Student–Teacher Appointment Booking System

1. Executive Summary

The **Student–Teacher Appointment Booking System** is a web-based appointment management platform designed to digitize and streamline the process of scheduling academic appointments within educational institutions. The system addresses the inefficiencies of manual appointment booking by providing a centralized, role-based digital solution for students, teachers, and administrators.

Developed using **Vanilla JavaScript, HTML5, and CSS3**, the application follows a frontend-focused architecture integrated with **Firebase** for authentication, real-time data synchronization, and secure data storage. User authentication is implemented through Firebase Authentication, while all appointment and user-related data is managed using Firebase Firestore, a NoSQL document database. Real-time updates are achieved using Firestore's onSnapshot listeners, ensuring instant synchronization across all user roles.

The system supports multi-role access control, allowing students to request appointments, teachers to manage and respond to requests, and administrators to oversee registrations, approve users, and monitor system activity. Security is enforced through comprehensive Firestore security rules, role-based permissions, and authenticated session handling.

The project demonstrates strong competency in modern web application development, particularly in real-time systems, frontend modularization, and secure cloud-based backend integration. The final application is deployed on **Vercel**, ensuring high availability and performance. Overall, the system successfully delivers an efficient, scalable, and secure solution for academic appointment management while showcasing industry-relevant development practices.

2. Introduction & Problem Statement

Educational institutions traditionally rely on manual or semi-digital processes for managing student–teacher appointments. These processes often involve physical visits, handwritten registers, email exchanges, or verbal coordination, which can lead to scheduling conflicts, lack of transparency, time wastage, and poor record management. As institutions scale, these inefficiencies become increasingly difficult to manage.

Students frequently face challenges such as limited access to teachers, uncertainty regarding appointment availability, and delayed confirmations. Teachers, on the other hand, struggle to manage overlapping requests, track appointment history, and maintain structured schedules. Administrators lack centralized visibility into appointment activities and user management, making system-wide oversight difficult.

The **Student–Teacher Appointment Booking System** is proposed as a digital solution to these challenges. The system aims to replace fragmented manual workflows with a unified, web-based platform that enables structured appointment scheduling, real-time updates, and transparent communication among all stakeholders.

Project Objectives

- Digitize the appointment booking process

- Enable role-based access for students, teachers, and administrators
- Provide real-time appointment status updates
- Ensure data security and privacy
- Improve efficiency, transparency, and accountability

Project Scope

The scope includes frontend development, real-time database integration, authentication, role-based dashboards, and secure deployment. Backend server development is intentionally excluded by leveraging Firebase's managed services.

---

3. System Design & Architecture

3.1 Overall Architecture

The system follows a **frontend-driven, cloud-backed architecture**:

- Frontend: HTML5, CSS3, Vanilla JavaScript
- Authentication: Firebase Authentication
- Database: Firebase Firestore
- Hosting: Vercel

All business logic runs in the browser, while Firebase handles authentication, storage, and real-time updates.

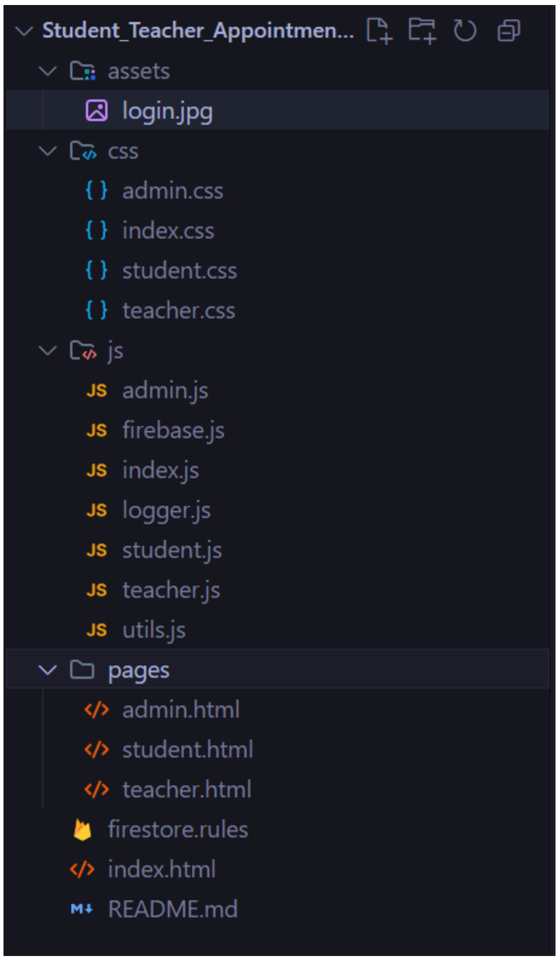
3.2 Database Design Rationale

Firestore is used due to its:

- NoSQL flexibility
- Real-time synchronization
- Scalability
- Fine-grained security rules

Collections are logically separated into users, appointments, and logs to ensure data normalization and efficient querying.

