Green Intelligent Systems and Applications

Research Article
Volume 3(2), 2023, 69–85
https://doi.org/10.53623/gisa.v3i2.269

Android Based College App Using Flutter Dart

Kavitha Marimuthu[1], Arunkumar Panneerselvam[2], Senthilkumar Selvaraj[1]*, Lakshmi Praba Venkatesan[1], Vetriselvi Sivaganesan[1]

[1]Department of Electronics and Communication Engineering, E.G.S. Pillay Engineering College, Nagapattinam, Tamilnadu, India, 611002.
[2]Department of Electronics and Communication Engineering, Sir Issac Newton College of Engineering and Technology, Nagapattinam, Tamilnadu, India, 611102.

*Correspondence: senthil.lanthiri@gmail.com

SUBMITTED: 23 June 2023; REVISED: 24 July 2023; ACCEPTED: 1 August 2023

ABSTRACT: In today's world, communication and information sharing between teachers and students have increasingly shifted to online platforms such as Google Classroom, Gmail, Google Forms, WhatsApp, and more. To address the diverse needs of educational institutions, we developed an app that supports all devices, including mobile phones, laptops, and tablets. The Android app for mobile and tablet websites supports all devices seamlessly. This app provides comprehensive information on attendance, examination schedules, lecture notes, fee details, event notifications, and online tests, catering to all the requirements of the institution. We developed this app using the latest technology, including Flutter and Dart, with Firebase integration. Additionally, we created a web application that is easily accessible via desktops. This website, along with the app, is connected to the same Firebase server, ensuring synchronized data access. The institute has taken a step further by developing its own Android application and website to enhance efficient communication with its students. These platforms are exclusively accessible and available to authorized users associated with the institute, ensuring privacy and security.

KEYWORDS: App for college management, android, flutter, dart

1. Introduction

In recent times, the use of mobile applications has become increasingly prevalent in various fields, including education. This work aimed to develop an Android application that simplified college management by providing an efficient platform for students and faculty to conduct academic activities seamlessly. The app was designed to help students manage their academic records, view results, communicate with their professors, and receive updates on college-related activities. Additionally, it provided a platform for faculty to monitor attendance, manage fees, and post important announcements. The system replaced traditional paper records, decreasing the paperwork and time needed to access faculty and student records.

To develop the Android app that simplified college management and streamlined academic activities, we aimed to provide an easy-to-use interface for students and faculty to access relevant information about academic activities. This app also replaced apps like Google Classroom and Google Forms. Flutter, an open-source mobile SDK, was used to build native-looking Android and iOS applications from the same code base. Flutter had been around since 2015 when Google introduced it and remained in the beta stage before its official launch in December 2018. Since then, the buzz around Flutter had been growing stronger.

The central idea behind Flutter was the use of widgets. Developers could build the entire UI by combining different widgets. Each of these widgets defined a structural element (like a button or menu), a stylistic element (a font or color scheme), a layout aspect (like padding), and many others. Flutter also provided developers with reactive-style views. To avoid performance issues deriving from using a compiled programming language to serve as the JavaScript bridge, Flutter used Dart. It compiled Dart ahead of time (AOT) into the native code for multiple platforms. Dart was an object-oriented client-optimized programming language used to build mobile and web applications on various platforms. It had easy-to-use applications that could be utilized to work on both the user and server end. Dart employed C-style syntax and was mainly used to create front-end user interfaces for mobile applications. It was used to build high-performance mobile or web applications and was the basic programming language for the Flutter framework, used to build scalable mobile applications. Being a general-purpose programming language, it was used to build native mobile apps for Android or iOS, desktop apps, and servers.

In the past, Flutter and Dart were an excellent combination for building high-quality, performance, and scalable applications for both mobile and web platforms. Here are some of the main advantages and scopes of using Flutter and Dart for building apps and web: Cross-Platform Development: Flutter and Dart allowed developers to build applications for both mobile and web platforms using a single codebase. This enabled faster development, reduced costs, and allowed developers to easily maintain and update the app on different platforms. Flutter's hot-reload feature allowed developers to see changes in real-time, reducing development time and speeding up the debugging process. Flutter allowed developers to create highly customized and interactive user interfaces, with a wide range of pre-built widgets and the ability to easily create custom widgets.

2. Literature Survey

The Android-based College Management System, named Breeze, offered a one-stop solution for all administrative tasks. It facilitated the easy access of data to all the stakeholders of the educational institution. The app allowed administrators, authorities, faculty members, students, and guardians to get the desired data directly. This app eliminated the need for physical paperwork, which was often time-consuming and prone to errors. The development of an Android-based College Management System was a significant advancement in the education industry. It streamlined the administrative process and made the management of educational institutions more efficient [1]. An application was proposed to serve as a centralized platform for all event alerts, including those on the website and physical notice boards placed in the college. By providing a single point of access to all event-related information, this application would streamline the process of event management and communication in the college [2]. An Android-based College Student Information application was developed that could effectively manage college management activities. The application had a powerful data management system and a user-friendly interface, making it easy to use and navigate. The main goal of the application was to reduce the amount of paperwork and time required for manual processing. As technology continued to advance, the education system in India had also evolved. This application would aid institutions in advancing more quickly, fulfilling their vision, and achieving their objectives [3]. A powerful application was suggested for a data management system and a user-friendly interface, making it easy to use and navigate. The main goal of the application was to reduce the amount of paperwork and time required for manual processing. As technology continued to advance, the education system in India had also evolved. This application would aid institutions in advancing more quickly, fulfilling their vision, and achieving their objectives [4].

