



iJRASET

International Journal For Research in
Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** XI **Month of publication:** November 2025

DOI: <https://doi.org/10.22214/ijraset.2025.75320>

www.ijraset.com

Call: ☎ 08813907089

E-mail ID: ijraset@gmail.com

Unilink - Staff Availability and Assignment System

Akbar E Azam A¹, Goutham E², Chandhan S³, Mrs. C. Akshaya⁴

^{1,2,3}Bachelor of Engineering In Computer Science And Engineering, Adhiyamaan College Of Engineering, (AN AUTONOMOUS INSTITUTION), ANNA University, Chennai

⁴Assistant Professor, Department of CSE, Adhiyamaan College of Engineering (Autonomous), Anna University, Chennai

Abstract: In educational institutions, managing communication between market potential of a students, staff, and administrators often becomes time-consuming and inefficient when done manually. The “UniLink – Staff Availability and Assignment System” is a web-based application developed to simplify and digitalize these processes. The system provides an online platform that allows students to check staff availability, send meeting requests, and submit assignments efficiently. The application is built using HTML, CSS, and JavaScript for the front end, with Firebase Firestore as the backend database for secure and real-time data management. Each user—student, staff, or Head of Department (HOD)—has specific access privileges that ensure proper control and coordination. Staff members can create and manage assignments, respond to student meeting requests, and view timetables, while the HOD can manage users and oversee all system activities through an admin dashboard. By integrating real-time alerts and notifications, UniLink ensures instant updates on meetings, approvals, and assignments, thereby reducing delays and improving academic workflow. The system enhances communication, saves time, and promotes a paperless environment within the institution. Overall, this project aims to create a reliable, scalable, and user-friendly academic management platform that improves productivity and coordination across all departments.

I. INTRODUCTION

A. Overview

The UniLink – Staff Availability and Assignment System is an innovative web-based platform developed to streamline communication and coordination within a college environment. This system serves as a bridge between students, faculty members, and the Head of Department (HOD), offering a unified digital solution for managing staff availability, academic timetables, and assignment distribution. Traditional manual processes—such as checking whether a staff member is available, assigning coursework, or requesting a meeting—often lead to confusion and time delays. UniLink effectively addresses these challenges by integrating all essential academic management features into one centralized and user-friendly interface.

The application allows students to log in using their register number and date of birth to access real-time details about staff schedules, upcoming assignments, and meeting options. Faculty members can use the system to upload and manage assignments, set their availability status, and respond to student meeting requests efficiently. Meanwhile, the HOD is provided with full administrative control to monitor all activities, manage user accounts, and ensure smooth communication across the department.

Built using HTML, CSS, and JavaScript for the frontend and Firebase Firestore as the backend database, UniLink ensures real-time data synchronization, secure access, and a seamless user experience across devices.

The system includes a dynamic timetable feature that visually displays daily staff schedules with color-coded subjects and merged sessions for clarity. Additionally, it incorporates an alert and notification system that instantly updates users about meeting confirmations, assignment uploads, or schedule changes.

UniLink stands out for its simplicity, accessibility, and responsiveness. The interface supports both light and dark modes, enhancing user comfort and visual appeal. Through this platform, all academic participants—students, staff, and administrators—can save valuable time, reduce paperwork, and maintain transparency in communication. Overall, UniLink contributes to a more organized and technology-driven academic environment, aligning with modern digital transformation goals in education.

B. Objectives

The primary objective of the UniLink – Staff Availability and Assignment System is to create a centralized digital platform that enhances communication, coordination, and information sharing between students, staff, and the Head of Department (HOD). The system aims to simplify academic management processes by automating staff availability tracking and assignment management.

- 1) To provide a unified web portal where students, staff, and HODs can access their respective modules based on user roles.
- 2) To display real-time staff availability, allowing students to know when faculty members are free or busy.

- 3) To enable staff members to create, update, and share assignments with students in a structured and timely manner.
- 4) To allow students to view and submit assignments digitally, reducing the dependency on manual submissions.
- 5) To implement a meeting request system through which students can request appointments with staff, and staff can accept or reject them.
- 6) To provide administrative control to HODs, enabling them to manage user accounts, edit profiles, and monitor department-level activities.
- 7) To introduce automated notifications and alerts for meetings, new assignments, and schedule updates to keep users informed.
- 8) To minimize time wastage and enhance productivity by replacing manual communication methods with an efficient digital platform.
- 9) To design a modern, responsive user interface with options for dark and light modes, ensuring accessibility across devices.
- 10) To ensure data storage, retrieval, and security through the use of Firebase Firestore, offering real-time synchronization and cloud-based access.

II. INTRODUCTION

A literature survey helps in understanding the background, existing systems, and technological advancements related to the development of the UniLink – Staff Availability and Assignment System. Several studies and systems have been developed in the area of academic management, faculty scheduling, and student– staff communication. This section reviews the most relevant works and technologies that inspired the creation of UniLink.

Traditional college management systems mainly relied on manual record-keeping and face-to-face communication between students and faculty. These methods were time-consuming, error-prone, and inefficient, particularly when dealing with large student groups and multiple departments. As technology evolved, many institutions adopted digital management portals to automate administrative processes such as attendance, timetable generation, and assignment submissions.

According to recent research on Learning Management Systems (LMS) like Moodle and Google Classroom, such platforms help improve communication between instructors and learners by centralizing academic information. However, these platforms focus mainly on coursework and lack real-time staff availability features. To address this limitation, several faculty management systems were developed, such as Timetable Management Systems that automate scheduling and conflict detection. Yet, they too are often limited to timetable generation and do not support interaction features like meeting requests or instant notifications.

Studies on Firebase-based educational systems highlight the efficiency of cloud-based databases in handling real-time data synchronization across users. Firebase Firestore, in particular, provides scalability, offline access, and secure authentication — all crucial for academic web applications. Integrating Firebase with web technologies such as HTML, CSS, and JavaScript has proven effective in developing dynamic, user-friendly interfaces for real-time collaboration.

Existing literature also emphasizes the importance of role-based access control in academic systems. Role segregation (Student, Staff, and Admin) ensures data privacy, accountability, and efficient task management. Research on user interface design further suggests that a clean and responsive interface improves usability and accessibility, encouraging wider adoption of such systems among students and faculty.

After analyzing these systems, the need for a more comprehensive platform that combines real-time staff availability, assignment management, and meeting coordination became evident. The UniLink system bridges this gap by integrating the most useful features of existing learning and scheduling platforms into a single, cloud-connected solution. It stands out for providing instant staff availability tracking, automated assignment workflows, and interactive meeting management, all within an intuitive and responsive web interface.

III. SYSTEM ANALYSIS

A. Existing System

The existing system in most colleges relies on manual or semi-digital processes to manage staff availability and assignment tracking. Students often approach staff directly to check their availability, and assignments are distributed or submitted offline or through informal platforms like email or WhatsApp. This results in time delays, communication gaps, and lack of proper record management. The absence of an integrated digital platform makes coordination between students, staff, and HODs inefficient.

Limitations of the Existing System:

- 1) No real-time information on staff availability.
- 2) Manual process for sharing and collecting assignments.
- 3) Lack of centralized communication between students, staff, and HOD.
- 4) No automated notification or meeting request system.
- 5) Data inconsistency and difficulty in maintaining academic records.
- 6) Time-consuming and prone to human errors.
- 7) Limited role-based access control and user management.

B. Proposed System

The proposed UniLink system aims to overcome the drawbacks of the existing manual process by providing a centralized digital platform that connects students, staff, and HODs efficiently. It allows real-time tracking of staff availability, digital assignment management, meeting requests, and automated notifications.

The system uses Firebase Firestore as its backend for secure cloud storage and HTML, CSS, and JavaScript for an interactive and responsive user interface. Through UniLink, all users can access their respective modules anytime, improving coordination, transparency, and productivity within the institution.

Features of the Proposed System:

- 1) Real-time staff availability display and status management
- 2) Digital assignment creation, submission, and tracking.
- 3) Role-based login system for students, staff, and HODs.
- 4) Meeting request and approval system between students and staff.
- 5) Automated alerts and notifications for meetings and assignments.
- 6) HOD admin panel for user management and department monitoring.
- 7) Cloud-based data storage using Firebase for security and reliability.
- 8) Modern and responsive UI with light/dark mode options.
- 9) Time-saving, error-free, and easily accessible web-based platform.

C. Proposed Solution

The proposed solution introduces UniLink, a web-based system designed to digitally manage staff availability and academic assignments within a college. It provides a structured, role-based environment that enhances communication and coordination among students, staff, and HODs.

The solution replaces manual processes with an automated platform that ensures real-time updates, centralized data storage, and easy accessibility from any device. By integrating Firebase Firestore as the backend database, the system enables instant synchronization of data such as user details, assignments, and meeting notifications.

Key Highlights of the Proposed Solution:

- 1) A unified digital platform for students, staff, and administrators.
- 2) Real-time staff availability display and schedule management.
- 3) Automated assignment creation, submission, and storage in Firebase.
- 4) Meeting request and response feature with instant notifications.
- 5) Role-based login for secure and personalized user access.
- 6) Centralized data management to avoid duplication and manual errors.
- 7) Modern, responsive, and user-friendly web interface.
- 8) Cloud-based access ensuring reliability, scalability, and efficiency.

D. Ideation & Brainstroming

The idea for developing the UniLink – Staff Availability and Assignment System arose from the need to simplify communication and task management between students, staff, and the HOD. The brainstorming sessions focused on understanding existing challenges and designing a solution that provides real-time information, automated workflows, and ease of use through a web-based platform.