3.3 Folder Structure



```
Student_Teacher_Appointmen...
  assets
    login.jpg
  css
    admin.css
    index.css
    student.css
    teacher.css
  js
    admin.js
    firebase.js
    index.js
    logger.js
    student.js
    teacher.js
    utils.js
  pages
    admin.html
    student.html
    teacher.html
  firestore.rules
  index.html
  README.md
```

3.4 Security Implementation Strategy

Security is enforced at multiple levels:

- Authentication via Firebase Auth
- Role-based Firestore security rule
- Data isolation between users
- Admin-only access to sensitive operations

This layered approach ensures confidentiality, integrity, and availability of data.

3.5 System Workflow (Textual Description)

1. User registers and logs in
2. Admin approves student or manages teacher profiles
3. Student browses teachers and submits appointment request
4. Teacher reviews and approves/rejects requests
5. Real-time updates reflect changes across all dashboards

6. All actions are logged for auditing

---

4. Implementation Details

4.1 Modular Code Organization

Each role has a dedicated JavaScript file:

- index.js – authentication
- student.js – student features
- teacher.js – teacher operations
- admin.js – administrative controls

Utility and logging logic are abstracted into reusable modules.

4.2 Authentication System

- Email/password authentication
- Session persistence
- Role-based redirection after login
- Password reset functionality

4.3 Real-Time Functionality

Firestore onSnapshot listeners ensure:

- Instant appointment status updates
- Live dashboard counters
- Real-time monitoring without page refresh

4.4 User Interface Development

- Responsive layouts using Flexbox and Grid
- Role-specific dashboards
- Toast notifications for actions and errors
- Modal dialogs for confirmations

4.5 Database Integration

All CRUD operations are securely routed through Firestore with validation and permission checks.

---

5. Testing & Validation

Testing Methodology

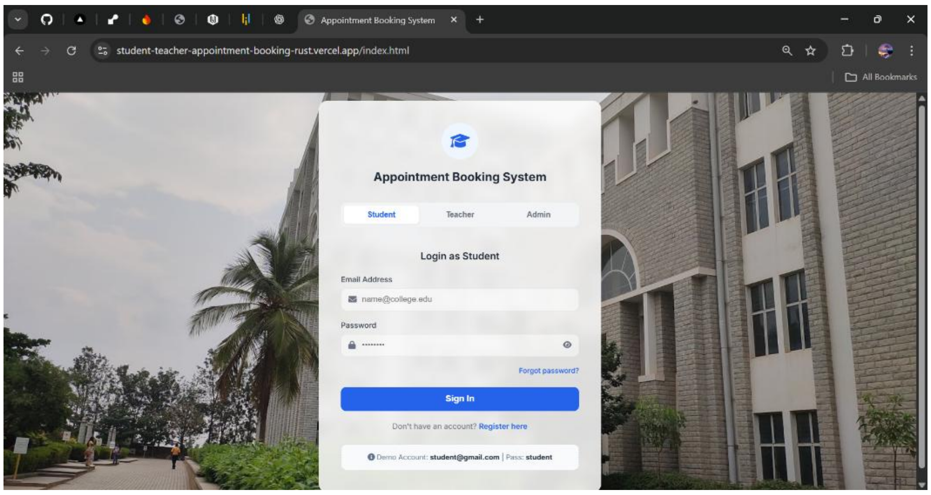
- Manual functional testing for each role
- Cross-browser testing (Chrome, Edge, Firefox)
- Responsive testing on mobile and desktop
- Security rule validation using Firebase Emulator

User Acceptance Testing

Demo users validated usability, clarity, and responsiveness.