Arun Sai S et.al presented a mobile application for the College Management System based on the Android platform and Firebase Database. The primary aim of this user-friendly application was to address the challenges faced by students in colleges and universities. The application had been tested on Android handsets running version 5 and above, and the results were promising. This application was reliable, saved time, and was simple to use. It allowed students and their parents to check their grades, attendance, and curricular information in real-time [5]. A cutting-edge cloud-based mobile application was developed that aimed to address the challenges that colleges or institutes faced when it came to providing centralized information to their students [6]. College Management Android Application (CMAA) provided a simple interface for the maintenance of college information. The admin, faculty, or the Student should have been a registered user. The Student information system dealt with all kinds of student details, academic-related reports, college details, course details, curriculum, and other resource-related details too. It also had faculty details, batch execution details, students' details in all aspects, the various academic notifications to the staff and students updated by the college administration [7]. The mobile application could perform various functions such as monitoring attendance of the students, disseminating information like notices, e-books, lab manuals, assignments, and event-related information and photos. College Administration and Management System (CAMS) maintained records of students and educators. This application provided login credentials to users of the system, and with the help of credentials, users could access as well as modify data as per the permissions given to them by the admin of the system. CAMS was an intranet-based application that aimed at providing and maintaining information [8]. An application provided accurate information at all times as faculty members or students needed. The college management could make useful decisions using the data that were stored in the university database server. So it was better to have an Android-Based College Management system like Pillai HOC app. All the administrators, authorities, faculty, students, and guardians would get the desired data directly. The proposed app was much better than existing Android-based applications or web-based applications in terms of features of the application, requirements of the students, parents, and public, technology, and design pattern used in the design and development of the application, usability of web pages, and ranking of web pages [9]. An easy-to-use mobile application was offered that allowed event organizers to add all the necessary details of an event, including registration information, and participants could register for events with ease. This feature ensured that participants had access to all event information and could register efficiently. Additionally, the project included information about all the clubs and student chapters at VNR VJIET [10].

3. Application for College Events

The existing College Management System application provided an autonomous solution for paper-based work. It was controlled and monitored by the admin, and it reduced manpower usage. It provided accurate information at all times as faculty members or students needed. The college management could make useful decisions using the data that were stored in the university database server. Therefore, having an Android-based College Management System like Breeze was preferred. All the administrators, authorities, faculty, students, and guardians would get the desired data directly. In the traditional approach, faculty members used bulletin boards to share any data, and teachers dictated notes to the students, who also wrote notes in their notebooks. This method was time-consuming for both teachers and students. In case of any imperative notices by higher management like the board or principal, it took time to reach everyone. This might have caused some issues for students.The proposed app of the College Management System provided an autonomous solution for paper-based work. It was controlled and monitored by the admin, and it reduced manpower usage. It provided accurate information at all times as faculty members or students needed. The college management could make useful decisions using the data that were stored in the university database server. Therefore, having an Android-Based College Management system like the Pillai HOC app was recommended. All the administrators, authorities, faculty, students, and guardians would get the desired data directly. The proposed app was much better than the existing Android-based application or the web-based application in terms of features, requirements of the students, parents, and the public, technology, design pattern used in the design and development of the application, usability of web pages, and ranking of web pages.

4. App and Web Modules Front-End

4.1. Login page.

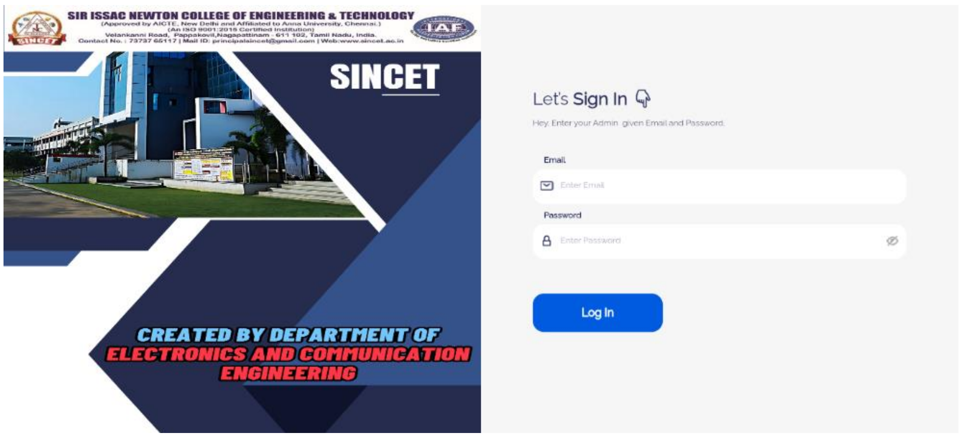

SIR ISSAC NEWTON COLLEGE OF ENGINEERING & TECHNOLOGY
(Approved by AICTE, New Delhi and Affiliated to Anna University, Chennai.)
(An ISO 9001:2015 Certified Institution)
Velankanni Road, Pappakovil,Nagapattinam - 611 102, Tamil Nadu, India.
Contact No. : 73737 65117 | Mail ID: principalsincet@gmail.com | Web:www.sincet.ac.in
SINCET
CREATED BY DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
Let's Sign In
Hey, Enter your Admin given Email and Password.
Email
Enter Email
Password
Enter Password
Log In

Figure 1. Log in page of the website.

From the code you provided, it appears to be a login page created using the Flutter framework and Firebase authentication. The page had a simple UI with two text fields for the user to input their email and password, along with a button to submit the login form. When the user clicked the "LOGIN" button, the entered email and password were validated, and if they were valid, an attempt to authenticate the user was made using Firebase authentication. If the authentication was successful, the user was redirected to the "home page" page. The user data was stored in the Firebase authentication system, which managed the authentication process and user accounts. When a user signed up for the first time, their email and password were stored in the Firebase authentication system, along with a unique user ID. This ID could be used to identify the user and retrieve their authentication information.

Figure 1 showed the login page of the website, and Figure 2 represented the login page of the application. Additionally, Firebase authentication provided tools for developers to monitor user activity and access logs, so they could see who logged in and when. The exact details of these logs and monitoring tools depended on how they were implemented by the developer.

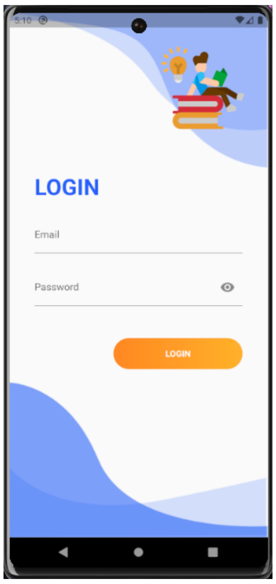


Figure 2. Log in page of the application.

4.2. Home screen

Mobile and Web app home screen with several functionalities. Here is a brief summary of what the home screen provides:

4.2.1. App bar.

The app bar has a customized toolbar with a height of 100 pixels, and it displays the user's name and rolls number. The app bar is positioned at the top of the screen.

4.2.2. Navigation drawer.

The end Drawer attribute contains a drawer widget with various menu options.

4.2.3. Body.

The body of the home screen has a padding of 8 pixels on all sides. The body contains a column with several widgets.