Key Points from Ideation and Brainstorming:

- 9) Identified common issues such as difficulty in tracking staff availability and managing assignments manually.
- 10) Discussed various solutions and decided on a web-based system instead of a mobile app for easy accessibility.
- 11) Focused on creating an interactive, user-friendly interface for all user roles.
- 12) Chose Firebase Firestore as the backend for real-time database and cloud synchronization.
- 13) Planned key modules: Login & Authentication, Staff Availability, Assignments, Meeting Requests, and Admin Panel.
- 14) Designed the system with role-based access control to ensure data privacy and organized functionality.
- 15) Emphasized using HTML, CSS, and JavaScript for frontend development for simplicity and responsiveness.
- 16) Ensured the system includes notifications, dark/light mode, and real-time updates for a modern user experience.

E. Problem-Solution Fit

1) Problem

In most educational institutions, students and staff face significant communication barriers and time management issues. Students often struggle to find out whether staff are available for queries or guidance. Assignments are usually managed manually or through informal communication channels, leading to confusion, missed deadlines, and loss of records. There is no unified system to manage staff schedules, assignments, and meeting requests efficiently. The lack of automation results in wasted time, poor coordination, and administrative difficulties for the Head of Department (HOD).

2) Solution

The UniLink – Staff Availability and Assignment System addresses these issues by providing a centralized web-based platform that connects students, staff, and HODs. It allows real-time tracking of staff availability, digital assignment management, and meeting scheduling through an interactive interface. Using Firebase Firestore as the backend, the system ensures secure cloud storage, instant data synchronization, and easy access from any device.

This proposed system perfectly fits the problem by offering:

- Automation: Replaces manual assignment tracking and staff coordination with a digital system.
- Accessibility: A web-based platform that can be accessed by students, staff, and HOD anytime, anywhere.
- Transparency: Real-time updates on staff status, assignments, and meeting requests.
- Efficiency: Reduces time wastage by automating notifications and streamlining academic tasks.
- Scalability: Cloud-based Firebase integration allows smooth handling of multiple users and data simultaneously.
- Security: Role-based access ensures that user data remains private and protected.

F. Aechitecture Design

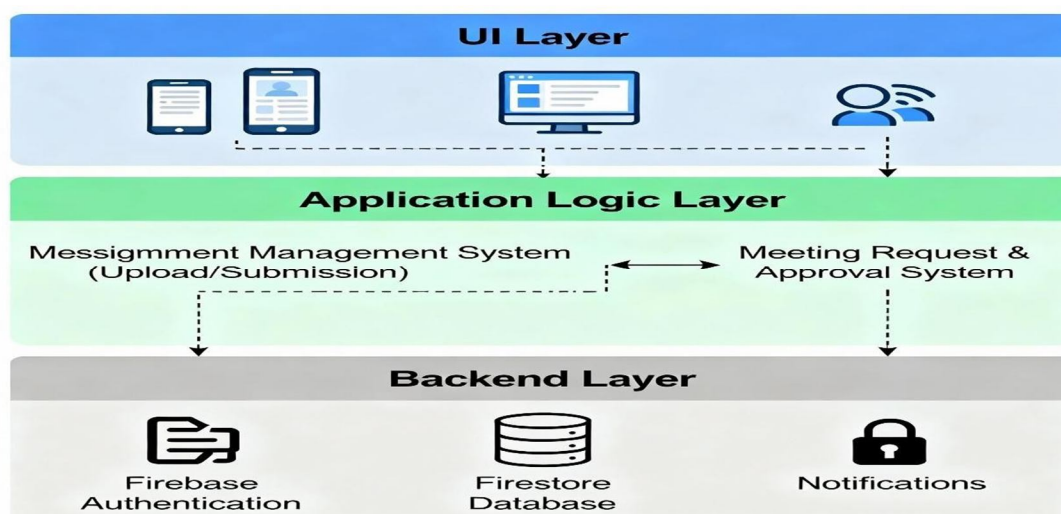


Figure 3.1: Architecture Diagram

- 1) UI Layer (User Interface Layer)
 - Represents the front-end of the system where users interact.
 - Designed for students, staff, and HODs using mobile phones or computers.
 - Provides a responsive web interface for login, viewing timetables, submitting assignments, and managing meetings.
- 2) Application Logic Layer

Contains the core modules such as:

 - Assignment Management System: Allows staff to upload assignments and students to submit their work digitally.
 - Meeting Request & Approval System: Manages student-staff meeting scheduling, acceptance, or rejection.
- 3) Backend Layer
 - Responsible for data management, authentication, and notifications.
 - Built using Firebase technologies to ensure secure, real-time, and cloud-based operations.
 - Components include:
 - Firebase Authentication: Verifies user credentials (student, staff, HOD) and ensures secure login access.
 - Firestore Database: Stores user details, assignments, meeting data, and staff availability information.
 - Notifications Module: Sends alerts for meetings, new assignments, and updates to respective users.
 - Provides scalability and synchronization, allowing multiple users to access data simultaneously.

G. Data Flow Diagram

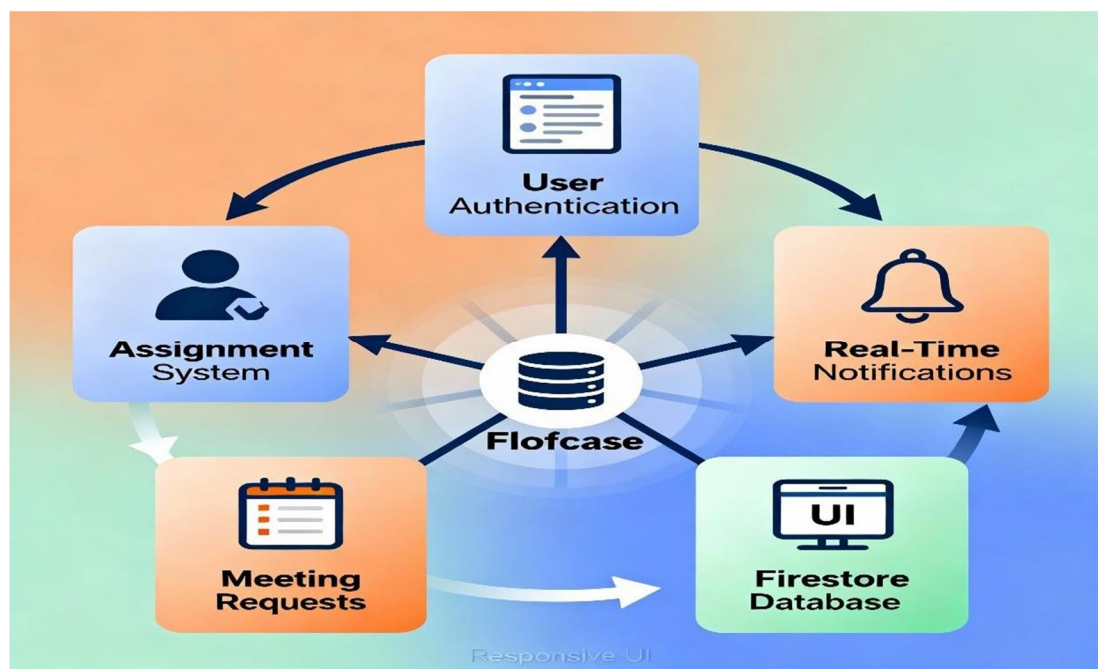


Figure 3.2: Data Flow Diagram

- 1) User Interface (UI)
 - Responsive design for various device sizes.
 - Login page with role-based authentication.
 - Pages for Timetable, Assignments, Meetings, Notifications, Profile, and Settings.
 - User-friendly forms for assignment uploads and meeting requests.

- 2) Firebase Authentication:
 - Secure login handling.
 - Role-based user validation (student, staff, HOD).
 - Session management.
- 3) Firestore Database:
 - Stores user data, timetable, assignments, meetings, notifications.
 - Supports real-time data sync and offline persistence
 - Structured collections for assignments, submissions, meetings, notifications.
- 4) Assignment Management System:
 - Create, read, update, delete assignments.
 - File upload and submission tracking.
 - Assignment status (pending, submitted, overdue).
 - Staff access to submission lists and downloads.
- 5) Meeting Request System:
 - Schedule and request meetings between users.
 - Meeting status management (pending, approved, rejected).
 - Notification triggers on request and approval.
- 6) Notification System:
 - Real-time notifications for assignments, meetings, and system alerts.
 - Accessible from UI in notifications section.
- 7) Application Logic Layer:
 - Mediates between UI and backend.
 - Implements business rules, validations.
 - Handles events, form submissions, state management.

IV. SYSTEM REQUIREMENT

A. Hardware Requirements

For Development System (Developer Machine)

- Processor: Intel Core i3 or above
- RAM: Minimum 8 GB
- Hard Disk: Minimum 250 GB (SSD recommended)
- Display: 1366 × 768 resolution or higher
- Internet Connection: Required for Firebase integration and real-time synchronization

For End Users (Students, Staff, HOD)

- Device: Desktop, Laptop, Tablet, or Smartphone
- Processor: Dual-Core or higher
- RAM: Minimum 2 GB
- Storage: Minimum 100 MB of free space for browser cache

- Internet Connection: Stable connection for accessing web app and Firebase services

-

B. Software Requirements

For Development Environment:

- Operating System: Windows 10 / 11, macOS, or Linux.
- Frontend Tools:
 - HTML5, CSS3, JavaScript (for interface design).
 - Visual Studio Code (IDE).
- Backend Tools:
 - Firebase Firestore (Database).
 - Firebase Authentication (Login & Role-based Access).
- Web Browser: Google Chrome, Mozilla Firefox, or Microsoft Edge.
- Version Control (optional): Git & GitHub for source code management.