---

6. Project Visuals



Appointment Booking System

student-teacher-appointment-booking-rust.vercel.app/index.html

Appointment Booking System

Login as Student

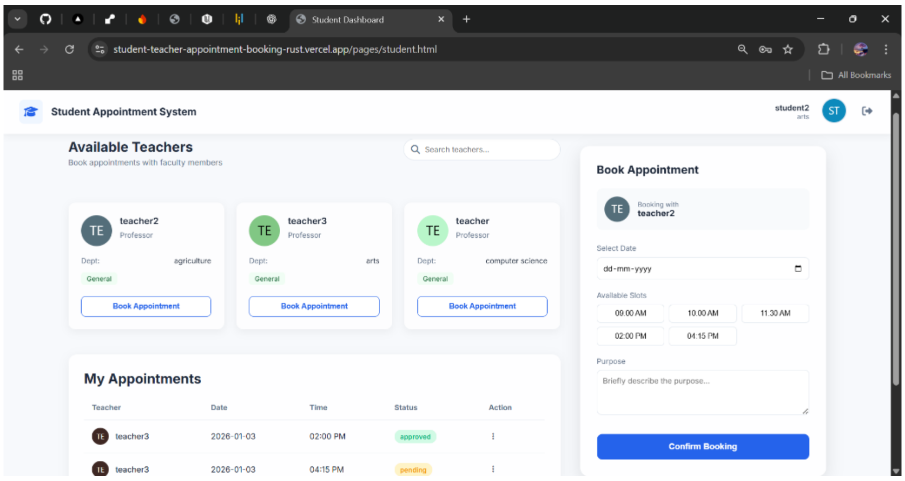
Email Address: name@college.edu

Password: ••••••••

Don't have an account? Register here

Demo Account: **student@gmail.com** | Pass: **student**

Figure 1: Login & registration screen



Student Dashboard

student-teacher-appointment-booking-rust.vercel.app/pages/student.html

Student Appointment System

student2 arts

Available Teachers

Book appointments with faculty members

teacher2 Professor	teacher3 Professor	teacher Professor
Dept: agriculture	Dept: arts	Dept: computer science
General	General	General

Book Appointment

Booking with teacher2

Select Date: dd-mm-yyyy

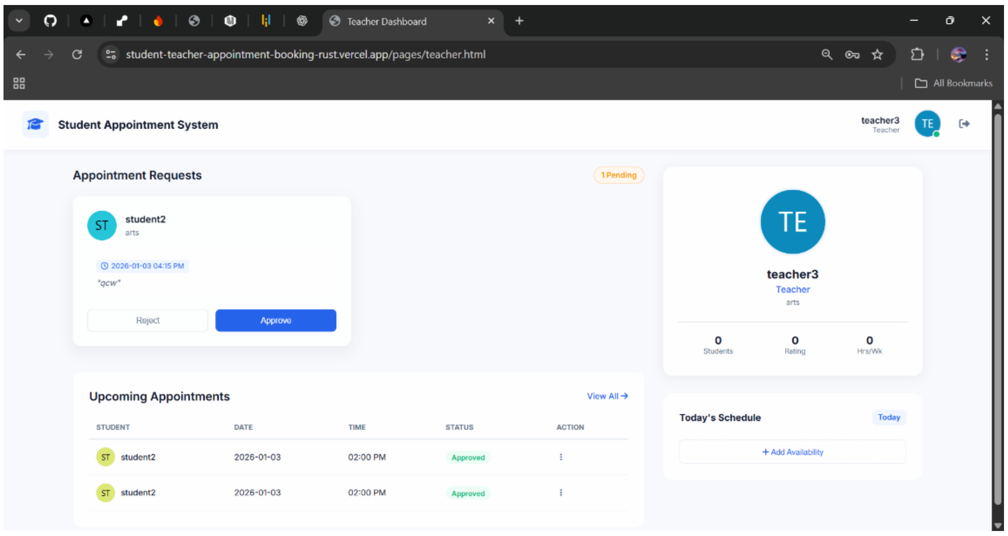
Available Slots: 09.00 AM, 10.00 AM, 11.30 AM, 02:00 PM, 04:15 PM

Purpose: Briefly describe the purpose...

My Appointments

Teacher	Date	Time	Status	Action
teacher3	2026-01-03	02:00 PM	approved	⋮
teacher3	2026-01-03	04:15 PM	pending	⋮

Figure 2: Student booking interface



student2
arts
2026-01-03 04:15 PM
"qcw"