4.2.4. Grid view.

The column contains a grid view with eight items. The grid view has a fixed cross-axis count of 2. The eight items in the grid view are:

4.2.5. Attendance

The attendance widget has a conditional statement that allows students to view their attendance sheet and teachers to mark attendance. Figure 3 illustrates the website home screen.

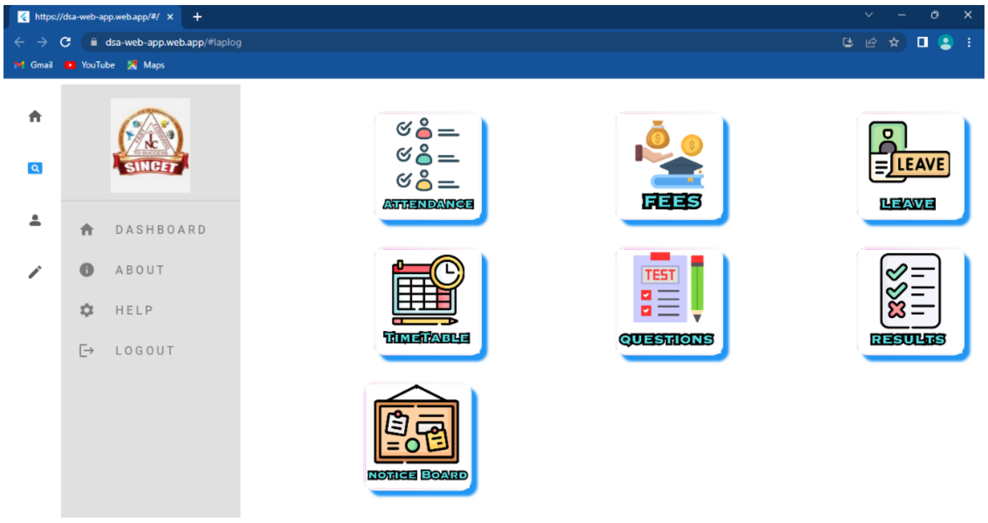


Figure 3. Website home screen.

4.3. END drawer.

The END Drawer widget was a navigation drawer that displayed four menu items: Dashboard, represented by an icon of a home. When the user tapped on it, they were navigated to the 'home screen' route, and all previous routes were removed from the stack. In the Settings, this menu item was represented by an icon of a gear. When the user tapped on it, they were navigated to the 'setting' route. About, this menu item was represented by an icon of an 'i' in a circle. Logout, this menu item was represented by an icon of a door with an arrow pointing out. When the user tapped on it, they were signed out of the app using the auth variable and then navigated to the 'moblog' route, replacing all previous routes in the stack. Figure 4 represented the website with the opened drawer.

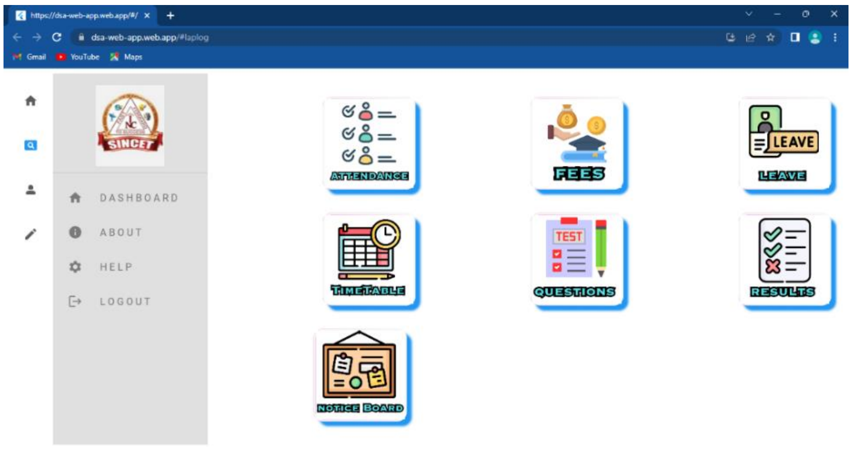


Figure 4. Website opened drawer.

The Drawer Header widget displayed an icon of a heart and was always present at the top of the drawer. The menu items were styled using the tile Padding and drawer text color variables, which were defined elsewhere in the code. Overall, the end Drawer widget provided a functional and visually appealing navigation drawer for a mobile app. Fees: The fees widget had a conditional statement that allowed students to view their fee status, and teachers could manage student fees. Leave: The leave widget allowed students to apply for a leave of absence, and teachers could approve or reject leave applications. Time table: The time table widget had a conditional statement that allowed students to view their class schedules, and teachers could manage the class schedules. Daily test: The daily test widget allowed students to view their daily test questions. Results: The results widget allowed students to view their examination results. Noticeboard: The noticeboard widget had a conditional statement that allowed students to view notices, and teachers could post notices. Bottom navigation: The home screen had a bottom navigation widget with four menu options: home, settings, profile, and circular.

4.4. Bottom navigation bar.

The Bottom Navigation Bar widget was a built-in widget in Flutter that provided a simple way to display a horizontal navigation bar at the bottom of the screen. The navigation bar consisted of multiple items like home, online test, college, profile, add user, etc., each containing an icon and a label. The current index property was used to track the currently selected item, and the on Tap callback was called when an item was tapped. The initState method initialized the user data with the current user using Firebase Auth and retrieved the user data from Firestore to display the user's name and role. Flutter Bottom Navigation Bar was a UI widget that allowed users to easily switch between different views or screens within a single app by tapping on a tab at the bottom of the screen. It was a common design pattern used in mobile app development to provide easy navigation and improve the user experience. The bottom navigation bar of admin and non-admin is shown in Figure 5.

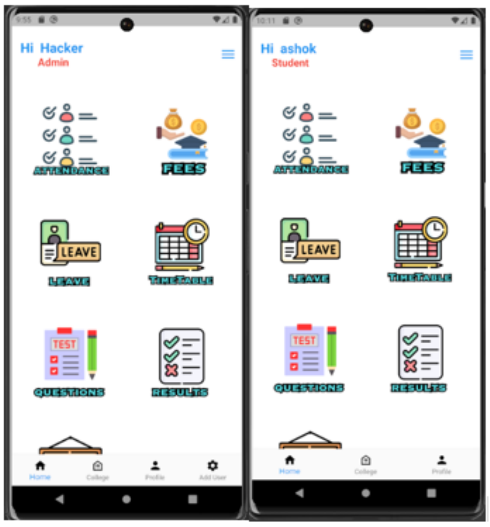


Figure 5. Bottom navigation bar of admin and non admin.