For End Users:

- Operating System: Any OS (Windows, Android, macOS, iOS, Linux).
- Web Browser: Latest version of Chrome, Firefox, Edge, or Safari.
- Plugins: JavaScript enabled in browser for full functionality.

V. IMPLEMENTATION

A. Data Collection

The main objective of data collection is to obtain all relevant details that support the creation of user profiles, staff schedules, assignment records, and departmental management. The collected data serves as the foundation for database design and helps in implementing accurate functionalities such as authentication, real-time updates, and role-based access.

Sources of Data:

User Information: Register number, name, department, role (Student/Staff/HOD), and authentication credentials.

Assignment Data: Assignment titles, descriptions, upload dates, due dates, and submission details.

Timetable Data: Staff schedules, class timings, and availability slots.

Meeting Data: Student meeting requests, approval or rejection status, and notification details.

System Data: Log entries for notifications, updates, and activity tracking.

B. Component Design

The component design of the UniLink – Staff Availability and Assignment System defines how the system's major modules and submodules interact to perform various academic management operations. The system is divided into multiple components, each responsible for a specific functionality such as user authentication, timetable management, assignment handling, meeting coordination, and notifications.

Each component is designed to ensure modularity, reusability, and efficiency, while maintaining a clear separation between the frontend, backend, and database layers. This design approach enhances maintainability and simplifies future system upgrades.

Main Components of UniLink:

1) Login and Authentication Component

- Handles user verification and secure access to the system.
- Authenticates credentials using Firebase Authentication.
- Supports three user roles: Student, Staff, and HOD (Admin).
- Redirects users to their respective dashboards after successful login.
- Ensures protection against unauthorized access.

2) User Dashboard Component

- Displays personalized information based on the user's role.
- Students: View staff list, assignments, and meeting requests.
- Staff: Manage availability, upload assignments, and approve meetings.
- HOD: Monitor all users, view department activity, and manage profiles.

3) Staff Availability Component:

- Maintains and displays each staff member's current status (Available, In-Class, or Busy).
- Updates automatically based on the timetable and manual input from staff.
- Helps students know whether staff are free before sending meeting requests.

4) Timetable Management Component:

- Stores and displays weekly schedules of staff.
- Uses color codes for subjects, labs, and breaks for visual clarity.
- Allows HODs to modify or update timetables as needed.
- Integrates with the staff availability system to update status dynamically.

5) Assignment Management Component:

- Enables staff to create, upload, and manage assignments for different subjects.
- Allows students to view and submit assignments digitally before the deadline.
- Stores all assignment data (title, description, file links, and due dates) in Firebase.
- Sends automatic notifications when new assignments are uploaded or updated.

6) Meeting Request Component:

- Allows students to send meeting requests to staff for guidance or discussion.
- Staff can accept, reject, or reschedule meetings directly through the platform.
- Notifications are generated instantly to inform both parties.
- Reduces face-to-face scheduling issues and enhances communication.

7) Notification Component:

- Sends real-time notifications for assignment updates, meeting confirmations, and timetable changes.
- Uses Firebase real-time features for instant alerts across users.
- Keeps students and staff informed about important academic events.

8) Admin (HOD) Management Component:

- Provides complete control to the HOD to manage system users.
- Allows adding, editing, or deleting staff and student profiles.
- Displays overall system reports and departmental activities.
- Ensures smooth supervision and coordination among all users.

C. Software Description

The UniLink – Staff Availability and Assignment System is a web-based application developed using modern web technologies and cloud infrastructure to improve communication and coordination within a college environment. The system is built using HTML, CSS, and JavaScript for the frontend and Firebase Firestore for the backend database. This combination provides a robust, scalable, and real-time platform that supports multi-user access and seamless synchronization.

Software Overview

The UniLink system is divided into three main software layers — Frontend, Backend, and Database. Each layer works together to ensure an efficient, secure, and user-friendly experience.

1) Frontend (User Interface):

- The frontend is designed using HTML, CSS, and JavaScript to provide an interactive and responsive user experience.
- It includes modules for login, dashboards, staff availability, assignments, and meeting management.
- The interface adapts to various screen sizes (desktop, tablet, and mobile).
- HTML5 is used to structure web pages, CSS3 for layout and design, and JavaScript for functionality and dynamic behavior.
- The frontend connects directly to Firebase using the Firebase SDK for data retrieval and updates.

Tools and Technologies Used

- HTML5 – Structure of the web pages
- CSS3 – Styling and layout customization
- JavaScript – Dynamic interaction and logic handling
- Visual Studio Code – Development environment
- Firebase SDK – Frontend connectivity with backend services

2) Backend (Application Logic Layer):

- The backend functionality is managed using Firebase, which handles authentication, data storage, and notifications.
- The system uses Firebase Authentication for secure role-based login (Student, Staff, and HOD).
- Firebase Firestore stores all real-time data such as user details, assignments, and meeting information.
- Firebase Hosting is used to deploy the web application securely on the cloud.
- The backend manages core operations like user authentication, assignment submission, and meeting scheduling without the need for a traditional server setup.

Key Features of the Backend:

- Real-time data updates across all users
- High scalability with cloud infrastructure
- Secure login and user verification
- Automatic data synchronization
- Serverless architecture for reduced maintenance

3) Database (Cloud Data Management):

- Firestore Database (a NoSQL database by Firebase) is used for storing and managing data in collections and documents.
- Each user, assignment, and meeting is stored as a document under a specific collection, ensuring quick retrieval and structured organization.
- Data is automatically synchronized in real-time, allowing instant reflection of changes across all connected users.
- The database is fully managed by Firebase, ensuring security, reliability, and scalability.

Database Collections Include:

- Users: Stores profile information (Name, Role, Department, etc.).
- Assignments: Details of uploaded and submitted assignments.
- Meetings: Records of meeting requests and approval status.
- Notifications: Alerts and updates sent to users.

D. Result

The UniLink – Staff Availability and Assignment System was thoroughly tested using various user roles—Student, Staff, and HOD—to evaluate its performance, accuracy, and usability. The results demonstrated that the system successfully simplifies academic communication, provides real-time staff availability updates, and efficiently manages assignments and meetings. It ensures accurate data synchronization and smooth operation across all connected users.

Sample Test Case 1 Input:

User Role: Student

Action: Login using Register Number and Date of Birth Request: View staff availability and send meeting request Predicted Output:

- i. Staff list displayed with real-time availability status (Available / In Class / Busy)
- ii. Option to send meeting request visible for available staff
- iii. Meeting notification sent instantly to the selected staff member
- iv. Confirmation alert displayed for successful request submission

Sample Test Case 2 Input:

User Role: Staff

Action: Upload new assignment for 3rd Year CSE students

Predicted Output:

- v. Assignment created successfully and stored in Firebase database
- vi. Students of the respective class notified automatically
- vii. Assignment visible in student dashboard under “Pending Submissions”
- viii. Staff can track submitted and pending assignments in real time

Sample Test Case 3 Input:

User Role: HOD

Action: Add new staff member and update timetable

Predicted Output:

- ix. New staff record created in Firebase “Users” collection
- x. Timetable updated and visible to respective staff and students
- xi. HOD dashboard displays updated department data instantly
- xii. All changes automatically synchronized across the system