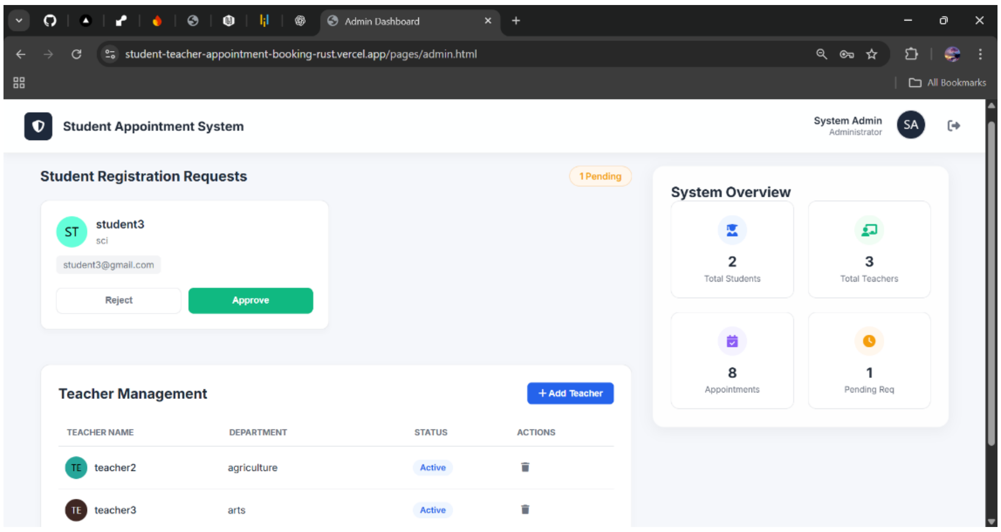
teacher3
Teacher
arts

0	0	0
Students	Rating	Hrs/Wk

Upcoming Appointments

STUDENT	DATE	TIME	STATUS	ACTION
student2	2026-01-03	02:00 PM	Approved	⋮
student2	2026-01-03	02:00 PM	Approved	⋮

Figure 3: Teacher approval dashboard


Admin Dashboard
student-teacher-appointment-booking-rust.vercel.app/pages/admin.html
Student Appointment System
System Admin
Administrator
SA
Student Registration Requests
1 Pending
Reject
Approve
+ Add Teacher

student3
sci
student3@gmail.com

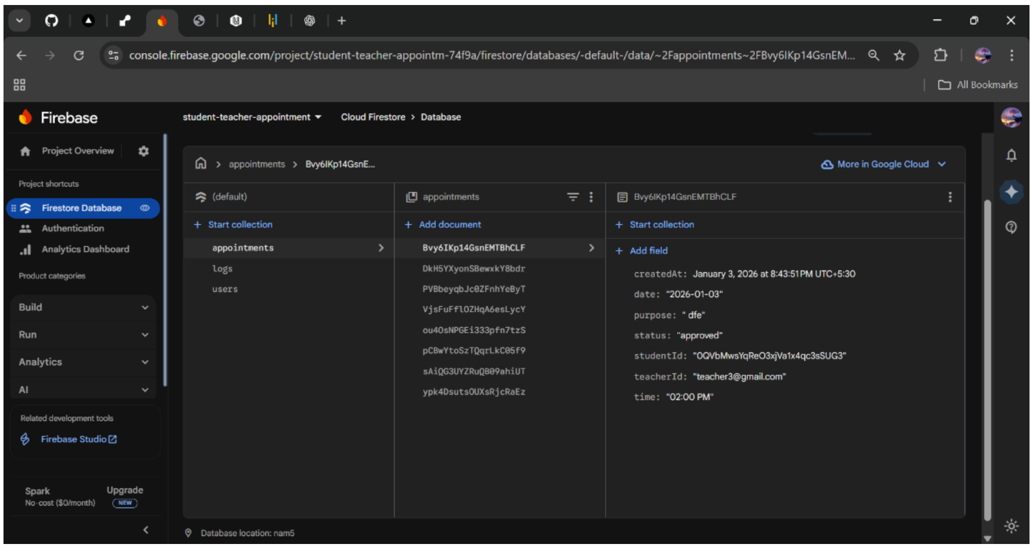
System Overview

2	3
Total Students	Total Teachers
8	1
Appointments	Pending Req

Teacher Management

TEACHER NAME	DEPARTMENT	STATUS	ACTIONS
teacher2	agriculture	Active	🗑
teacher3	arts	Active	🗑

Figure 4: Admin control panel



appointments > Bvy6IKp14GsnE...

