.

The app can also benefit from a more robust notification and reminder system. Implementing push notifications and email alerts for upcoming appointments, cancellations, and important announcements will ensure that users stay informed and engaged. Additionally, enhancing the calendar integration to sync with popular calendar apps like Google Calendar and Outlook can streamline the scheduling process for both students and teachers.

Furthermore, expanding the user management system to include more roles and permissions, such as guest speakers, tutors, and counsellors, can diversify the app's functionality and cater to a broader audience. Integrating third-party educational resources and platforms, like learning management systems (LMS) and digital libraries, can also enrich the app's offerings and provide students with a one-stop solution for their academic needs.

Lastly, ensuring scalability and adaptability to different educational institutions and geographical regions is crucial. By developing customizable templates and localization features, the app can be tailored to meet the specific requirements of various schools, colleges, and universities worldwide.