Output Pages

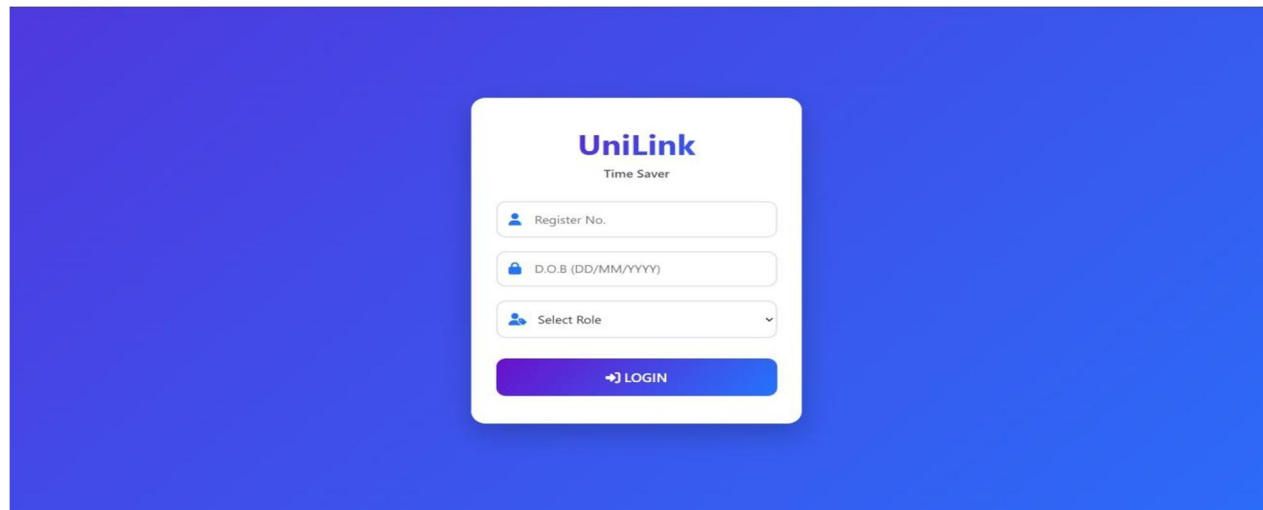


Figure 5.1: Login Page

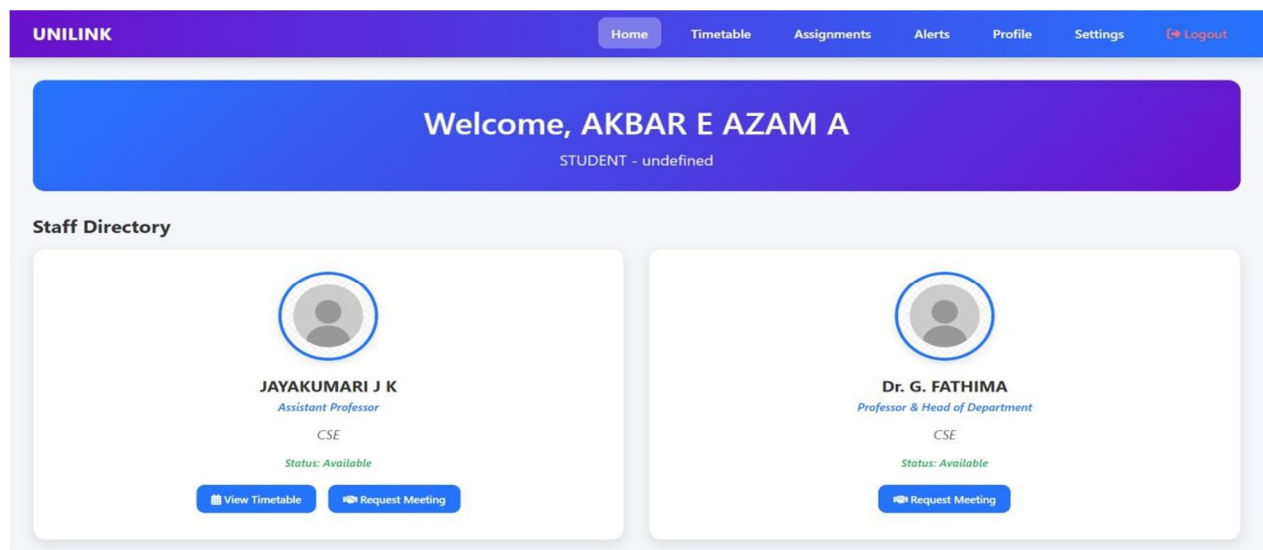


Figure 5.2: Home Page

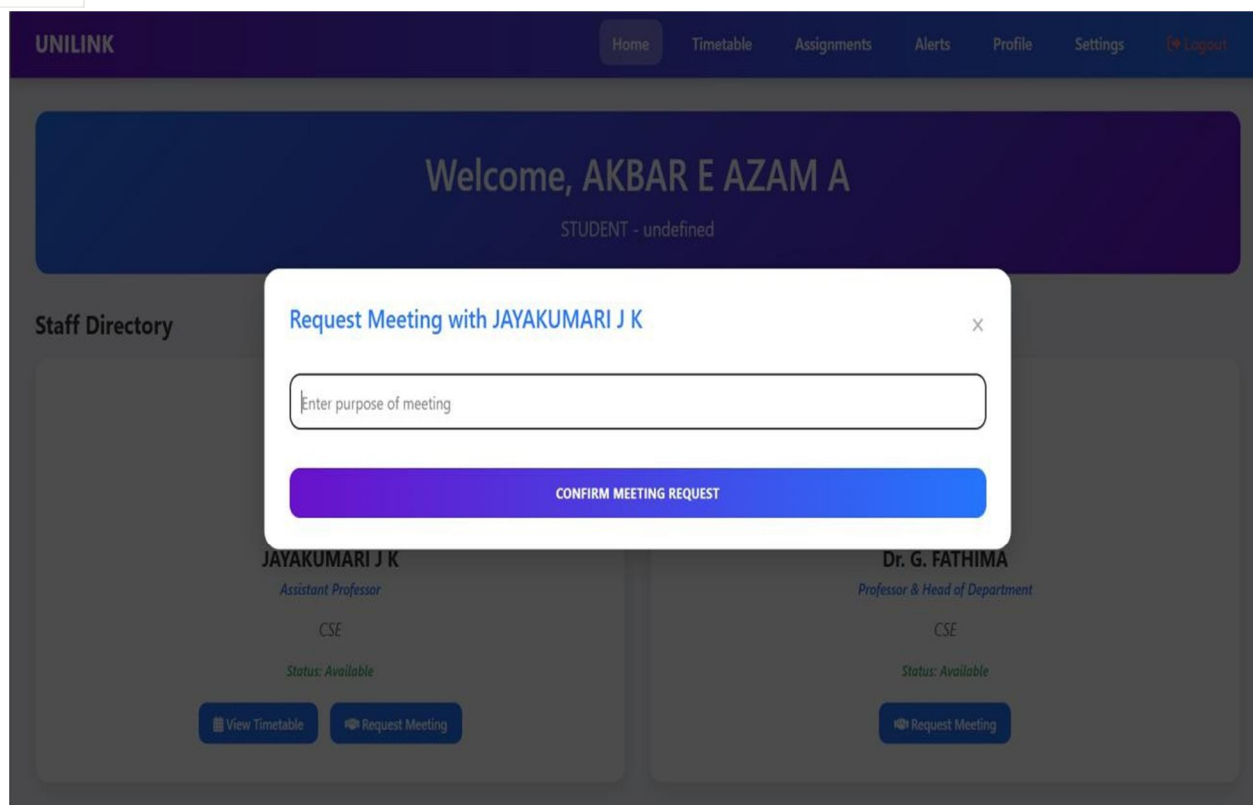


Figure 5.3: Student Meet Request Section

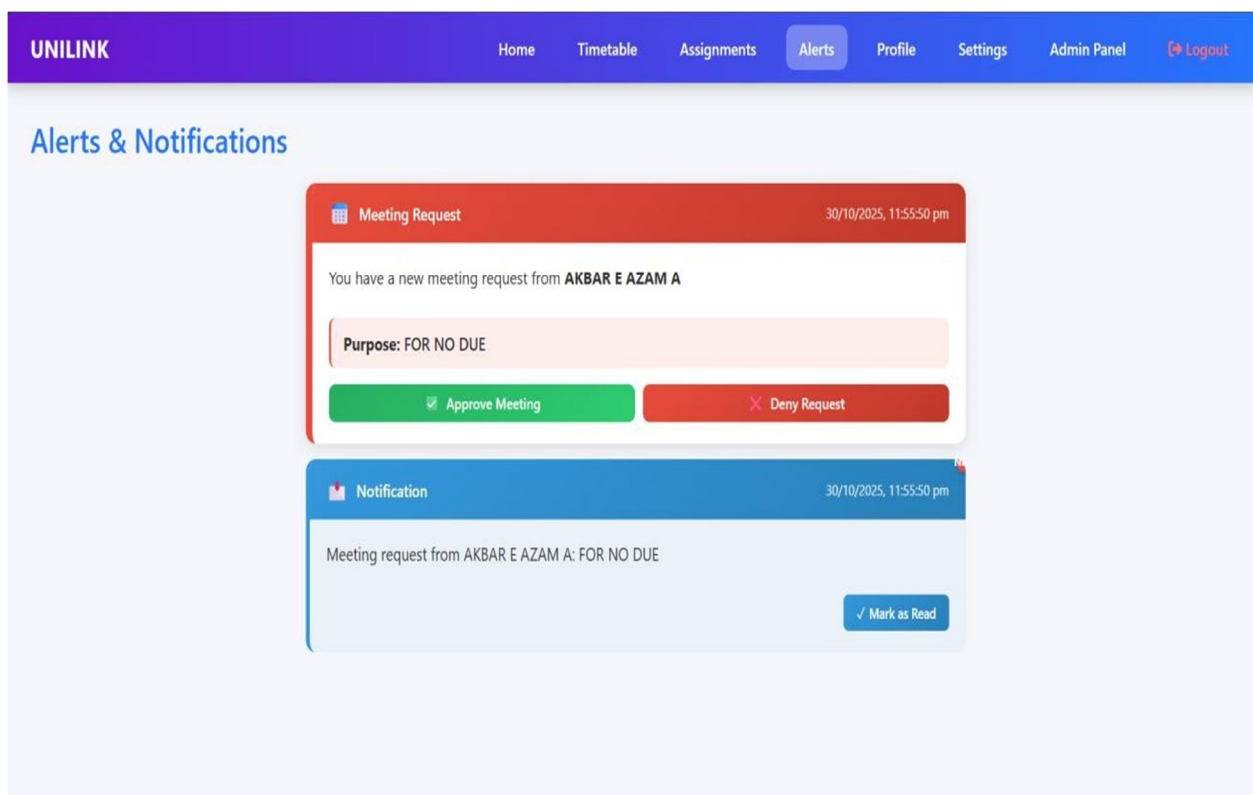


Figure 5.4: Staff Alert Panel

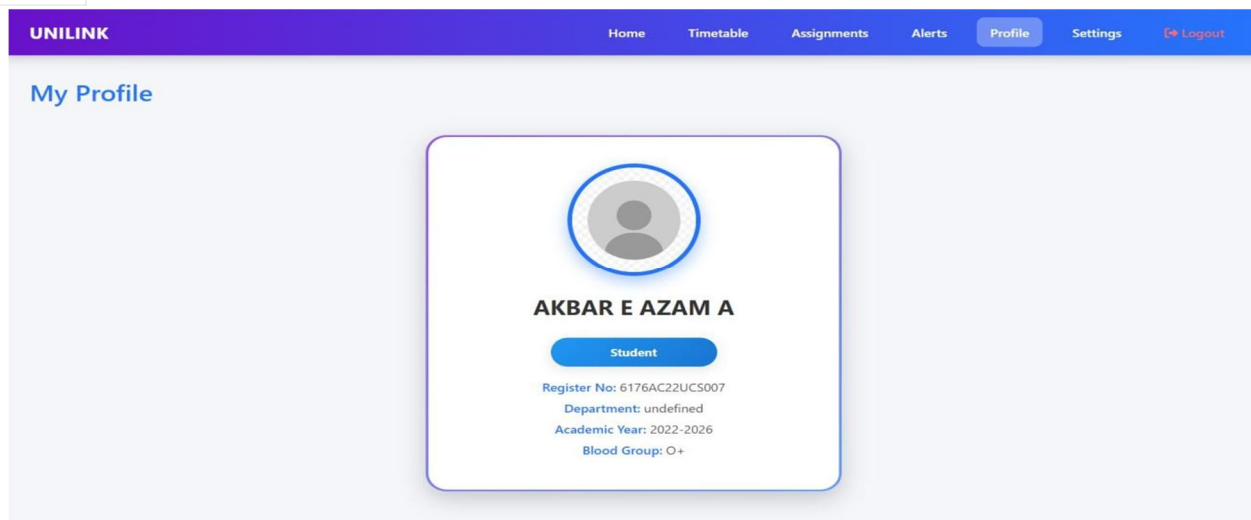


Figure 5.5: Profile Section

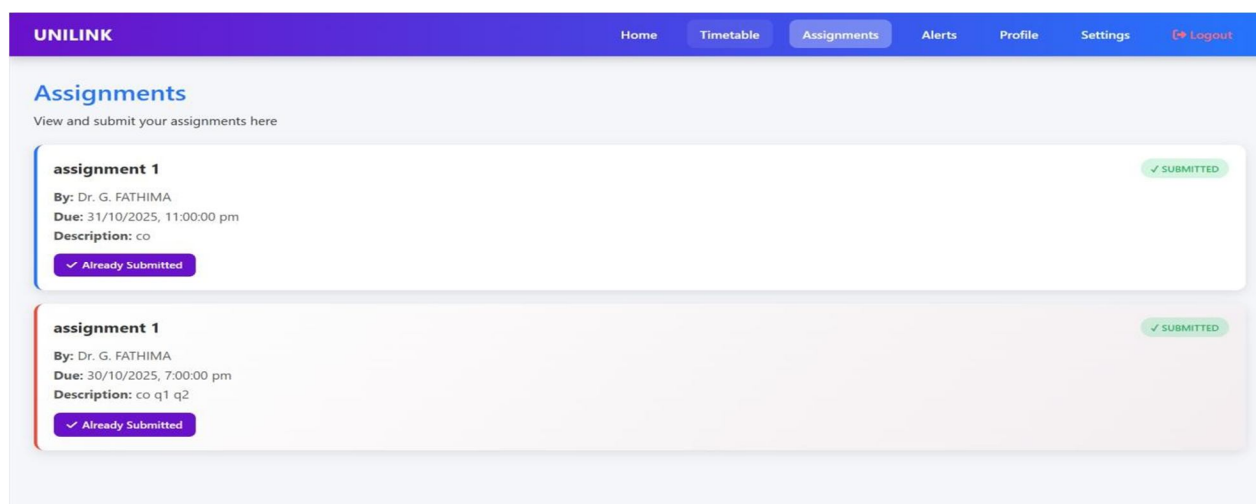


Figure 5.6: Assignment Panel

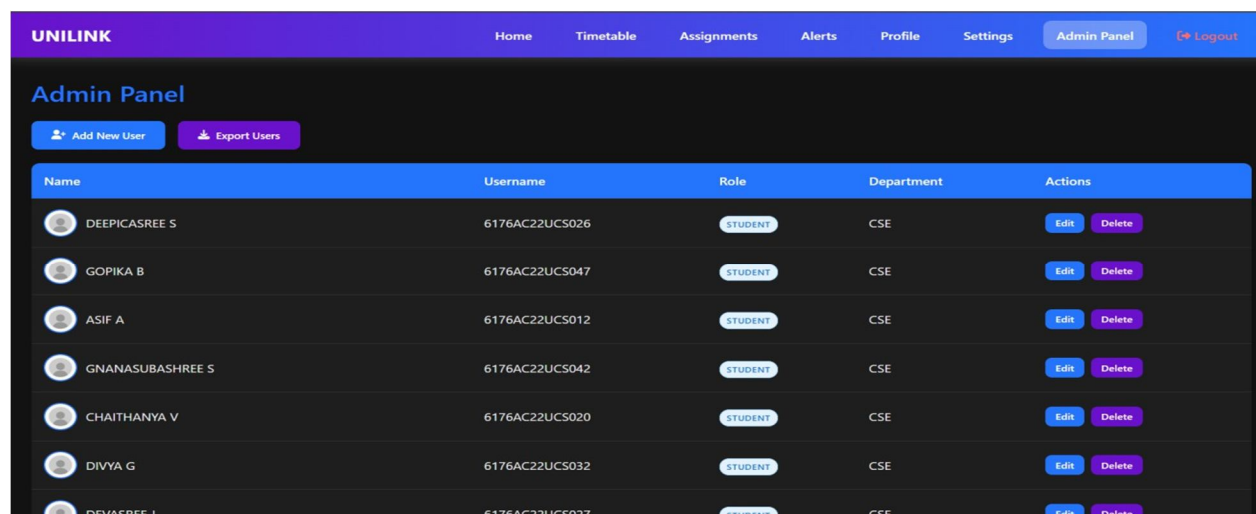


Figure 5.7: HOD Admin Panel

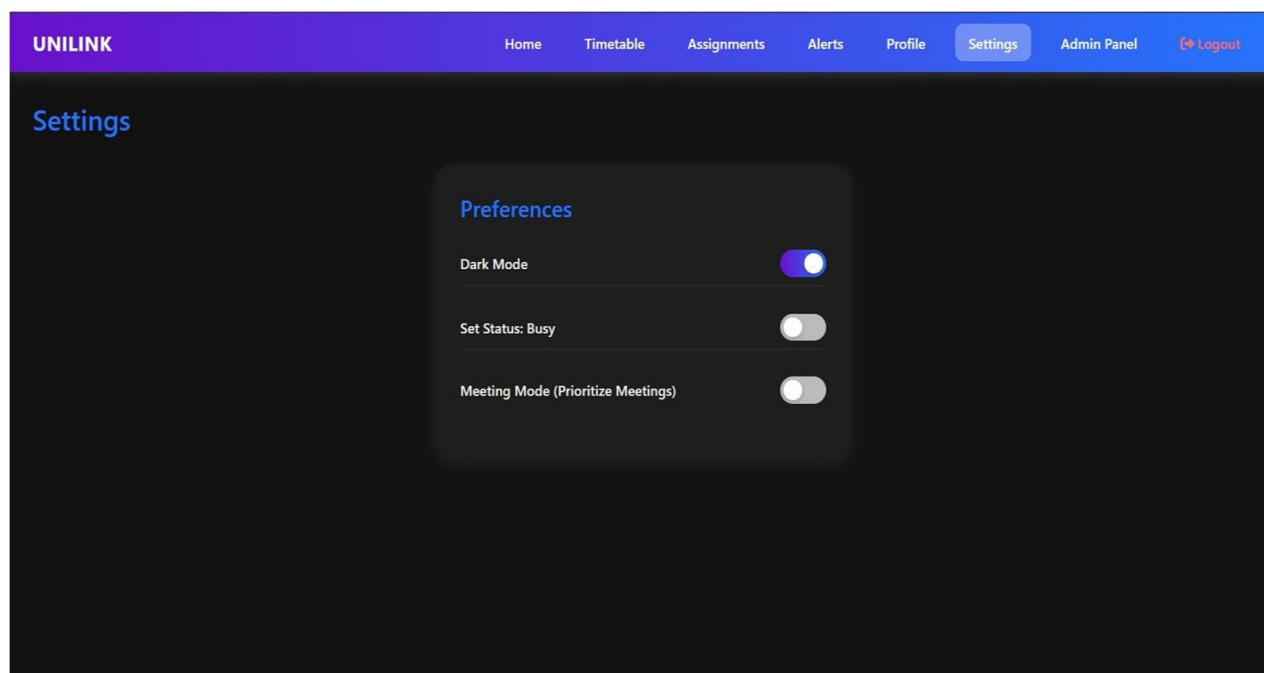


Figure 5.8: Settings Panel

VI. CONCLUSION AND FUTURE ENHANCEMENT

A. Conclusion

The UniLink – Staff Availability and Assignment System successfully achieves its objective of providing a centralized digital platform for effective communication and coordination among students, staff, and the Head of Department. The system eliminates the inefficiencies of traditional manual methods by integrating all essential academic operations such as staff availability tracking, assignment management, meeting requests, and real-time notifications into a single, user- friendly web interface.

Through the implementation of Firebase Firestore as the backend database, the system ensures real-time synchronization, secure data handling, and cloud-based accessibility. The HTML, CSS, and JavaScript technologies used for the frontend enhance the usability and responsiveness of the application, making it accessible across devices like desktops, laptops, and smartphones.

Testing results have shown that UniLink operates efficiently, with accurate data updates and seamless user interaction. It has simplified the workload for staff members, improved transparency for students, and provided administrative control for the HOD. By automating key academic processes, the system not only saves time but also promotes a more organized and technology-driven academic environment.