The Flutter website navigation rail is a navigation component used on the Flutter website to help users easily navigate between different sections of the site. Itis similar to the navigation rail described earlier, with a list of destinations represented by icons and labels. However, instead of being vertical, the Flutter website navigation rail is horizontal and located at the top of the page. The navigation rail on the Flutter website had six destinations: Get Started, Documentation, Community, Showcase, Blog, and More. Each destination was represented by an icon and a label. The website navigation rail was shown in Figure 6. The Get Started destination had a rocket icon, the Documentation destination had a book icon, the Community destination had a people icon, the Showcase destination had a camera icon, the Blog destination had a pen icon, and the More destination had a menu icon. By default, the Get Started destination was selected, and when the user selected another destination, the site navigated to the corresponding page.

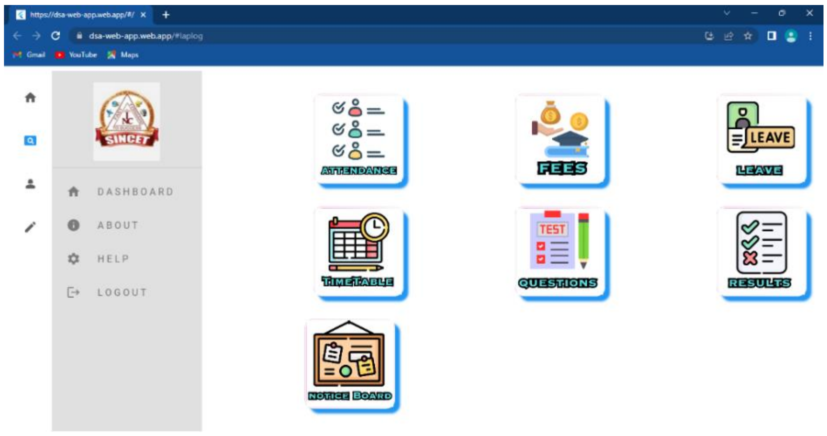


Figure 6. Website navigation rail.

5. App And Web Modules Back-End.

5.1. User authentication.

The user registration process contains fields for the user's name, email, mobile number, password, department, and role. The user's details are then validated and sent to Firebase for storage. The user is prompted to enter their details in the form fields, including selecting their department and role from dropdown menus. Once the form is filled out, and the user clicks the "Sign Up" button, their details are validated and sent to Firebase using the signup and post Details to Firestore functions. The signup function validates the user's input and attempts to create a new user with their email and password using Firebase's "create user with Email and Password" method. If successful, the function then calls post Details to Firestore to send the user's details to Firebase for storage. The post Details to Firestore function creates a new UserModel object with the user's details and sends it to Firebase using the "set" method. Once the data has been sent to Firebase, the user is redirected to the moblog page. Figure 7 illustrates the user authentication and Firebase authentication process.

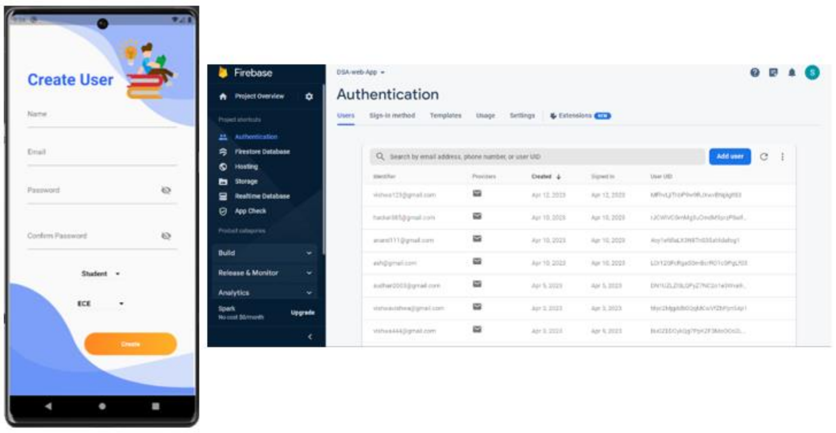
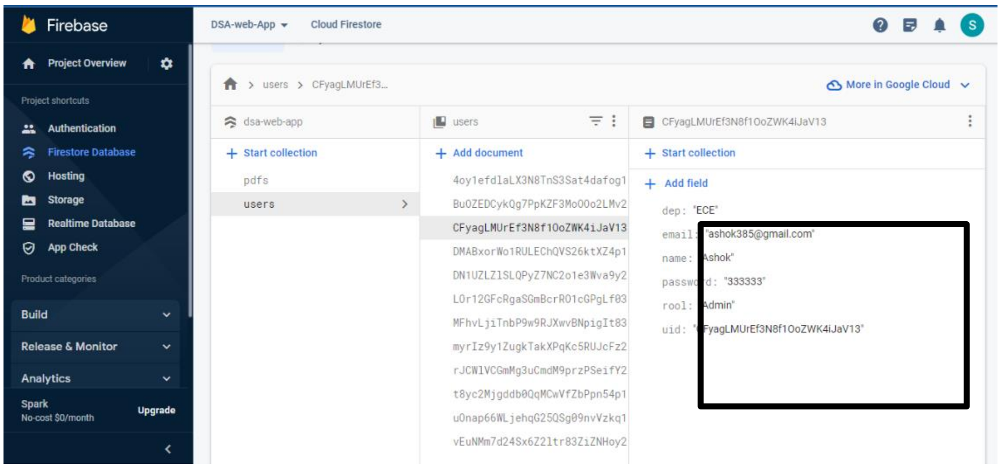


Figure 7. User authentication and firebase authentication.