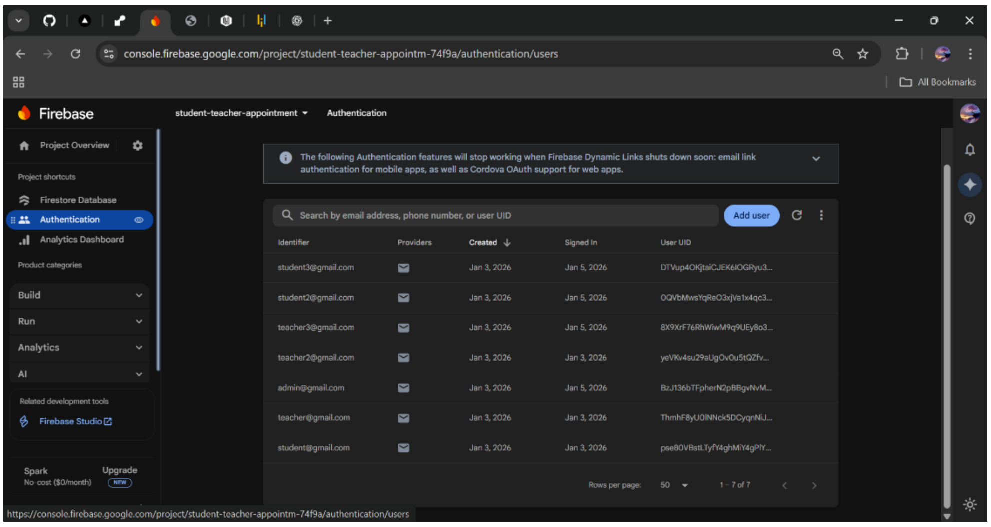
(default): appointments, logs, users

appointments: Bvy6IKp14GsnEMTBhCLF, DkH5YXyonSBewxkY8bdr, PVBbeyqbJc0ZFnhYeByT, VjsFuFfLOZHqA6esLycY, ou4OsNPGEi333pfn7tzS, pCBwYtoSzTQqrLkC05f9, sAiQG3UYZRuQ809ahiUT, ypk4DsutsOUXsRjcRaEz

Bvy6IKp14GsnEMTBhCLF

```
createdAt: January 3, 2026 at 8:43:51 PM UTC+5:30
date: "2026-01-03"
purpose: " dfe"
status: "approved"
studentId: "OQVbMwsYqReO3xjVa1x4qc3sSUG3"
teacherId: "teacher3@gmail.com"
time: "02:00 PM"
```

Figure 5: Firestore Database


console.firebase.google.com/project/student-teacher-appointm-74f9a/authentication/users
All Bookmarks
Firebase
student-teacher-appointment
Authentication
Project Overview
Project shortcuts
Firestore Database
Authentication
Analytics Dashboard
Product categories
Build
Run
Analytics
AI
Related development tools
Firebase Studio
Spark
No-cost ($0/month)
Upgrade
NEW
Search by email address, phone number, or user UID
Add user
Rows per page: 50
1 – 7 of 7
https://console.firebase.google.com/project/student-teacher-appointm-74f9a/authentication/users

The following Authentication features will stop working when Firebase Dynamic Links shuts down soon: email link authentication for mobile apps, as well as Cordova OAuth support for web apps.

Identifier	Providers	Created	Signed In	User UID
student3@gmail.com	✉	Jan 3, 2026	Jan 5, 2026	DTVup4OKjtaiCJEK6IOGRyu3...
student2@gmail.com	✉	Jan 3, 2026	Jan 5, 2026	0QVbMwsYqReO3xjVa1x4qc3...
teacher3@gmail.com	✉	Jan 3, 2026	Jan 5, 2026	8X9XrF76RhWiwM9q9UEy8o3...
teacher2@gmail.com	✉	Jan 3, 2026	Jan 3, 2026	yeVKv4su29aUgOv0u5tQZfv...
admin@gmail.com	✉	Jan 3, 2026	Jan 5, 2026	BzJ136bTFpherN2pBBgvNvM...
teacher@gmail.com	✉	Jan 3, 2026	Jan 3, 2026	ThmhF8yU0lNNck5DCyqnNiJ...
student@gmail.com	✉	Jan 3, 2026	Jan 3, 2026	pse80VBstLTyfY4ghMiY4gPlY...

Figure 6: Firebase Authentication (Test data added for testing)

7. Conclusion

The Student–Teacher Appointment Booking System successfully achieves its objective of digitizing and optimizing academic appointment scheduling. By leveraging Firebase's real-time capabilities and a modular frontend architecture, the system delivers a secure, scalable, and user-friendly solution.

The project demonstrates strong understanding of real-time web applications, cloud-based backend services, and role-based system design. It is well-suited for academic evaluation and serves as a solid foundation for future enhancements and production deployment.