By integrating Firebase Firestore as the backend, the system achieves real-time data synchronization, secure storage, and cloud accessibility without the need for complex server management. The use of HTML, CSS, and JavaScript in the frontend ensures a responsive, user-friendly interface that is compatible across various devices and screen sizes.

In conclusion, UniLink serves as a reliable, efficient, and scalable solution for modern educational institutions, effectively bridging the gap between staff and students while enhancing productivity and digital transformation within the campus.

B. Future Scope

The UniLink – Staff Availability and Assignment System has been designed with scalability and flexibility in mind, allowing for future improvements and the addition of advanced functionalities. Although the current version efficiently manages staff availability, assignment distribution, and meeting coordination, there are several opportunities to enhance the system's overall performance, usability, and automation through future development.

The future scope of UniLink focuses on making the system more intelligent, integrated, and adaptable to the evolving needs of modern educational institutions.

Possible Future Enhancements

1) Integration of Attendance Management:

- Add an attendance tracking module for both students and staff.
- Automatically link attendance data with class schedules and staff availability.

2) AI-Based Timetable Generation:

- Implement artificial intelligence to generate and optimize timetables automatically.
- Detect and prevent scheduling conflicts between staff and classrooms.

3) Mobile Application Development:

- Create a dedicated Android and iOS app version for easier access and real-time notifications.
- Include push notifications for instant alerts about assignments, meetings, and updates.

4) Performance Analytics Dashboard:

- Introduce graphical analytics to evaluate staff and student performance.
- Provide visual charts for assignment completion rates, meeting trends, and system activity logs.

5) Chat and Communication Module:

- Add an in-app chat feature for direct communication between students and staff.
- Include group discussions or department-wide announcements.

6) Automated Reminder and Alert System:

- Implement scheduled reminders for assignment due dates, meeting timings, and academic deadlines.
- Support customizable notification preferences for each user.

7) Cloud Backup and Export Options:

- Enable automatic data backup and export to external cloud storage such as Google Drive.
- Provide download options for reports, assignments, and attendance summaries.

8) Multi-Institution Support:

- Expand the system to support multiple colleges or departments under one unified portal.
- Include super-admin access for institutional-level management.

9) Enhanced Security and Role Management:

- Strengthen authentication with OTP or biometric verification for sensitive operations.
- Provide more granular access control for department-specific roles.

10) Integration with Learning Management Systems (LMS):

- Connect UniLink with popular LMS platforms like Moodle or Google Classroom.
- Allow seamless transfer of assignment data and grades between systems.

APPENDICES

Appendix – A: Firebase Configuration:

This section contains the configuration details used to connect the web application to the Firebase backend.

- Firebase SDK initialization script.
- API Key and Project ID setup.
- Authentication and Firestore database integration code snippet.

Example Code (Firebase Setup): `const firebaseConfig = { apiKey: "Your_API_Key",
authDomain: "yourapp.firebaseio.com", projectId: "yourapp",`

```
storageBucket: "yourapp.appspot.com",  
messagingSenderId: "123456789",  
appId: "1:123456789:web:abcdef123456", measurementId: "123456789"  
};  
//Initialize firebase  
Firebase.initializeApp(firebaseConfig);  
Const db = firebase.firestore();  
Const auth = firebase.auth();
```

Appendix – B: Database Structure:

The UniLink database is built using Firebase Firestore (NoSQL) and consists of several collections and documents.

Collections Used:

- Users – Stores user details (name, role, department, year, etc.)
- Assignments – Holds assignment data (title, description, upload date, due date, submitted list)
- Meetings – Contains meeting requests and approval status between students and staff
- Notifications – Manages alerts for meetings, assignments, and timetable changes

Appendix – C: System Interfaces:

Screenshots and UI pages of the UniLink web application include:

- Login Page – Role-based authentication (Student, Staff, HOD)
- Dashboard – Displays user information and role-based features
- Staff Availability Page – Shows real-time status of all faculty members
- Assignments Module – Allows upload and submission of assignments
- Meeting Request Panel – For scheduling and approval of meetings
- Admin (HOD) Panel – Manage user accounts and department activities
- Settings Page – Contains light/dark mode and profile customization options

SOURCE CODE:

App.html:

```
<!DOCTYPE html>  
<html lang="en">  
<head>  
<meta charset="UTF-8">  
<meta name="viewport" content="width=device-width, initial-scale=1.0">  
<title>UniLink - Time Saver</title>  
<link rel="stylesheet" href="style.css">  
<!-- Firebase SDK -->  
<script src="https://www.gstatic.com/firebasejs/9.22.0/firebase-app- compat.js"></script>  
<script src="https://www.gstatic.com/firebasejs/9.22.0/firebase-firestore- compat.js"></script>  
<!-- Font Awesome -->  
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font- awesome/6.4.0/css/all.min.css">  
</head>  
<body>  
<!-- LOGIN PAGE -->  
<div id="loginPage" class="login-page">  
<div class="login-box">  
<div class="logo-container">  
<h1>UniLink</h1>  
<p>Time Saver</p>  
</div>  
<form id="loginForm">  
<div class="input-group">
```



```
<i class="fas fa-user"></i>
<input type="text" id="username" placeholder="Register No." required>
</div>
<div class="input-group">
<i class="fas fa-lock"></i>
<input type="password" id="password" placeholder="D.O.B (DD/MM/YYYY)" required>
</div>
<div class="input-group">
<i class="fas fa-user-tag"></i>
<select id="role" required>
<option value="" disabled selected>Select Role</option>
<option value="student">Student</option>
<option value="staff">Staff</option>
<option value="hod">HOD</option>
</select>
</div>
<button type="submit" id="loginBtn">
<i class="fas fa-sign-in-alt"></i> LOGIN
</button>
</form>
<p id="loginError" class="error">Invalid Register No. or DOB!</p>
</div>
</div>
```

```
<!-- APP PAGE -->
<div id="appPage" class="hidden">
<nav class="navbar">
<div class="logo">UNILINK</div>
<ul id="navList"></ul>
</nav>
<main>
<!-- Home -->
<section id="homePage" class="page">
<div class="welcome-section">
<h1 id="welcomeTitle">Welcome!</h1>
<p id="roleInfo">Please wait...</p>
</div>
<h2>Staff Directory</h2>
<div id="staffContainer" class="staff-container"></div>
</section>
<!-- Timetable -->
<section id="timetablePage" class="page hidden">
<h1>My Timetable</h1>
<div id="timetableContainer"></div>
</section>
<!-- Assignments -->
<section id="assignmentsPage" class="page hidden">
<h1>Assignments</h1>
<!-- Staff View -->
<div id="staffAssignmentsView" class="hidden">
```

```
<div class="assignments-controls">
<button id="createAssignmentBtn" class="btn-primary">
<i class="fas fa-plus"></i> Create New Assignment
</button>
<div class="filter-controls">
<select id="classFilter">
<option value="">All Classes</option>
<option value="I CSE A">I CSE A</option>
<option value="I CSE B">I CSE B</option>
<option value="I CSE C">I CSE C</option>
<option value="II CSE A">II CSE A</option>
<option value="II CSE B">II CSE B</option>
<option value="II CSE C">II CSE C</option>
<option value="III CSE A">III CSE A</option>
<option value="III CSE B">III CSE B</option>
<option value="III CSE C">III CSE C</option>
<option value="IV CSE A">IV CSE A</option>
<option value="IV CSE B">IV CSE B</option>
<option value="IV CSE C">IV CSE C</option>
</select>
</div>
</div>
```

```
<div id="staffAssignmentsContainer" class="assignments-container"></div>
</div>
<!-- Student View -->
<div id="studentAssignmentsView" class="hidden">
<div class="assignments-info">
<p>View and submit your assignments here</p>
</div>
<div id="studentAssignmentsContainer" class="assignments- container"></div>
</div>
```

```
</section>
<!-- Alerts -->
```

```
<section id="alertPage" class="page hidden">
```

```
<h1>Alerts & Notifications</h1>
<div id="alertsContainer" class="alerts-container"></div>
```

```
</section>
<!-- Profile -->
```

```
<section id="profilePage" class="page hidden">
<h1>My Profile</h1>
<div id="profileContainer"></div>
</section>
<!-- Settings -->
<section id="settingsPage" class="page hidden">
<h1>Settings</h1>
<div class="settings-box">
<h2>Preferences</h2>
<label> Dark Mode
<span class="toggle">
```



```
<input type="checkbox" id="themeSwitch">
<span class="slider"></span>
</span>
</label>
<label>
Set Status: Busy
<span class="toggle">
<input type="checkbox" id="busySwitch">
<span class="slider"></span>
</span>
</label>
<label>
Meeting Mode (Prioritize Meetings)
<span class="toggle">
<input type="checkbox" id="meetingSwitch">
<span class="slider"></span>
</span>
</label>
</div>
</section>
<!-- Admin Panel (HOD Only) -->
<section id="adminPage" class="page hidden">
<h1>Admin Panel</h1>
<div class="admin-controls">
<button onClick="adminPanel.addUser()" class="btn-primary">
<i class="fas fa-user-plus"></i> Add New User
</button>
<button onClick="adminPanel.exportUsers()" class="btn-secondary">
<i class="fas fa-download"></i> Export Users
</button>
</div>
<div class="users-table-container">
<table id="adminUsersTable" class="admin-table">
<thead>
<tr>
<th>Name</th>
<th>Username</th>
<th>Role</th>
<th>Department</th>
<th>Actions</th>
</tr>
</thead>
<tbody></tbody>
</table>
</div>
</section>
</main>
</div>
<!-- Timetable Modal -->
```

```

<div id="modal" class="modal">
<div class="modal-content compact-timetable">
<span id="closeBtn" class="close">&times;</span>
<h2 id="staffName"></h2>
<div id="modalTimetableContainer"></div>
</div>
</div>
<!-- Meeting Modal -->
<div id="meetModal" class="modal">
<div class="modal-content">
<span id="meetCloseBtn" class="close">&times;</span>
<h2 id="meetStaffName">Request Meeting</h2>
<input type="text" id="meetingPurpose" placeholder="Enter purpose of meeting" required>
<button id="confirmMeet">Confirm Meeting Request</button>
</div>
</div>
<!-- Edit User Modal (for Admin Panel) -->
<div id="editUserModal" class="modal">
<div class="modal-content">
<span id="editUserCloseBtn" class="close">&times;</span>
<h2>Edit User</h2>
<div id="editUserName" style="margin-bottom: 20px; font-weight: 600; color:
var(--primary-color);"></div>
<div style="margin-bottom: 15px;">
<label style="display: block; margin-bottom: 8px; font-weight: 500; color: var(--text-primary);">Name:</label>
<input type="text" id="editUserNameInput" placeholder="Enter user name">
</div>
<div style="margin-bottom: 15px;">
<label style="display: block; margin-bottom: 8px; font-weight: 500; color: var(--text-primary);">Username:</label>
<input type="text" id="editUserUsernameInput" placeholder="Enter username">
</div>
<div style="margin-bottom: 15px;">
<label style="display: block; margin-bottom: 8px; font-weight: 500; color: var(--text-primary);">Role:</label>
<select id="editUserRoleSelect">
<option value="student">Student</option>
<option value="staff">Staff</option>
<option value="hod">HOD</option>
</select>
</div>
<div style="margin-bottom: 20px;">
<label style="display: block; margin-bottom: 8px; font-weight: 500; color: var(--text-primary);">Department:</label>
<input type="text" id="editUserDeptInput" placeholder="Enter department">
</div>
<div style="display: flex; gap: 10px;">
<button id="saveUserChanges" class="btn-primary" style="flex: 1;">Save Changes</button>
<button onclick="adminPanel.closeEditModal()" class="btn-secondary" style="flex: 1;">Cancel</button>
</div>
</div>
</div>
<!-- Add User Modal (for Admin Panel) -->