5.2. Firebase data store and retrieval.

Firebase Firestore is a flexible, scalable, and highly available cloud-based NoSQL document database service that allows developers to store, synchronize, and query data for their applications. It is a part of the Firebase platform developed by Google and is designed to store and manage data for mobile, web, and server-side applications. Firestore is a schemaless database that uses collections and documents to organize data. Each document is a JSON object that contains fields and values, and each collection is a group of related documents. Firestore provides features like real-time synchronization, automatic scaling, offline support, and a powerful query engine that enables developers to efficiently retrieve and filter data. Firestore has two modes of operation: Native mode and Datastore mode. In native mode, Firestore is a standalone database service that provides real-time synchronization and offline data access. In Datastore mode, Firestore is used as a datastore backend for Google Cloud Datastore, which provides additional features like high availability, multi-region replication, and advanced indexing. Firestore is designed to work seamlessly with other Firebase services like Firebase Authentication, Cloud Functions, Cloud Messaging, and Cloud Storage. This makes it easy for developers to build scalable, real-time applications that can handle large amounts of data and users. To use Firestore in a Flutter app or web, developers can use the official FlutterFire library. FlutterFire provides a set of plugins that allow developers to access Firestore and other Firebase services from their Flutter apps. The plugins are easy to install and use, and they provide a simple API for reading and writing data to Firestore. Firestore provides a set of powerful query capabilities that allow developers to filter and sort data based on specific criteria. Developers can use queries to retrieve documents from a collection, order documents by a field, limit the number of results, and perform complex filtering operations. Firestore also provides real-time synchronization that allows developers to receive updates to their data in real-time as they are made by other users or devices. The Firebase Firestore database is shown in Figure 8.



```
dsa-web-app          users                              CFyagLMUrEf3N8f1OoZWK4iJaV13
  pdfs                 4oy1efdlaLX3N8TnS3Sat4dafog1       dep: "ECE"
  users                BuOZEDCykQg7PpKZF3MoOOo2LMv2       email: "ashok385@gmail.com"
                       CFyagLMUrEf3N8f1OoZWK4iJaV13       name: "Ashok"
                       DMABxorWo1RULEChQVS26ktXZ4p1       password: "333333"
                       DN1UZLZ1SLQPyZ7NC2o1e3Wva9y2       rool: "Admin"
                       LOr12GFcRgaSGmBcrR01cGPgLf03       uid: "CFyagLMUrEf3N8f1OoZWK4iJaV13"
                       MFhvLjiTnbP9w9RJXwvBNpigIt83
                       myrIz9y1ZugkTakXPqKc5RUJcFz2
                       rJCW1VCGmMg3uCmdM9przPSeifY2
                       t8yc2Mjgddb0QqMCwVfZbPpn54p1
                       uOnap66WLjehqG25QSg09nvVzkq1
                       vEuNMm7d24Sx6Z21tr83ZiZNHoy2
```

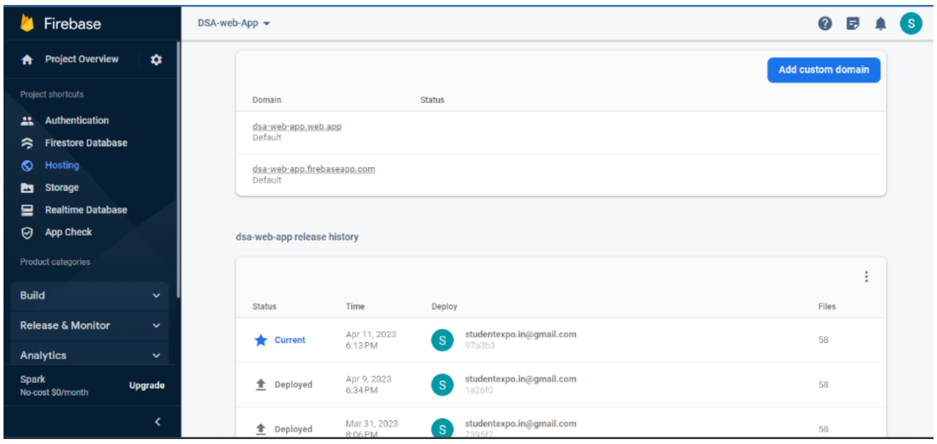
Figure 8. Firebase firestore database.

5.3. Firebase storage.

Firebase Storage is a cloud-based storage service provided by Google as a part of their Firebase suite. It allows developers to store and retrieve user-generated content like images, videos, and other media files in a secure and scalable way. It provides developers with an easy-to-use API and SDKs for various platforms, including Flutter and web, making it a popular choice for mobile and web app development. Firebase Storage provides a simple and efficient way to store files in the cloud, which can be accessed from anywhere with an internet connection. It is designed to scale automatically to meet the needs of the app and can store files of any size. The storage service also provides different security features, like access control rules, that can be used to restrict access to certain files or folders. One of the benefits of using Firebase Storage is that it integrates seamlessly with other Firebase services, like Firebase Authentication and Firebase Cloud Functions, allowing you to build a full-stack app with ease. For example, you can use Firebase Authentication to authenticate your users and Firebase Cloud Functions to process and manipulate files before storing them in Firebase Storage. Firebase Storage uses a simple bucket-based system to organize files. A bucket is a container for files that can be accessed using a unique URL. You can create multiple buckets for your app and use them to organize your files according to different criteria, like user or file type. To store a file in Firebase Storage, you first need to create a reference to the file's location in the bucket. You can then use this reference to upload the file from your app or web client. The uploaded file will be automatically assigned a unique URL that you can use to retrieve the file later. Firebase Storage also provides built-in support for resumable uploads, which can be useful when uploading large files that may be interrupted by network issues or other factors. Resumable uploads allow you to pick up where you left off if an upload is interrupted, rather than starting the upload from the beginning. Firebase Storage also integrates with Firebase Analytics, allowing you to track how users interact with your stored files. For example, you can track how many times a file has been downloaded or which files are most popular among your users. Overall, Firebase Storage is a powerful and flexible cloud storage solution that provides developers with a simple and efficient way to store and retrieve files in their apps. Its integration with other Firebase services and its ease of use make it a popular choice for mobile and web app developers who want to focus on building great user experiences without having to worry about the complexities of managing their own server infrastructure.

5.4. Firebase web hosting.

Firebase Hosting is a fast and secure hosting service that allows developers to deploy static and dynamic web content to a global content delivery network (CDN). Firebase Hosting provides several benefits to developers, including scalability, fast content delivery, security, and easy deployment with custom domain support. It supports multiple types of content, including HTML, CSS, JavaScript, and images. Additionally, Firebase Hosting allows for the deployment of dynamic content through Cloud Functions or other serverless services. The platform offers a simple command-line interface for deployment, enabling developers to deploy their content with a single command. Moreover, Firebase Hosting seamlessly integrates with popular web development tools like GitHub and GitLab, streamlining the process of deploying code changes. Figure 9 represents Firebase web hosting.