```

```

<div id="addUserModal" class="modal">
<div class="modal-content">
<span id="addUserCloseBtn" class="close">&times;</span>
<h2>Add New User</h2>
<div style="margin-bottom: 15px;">
<label style="display: block; margin-bottom: 8px; font-weight: 500; color: var(--text-primary);">Name:</label>
<input type="text" id="addUserNameInput" placeholder="Enter full name">
</div>
<div style="margin-bottom: 15px;">
<label style="display: block; margin-bottom: 8px; font-weight: 500; color: var(--text-primary);">Username:</label>
<input type="text" id="addUserUsernameInput" placeholder="Enter username">
</div>
<div style="margin-bottom: 15px;">
<label style="display: block; margin-bottom: 8px; font-weight: 500; color:
var(--text-primary);">Password:</label>
<input type="password" id="addUserPasswordInput" placeholder="Enter password">
</div>
<div style="margin-bottom: 15px;">
<label style="display: block; margin-bottom: 8px; font-weight: 500; color: var(--text-primary);">Role:</label>
<select id="addUserRoleSelect">
<option value="student">Student</option>
<option value="staff">Staff</option>
<option value="hod">HOD</option>
</select>
</div>
<div style="margin-bottom: 15px;">
<label style="display: block; margin-bottom: 8px; font-weight: 500; color: var(--text-primary);">Department:</label>
<input type="text" id="addUserDeptInput" placeholder="Enter department">
</div>
<div style="margin-bottom: 15px;">
<label style="display: block; margin-bottom: 8px; font-weight: 500; color: var(--text-primary);">Academic Year:</label>
<input type="text" id="addUserYearInput" placeholder="Enter academic year">
</div>
<div style="margin-bottom: 20px;">
<label style="display: block; margin-bottom: 8px; font-weight: 500; color:
var(--text-primary);">Blood Group:</label>
<input type="text" id="addUserBloodInput" placeholder="Enter blood group">
</div>
<div style="display: flex; gap: 10px;">
<button id="saveNewUser" class="btn-primary" style="flex: 1;">Add User</button>
<button onclick="adminPanel.closeAddModal()" class="btn-secondary" style="flex: 1;">Cancel</button>
</div>
</div>
</div>
<!-- Enhanced Edit Timetable Modal -->
<div id="editTimetableModal" class="modal">
<div class="modal-content timetable-edit-simplified">
<span id="editTimetableCloseBtn" class="close">&times;</span>

```

```

<div class="timetable-edit-toolbar">
<h3 id="editTimetableTitle">Edit Timetable</h3>
<div class="toolbar-actions">
<button id="saveTimetableBtn" class="btn-primary">
<i class="fas fa-save"></i> Save
</button>
<button id="resetTimetableBtn" class="btn-secondary">
<i class="fas fa-undo"></i> Reset
</button>
</div>
</div> <div class="quick-edit-tools">
<div class="tool-button" onclick="timetableEditor.fillAllBreaks()">
<i class="fas fa-coffee"></i> Fill Breaks
</div>
<div class="tool-button" onclick="timetableEditor.clearAllPeriods()">
<i class="fas fa-broom"></i> Clear All
</div>
<div class="tool-button" onclick="timetableEditor.autoSchedule()">
<i class="fas fa-magic"></i> Auto Schedule
</div>
<div class="tool-button" onclick="timetableEditor.showQuickHelp()">
<i class="fas fa-question-circle"></i> Help
</div>
<div class="tool-button" onclick="timetableEditor.showMergeHelp()">
<i class="fas fa-object-group"></i> Merge Help
</div>
</div><div class="timetable-edit-container">
<div id="editTimetableContainer" class="timetable-editor"></div>
</div>
</div><!-- Merge Periods Modal -->
<div id="mergeModal" class="modal">
<div class="modal-content">
<span id="mergeCloseBtn" class="close">&times;</span>
<h2>Merge Periods</h2>
<div class="merge-controls">
<div class="merge-info">
<p>Select adjacent periods to merge them into a single longer period.</p>
<div id="mergePreview" class="merge-preview"></div>
</div>
<div class="merge-options">
<label>Subject:</label>
<select id="mergeSubjectSelect">
<option value="">Select Subject</option>
</select>
<label>Class:</label>
<input type="text" id="mergeClassInput" placeholder="Enter class (e.g., III CSE B)">
<label>Custom Subject:</label>
<input type="text" id="customSubjectInput" placeholder="Or enter custom subject">
</div>

```

```

<div class="merge-actions">
<button id="confirmMerge" class="btn-primary">Merge Periods</button>
<button id="cancelMerge" class="btn-secondary">Cancel</button>
</div>
</div>
</div>
</div>
<!-- Quick Edit Popover -->
<div id="quickEditPopover" class="quick-edit-popover hidden">
<div class="popover-content">
<h4>Quick Edit Period</h4>
<select class="quick-subject-select">
<option value="">Select Subject</option>
</select>
<input type="text" class="quick-class-input" placeholder="Class (e.g., III CSE B)">
<div class="popover-actions">
<button class="btn-save-quick">Save</button>
<button class="btn-cancel-quick">Cancel</button>
</div>
</div>
</div>
<!-- Create Assignment Modal -->
<div id="createAssignmentModal" class="modal">
<div class="modal-content">
<span id="createAssignmentCloseBtn" class="close">&times;</span>
<h2>Create New Assignment</h2>
<div style="margin-bottom: 15px;">
<label style="display: block; margin-bottom: 8px; font-weight: 500;">Assignment Title:</label>
<input type="text" id="assignmentTitle" placeholder="Enter assignment title" style="width: 100%; padding: 10px; border: 1px solid #ddd; border-radius: 6px;">
</div>
<div style="margin-bottom: 15px;">
<label style="display: block; margin-bottom: 8px; font-weight: 500;">Description:</label>
<textarea id="assignmentDescription" placeholder="Enter assignment description" rows="3" style="width: 100%; padding: 10px; border: 1px solid #ddd; border-radius: 6px;"></textarea>
</div>
<div style="margin-bottom: 15px;">
<label style="display: block; margin-bottom: 8px; font-weight: 500;">Due Date:</label>
<input type="datetime-local" id="assignmentDueDate" style="width: 100%; padding: 10px; border: 1px solid #ddd; border-radius: 6px;">
</div>
<div style="margin-bottom: 15px;">
<label style="display: block; margin-bottom: 8px; font-weight: 500;">Select Class:</label>
<select id="assignmentClass" style="width: 100%; padding: 10px; border: 1px solid #ddd; border-radius: 6px;">
<option value="I CSE A">I CSE A</option>
<option value="I CSE B">I CSE B</option>
<option value="I CSE C">I CSE C</option>
<option value="II CSE A">II CSE A</option>
<option value="II CSE B">II CSE B</option>
<option value="II CSE C">II CSE C</option>

```

```

<option value="III CSE A">III CSE A</option>
<option value="III CSE B">III CSE B</option>
<option value="III CSE C">III CSE C</option>
<option value="IV CSE A">IV CSE A</option>
<option value="IV CSE B">IV CSE B</option>
<option value="IV CSE C">IV CSE C</option>
</select>
</div>
<div style="margin-bottom: 20px;">
<label style="display: block; margin-bottom: 8px; font-weight: 500;">Upload File (PDF/DOC):</label>
<input type="file" id="assignmentFile" accept=".pdf,.doc,.docx" style="width: 100%; padding: 10px;">
</div>
<div style="display: flex; gap: 10px;">
<button id="saveAssignmentBtn" class="btn-primary" style="flex: 1;">Create Assignment</button>
<button onclick="assignmentsSystem.closeCreateModal()" class="btn-secondary" style="flex: 1;">Cancel</button>
</div>
</div>
</div> <!-- Submit Assignment Modal -->
<div id="submitAssignmentModal" class="modal">
<div class="modal-content">
<span id="submitAssignmentCloseBtn" class="close">&times;</span>
<h2 id="submitAssignmentTitle">Submit Assignment</h2>
<div id="assignmentDetails" style="margin-bottom: 20px; padding: 15px;
background: #f8f9fa; border-radius: 8px;">
<!-- Assignment details will be populated here -->
</div>
<div style="margin-bottom: 20px;">
<label style="display: block; margin-bottom: 8px; font-weight: 500;">Upload Your Submission (PDF/DOC):</label>
<input type="file" id="submissionFile" accept=".pdf,.doc,.docx" style="width: 100%; padding: 10px;">
</div>
<div style="display: flex; gap: 10px;">
<button id="confirmSubmissionBtn" class="btn-primary" style="flex: 1;">Submit Assignment</button><button
onclick="assignmentsSystem.closeSubmitModal()" class="btn-secondary" style="flex: 1;">Cancel</button>
</div></div>
</div>
<!-- View Submissions Modal -->
<div id="viewSubmissionsModal" class="modal">
<div class="modal-content" style="max-width: 800px;">
<span id="viewSubmissionsCloseBtn" class="close">&times;</span>
<h2 id="submissionsAssignmentTitle">Assignment Submissions</h2>
<div class="submissions-actions">
<button id="downloadSubmittedList" class="btn-primary">
<i class="fas fa-download"></i> Download Submitted List
</button>
<button id="downloadNotSubmittedList" class="btn-secondary">
<i class="fas fa-download"></i> Download Not Submitted List
</button>
</div>
<div id="submissionsContainer" class="submissions-container">
<!-- Submissions list will be populated here -->

```

```
</div>
</div>
</div><script src="script.js"></script>
</body>
</html>
<button id="confirmSubmissionBtn" class="btn-primary" style="flex: 1;">Submit Assignment</button><button
onclick="assignmentsSystem.closeSubmitModal()" class="btn-secondary" style="flex: 1;">Cancel</button>
</div></div>
</div>
<!-- View Submissions Modal -->
<div id="viewSubmissionsModal" class="modal">
<div class="modal-content" style="max-width: 800px;">
<span id="viewSubmissionsCloseBtn" class="close">&times;</span>
<h2 id="submissionsAssignmentTitle">Assignment Submissions</h2>

<div class="submissions-actions">
<button id="downloadSubmittedList" class="btn-primary">
<i class="fas fa-download"></i> Download Submitted List
</button>
<button id="downloadNotSubmittedList" class="btn-secondary">
<i class="fas fa-download"></i> Download Not Submitted List
</button>
</div>
<div id="submissionsContainer" class="submissions-container">
<!-- Submissions list will be populated here -->
</div>
</div>
</div><script src="script.js"></script>
</body>
</html>
```

VII. ACKNOWLEDGEMENT

It is one of the most efficient tasks in life to choose the appropriate words to express one's gratitude to the beneficiaries. We are very much grateful to God who helped us all the way through the project and how molded us into what we are today.

We are grateful to our beloved Principal Dr. R. RADHAKRISHNAN, M.E., Ph.D., Adhiyamaan College of Engineering (An Autonomous Institution), Hosur for providing the opportunity to do this work in premises.

We acknowledge our heartfelt gratitude to Dr. G. FATHIMA, M.E., Ph.D., Professor and Head of the Department, Department of Computer Science and Engineering, Adhiyamaan College of Engineering (An Autonomous Institution), Hosur, for her guidance and valuable suggestions and encouragement throughout this project and made us to complete this project successfully.

We are highly indebted to Mrs. C. AKSHAYA, M.E., Supervisor, Assistant Professor, Department of Computer Science and Engineering, Adhiyamaan College of Engineering (An Autonomous Institution), Hosur, whose immense support encouragement and valuable guidance were responsible to complete the project successfully.

We also extend our thanks to Project Coordinator and all Staff Members for their support in complete this project successfully.

Finally, we would like to thank to our parents, without their motivational and support would not have been possible for us to complete this project successfully.

REFERENCES

- [1] Herbert Schildt, "JavaScript: The Complete Reference," McGraw-Hill Education, 2020.
- [2] Steven Holzner, "HTML, CSS, JavaScript & jQuery All in One," Pearson Education, 2019.
- [3] Kogent Learning Solutions Inc., "Web Technologies: HTML, CSS, JavaScript," Dreamtech Press, 2018.
- [4] Ramesh Bangia, "Web Technology and Design," Firewall Media, 2017.
- [5] Gaurav Gupta, "Cloud Computing & Firebase Database Concepts," TechKnowledge Publications, 2021.

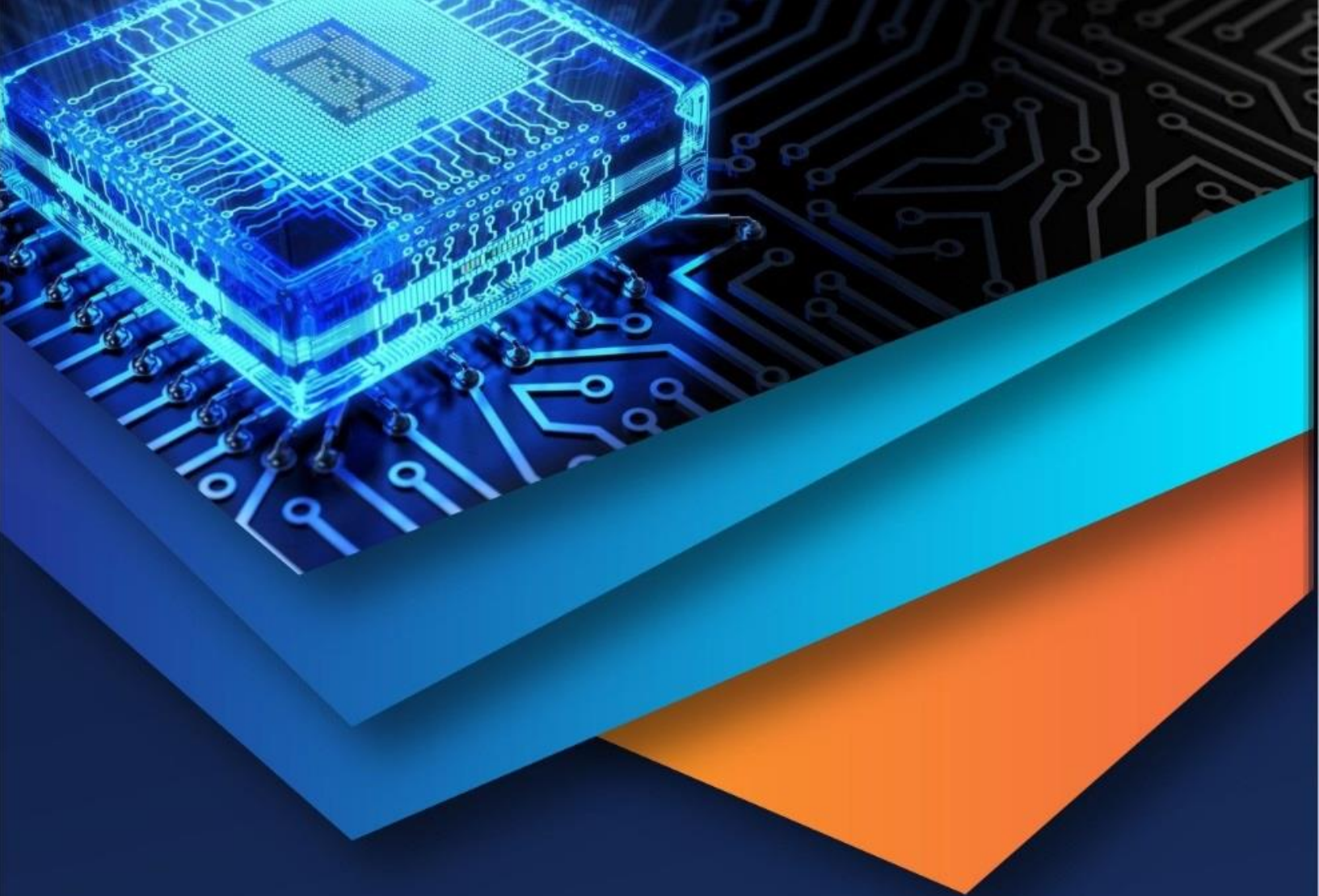


Research Articles & Technical Papers:

“Real-Time Database Applications using Firebase for Cloud-Based Systems,” IEEE Xplore Digital Library, 2021.

Design and Implementation of Web-Based Academic Management Systems,” International Journal of Computer Applications (IJCA), Vol. 183, 2022.

“Role-Based Authentication for Secure Educational Portals,” International Research Journal of Engineering and Technology (IRJET), Vol. 9, Issue 5, 2023.



10.22214/IJRASET



45.98



IMPACT FACTOR:
7.129



IMPACT FACTOR:
7.429



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089  (24*7 Support on Whatsapp)