Domain	Status
dsa-web-app.web.app Default	
dsa-web-app.firebaseapp.com Default	

dsa-web-app release history

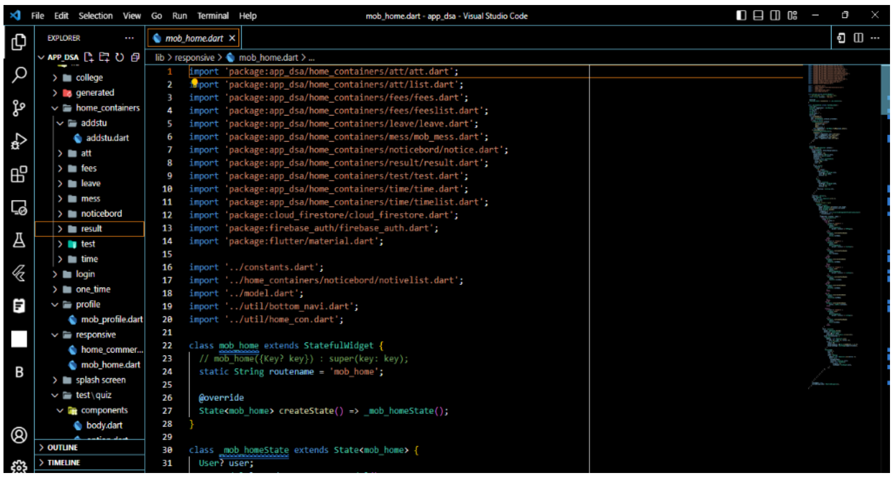
Status	Time	Deploy	Files
Current	Apr 11, 2023 6:13 PM	studentexpo.in@gmail.com 97a3b3	58
Deployed	Apr 9, 2023 6:34 PM	studentexpo.in@gmail.com 1e26f0	58
Deployed	Mar 31, 2023 8:06 PM	studentexpo.in@gmail.com 7395f7	58

Figure 9. Firebase web hosting.

Firebase Hosting has a pay-as-you-go pricing model, which means developers only pay for the resources they use. Additionally, it provides a free tier, offering hosting for small-scale applications with low traffic. Apart from web hosting, Firebase offers a wide range of other services for application development, including authentication, real-time database, cloud storage, cloud messaging, and more. With SDKs available for Android, iOS, and web, developers can easily integrate Firebase services into their applications. Overall, Firebase Hosting is a reliable and scalable hosting service, providing developers with a simple and user-friendly platform for deploying their web content. Its seamless integration with other Firebase services and support for custom domains make it an excellent choice for developers seeking a fast and secure hosting solution. Web hosting, on the other hand, is a service that allows individuals and organizations to publish their website or web application on the internet. By signing up for a web hosting service, you essentially rent space on a server where your website's files will be stored and made accessible to internet users.

5.5. Visual studio code.

Visual Studio Code is a popular, free, and open-source code editor developed by Microsoft. It is designed to be lightweight yet powerful, offering a wide range of features and extensions to boost developers' productivity. VS Code has gained popularity among developers because of its cross-platform compatibility, user-friendly interface, and support for various programming languages and frameworks, including Flutter. Flutter is a mobile app SDK (Software Development Kit) that enables developers to build high-performance, high-fidelity apps for iOS, Android, web, and desktop platforms. It is an open-source framework created by Google and uses the Dart programming language. Flutter comes with a rich set of pre-built widgets, tools, and libraries that simplify the process of creating impressive UIs and powerful applications. The Flutter extension is included in VS Code by default, but it can also be separately installed from the VS Code Marketplace. Visual Studio Code proves to be an excellent tool for developing Flutter apps. Its built-in Flutter extension offers a wide range of features and tools specifically tailored for Flutter development, streamlining the process of creating top-notch apps. Thanks to its ease of use, cross-platform compatibility, and support for various programming languages and frameworks, VS Code has become an essential tool for any Flutter developer. You can find a visual representation of the proposed model using Visual Studio Code in Figure 10.


mob_home.dart - app_dsa - Visual Studio Code
mob_home.dart
lib > responsive > mob_home.dart > ...

```dart
import 'package:app_dsa/home_containers/att/att.dart';
import 'package:app_dsa/home_containers/att/list.dart';
import 'package:app_dsa/home_containers/fees/fees.dart';
import 'package:app_dsa/home_containers/fees/feeslist.dart';
import 'package:app_dsa/home_containers/leave/leave.dart';
import 'package:app_dsa/home_containers/mess/mob_mess.dart';
import 'package:app_dsa/home_containers/noticebord/notice.dart';
import 'package:app_dsa/home_containers/result/result.dart';
import 'package:app_dsa/home_containers/test/test.dart';
import 'package:app_dsa/home_containers/time/time.dart';
import 'package:app_dsa/home_containers/time/timelist.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';

import '../constants.dart';
import '../home_containers/noticebord/notivelist.dart';
import '../model.dart';
import '../util/bottom_navi.dart';
import '../util/home_con.dart';

class mob_home extends StatefulWidget {
  // mob_home({Key? key}) : super(key: key);
  static String routename = 'mob_home';

  @override
  State<mob_home> createState() => _mob_homeState();
}

class _mob_homeState extends State<mob_home> {
  User? user;
```

Figure 10. Visual studio code tool.

5.6. Android studio emulator.

The Android Studio Emulator is a robust tool that empowers developers to test and debug their applications on virtual devices with diverse configurations, screen sizes, and Android versions. This tool is especially beneficial for mobile app developers who aim to create applications for Android devices but lack access to physical Android devices for testing. Flutter, on the other hand, is a widely-used open-source framework designed for developing high-performance, cross-platform mobile applications. It utilizes a reactive programming model and offers a wide range of pre-built widgets, making it convenient for developers to swiftly create stunning, native-quality user interfaces. Figure 11 displays the Android Studio Emulator, showcasing its capabilities and providing a glimpse of the virtual device testing environment available to developers using this tool.

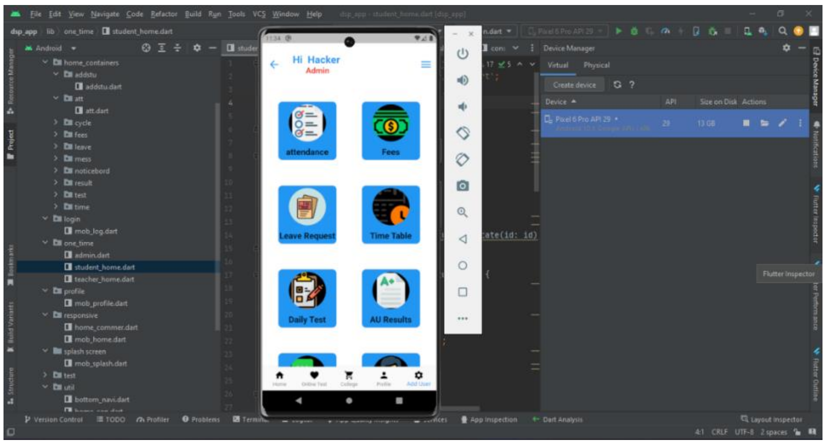


Figure 11. Android studio emulator.

The Android Studio Emulator, when combined with Flutter, provides a seamless way for developers to test and debug their Flutter applications on virtual devices, eliminating the need for physical devices. In this article, we will delve into how the Android Studio Emulator can be effectively utilized with Flutter to streamline the process of building, testing, and deploying mobile applications. One of the major advantages of employing the Android Studio Emulator with Flutter is the ability to test applications on a diverse array of screen sizes and resolutions. This is crucial since applications must appear visually appealing and function flawlessly across various devices with distinct screen sizes and resolutions. To accomplish this, developers can generate virtual devices with varying configurations using the AVD Manager. By launching each virtual device, they can thoroughly assess their applications on different screen sizes and resolutions. The Android emulator is developed using a combination of virtualization and emulation technologies. It leverages the QEMU (Quick Emulator), an open-source machine emulator, to establish a virtual machine that replicates the hardware and software environment of an Android device. This emulation process enables developers to simulate the actual behavior of Android devices and thoroughly evaluate the performance of their applications. With the Android Studio Emulator and Flutter in tandem, developers can confidently fine-tune their mobile applications, ensuring optimal performance and user experience on a wide range of virtual devices, all without the need for physical devices. This synergy proves to be a valuable asset for developers seeking efficiency and reliability in their app development journey.

6. App User Role Management

6.1. Student.

Role students can easily access essential information and services through the use of a mobile application, providing them with convenience and efficiency. With just a few clicks on their mobile devices, students can access their department's attendance and fees details, timetable, and notice board updates. Moreover, the app enables students to request leave permission from faculty members, eliminating the need for physical paperwork and expediting the process. Additionally, the app facilitates timely preparation for tests, as students can receive tomorrow's test questions through it. Students can also submit assignments through the app, ensuring timely and hassle-free submissions. The app further allows students to attend online tests from their own devices, providing flexibility and maintaining academic integrity. Lastly, the app provides access to the university website, offering students additional resources and information.

6.2. Professor.

For professors, the educational software applications serve as valuable tools to manage and oversee their students' academic progress. These applications enable professors to efficiently handle attendance and fee details, timetable, and notice board updates for their department. Professors can upload and edit attendance and fee information, ensuring accuracy and up-to-date records. They can also manage leave requests, reviewing and responding to them promptly. By uploading test questions, professors can ensure students' preparedness for exams. The app allows professors to view students' assignments, providing insights into their progress and offering targeted support and feedback. Furthermore, professors can conduct online tests, simplifying the exam administration process and providing timely feedback to students. These applications enhance efficiency, accuracy, and access to student information, facilitating a more engaging and supportive learning experience. This allows professors to provide targeted feedback and support, helping students achieve their academic goals. Finally, professors can conduct online tests for their students through the app, making it easier to administer exams and assess student progress. Online testing allows professors to provide immediate feedback to students, reducing the time required to grade exams and providing students with timely information on their performance. Overall, these educational software applications offer a range of benefits to professors, including improved efficiency, accuracy, and access to student information. By leveraging technology, professors can better manage their academic responsibilities and provide students with a more engaging and supportive learning experience. Following Figure 12 represents the overall flow chart of the proposed system.

6.3. Admin.

As an Admin, one of the primary responsibilities is user authentication, ensuring only authorized individuals access the app to maintain its privacy and security. The Admin plays a vital role in uploading and editing college circulars, ensuring that vital information reaches the users accurately and on time. With exclusive access to view all pages and options related to content management, the Admin efficiently manages the app's functionality. Moreover, the Admin has the authority to add or remove users, controlling access and enhancing security and privacy measures for the app.

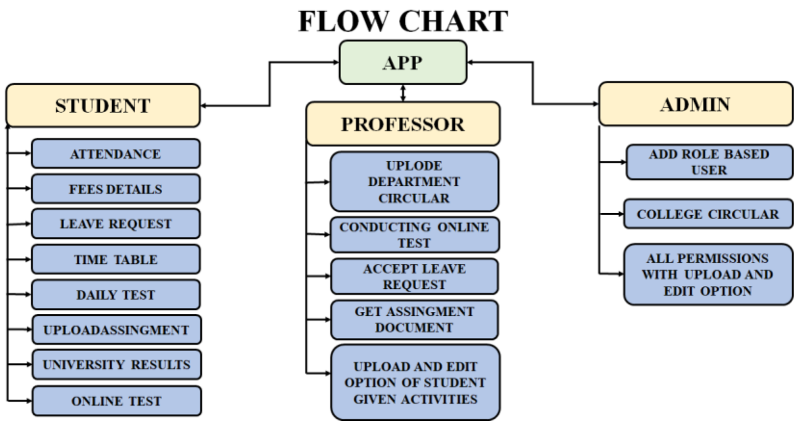


Figure 12. Flow chart of role management.

7. Testing

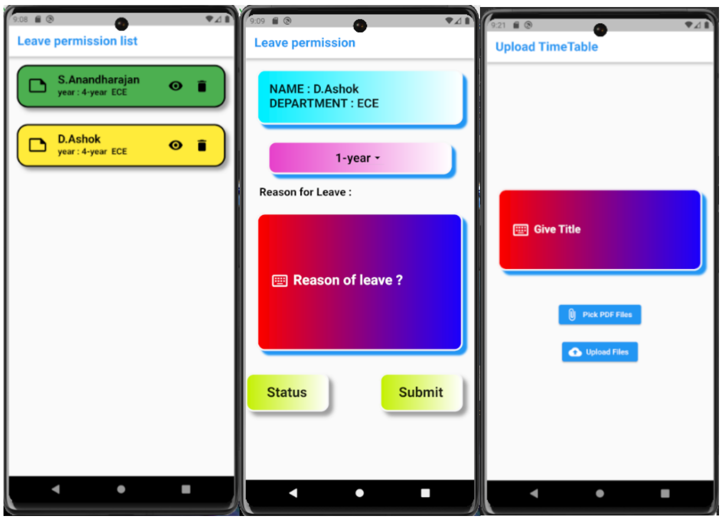
Different types of testing are used in this article for proper execution of the proposed application.

– Unit Testing: Test individual components or modules of the app and website in isolation.
– Each unit works as expected and meets its functional requirements. Testing frameworks are used to automate unit tests.
– Integration Testing: Here the interactions between different components are tested when integrated into the entire system. It is verified from this test that data flow and communication between modules are smooth and accurate.
– Functional Testing: Test the functional requirements of the app and website against specified use cases and user stories are done in this test. All features, such as leave request, attendance view, fees view, etc., work correctly are correctly ensured.
– Usability Testing: Evaluation of the user interface and user experience to ensure the app and website are user-friendly are takes place in this test. Obtain feedback from representative users to identify usability issues and make improvements also considered.
– User Acceptance Testing (UAT): It allows end-users (students, faculty, and administrators) to test the system in a real-world environment. The app and website meet the users' needs and expectations are ensured.

– Performance Testing: Evaluation the app and website's performance under different conditions (e.g., varying user loads) are tested here. Measurement of response times, loading times, and overall system efficiency also done in this test.
– Security Testing: Assess the system's security measures to identify vulnerabilities and potential threats. Perform penetration testing to check for weaknesses in the app and website.
– Compatibility Testing: Testing the app and website on various devices, operating systems, and browsers are taken. Compatibility and responsiveness across different platforms are ensured.
– Regression Testing: This test is conducted to verify that new changes or updates do not negatively impact existing functionalities. Re-test previously working features to prevent regression issues.
– Accessibility Testing: Check that the app and website are accessible to users with disabilities. Comply with accessibility standards like WCAG (Web Content Accessibility Guidelines).
– Database Testing: Here the integrity and accuracy of data stored in the database are validated. Test data retrieval, storage, and manipulation processes are also considered.
– Load Testing: Evaluation of the system's performance under heavy user loads are tested in this test. Also bottlenecks of the proposed application are identified and ensure the app and website can handle concurrent users.

8. Result and Discussion

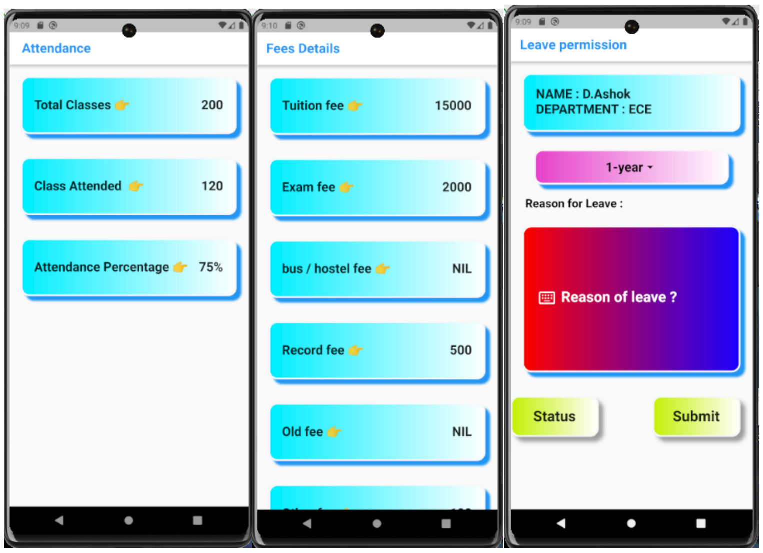
The development of the app and website for our educational institution has brought about numerous benefits, particularly in the realm of communication and information sharing. Through the use of various platforms such as Google Classroom, Gmail, Google Forms, WhatsApp, and more, we have successfully transitioned our communication channels to online platforms, adapting to the changing needs of teachers and students in today's digital age. The app, specifically designed for Android devices, including mobile phones and tablets, offers a user-friendly interface that makes it convenient for students to access comprehensive information. With features like attendance tracking, examination schedules, lecture notes, fee details, notification events, and online tests, students have a centralized hub to access crucial information related to their academic journey. The integration of Firebase technology has further elevated the functionality and connectivity of our app and website. By utilizing Flutter and Dart programming languages, our development team has ensured compatibility across multiple devices and operating systems, providing users with a seamless experience. With both the app and website connected to the same Firebase server, data synchronization happens in real-time, enabling efficient management of information. Additionally, the web application developed alongside the app ensures accessibility via desktops, catering to the diverse needs of our students and faculty members. This multi-platform approach allows us to reach a broader audience and facilitates effective communication between teachers and students. Since implementing these platforms, communication and information dissemination within our educational institution have significantly improved. Students can easily access their attendance records, examination schedules, and lecture notes, enhancing their engagement and productivity. Faculty members, on the other hand, can efficiently share important announcements, notifications, and even conduct online tests, streamlining administrative processes. Overall, the app and website have played a crucial role in enhancing communication and information sharing within our educational institution, leading to a more connected and efficient learning environment. For a visual representation of the app from the admin and student perspectives, please refer to Figures 13 and 14 respectively.


Leave permission list
S.Anandharajan
year : 4-year ECE
D.Ashok
year : 4-year ECE

Leave permission
NAME : D.Ashok
DEPARTMENT : ECE
1-year
Reason for Leave :
Reason of leave ?
Status
Submit

Upload TimeTable
Give Title
Pick PDF Files
Upload Files

Figure 13. Admin view of Application.


Attendance
Total Classes 200
Class Attended 120
Attendance Percentage 75%

Fees Details
Tuition fee 15000
Exam fee 2000
bus / hostel fee NIL
Record fee 500
Old fee NIL

Leave permission
NAME : D.Ashok
DEPARTMENT : ECE
1-year
Reason for Leave :
Reason of leave ?
Status
Submit

Figure 14. Student view of application.

9. Conclusion and Future Scope

The development of a comprehensive app and website, integrated with the latest technology and powered by Firebase, has significantly improved the educational institution's communication with its students. Through this app and website, students can access crucial information such as attendance records, examination schedules, lecture notes, fee details, notification events, and online tests. The platforms are accessible from various devices, including mobile phones, laptops, and tablets, and are accessible exclusively to authorized users associated with the institute. This project underscores the importance of utilizing technology to enhance communication and information sharing in the education sector. The app and website have been designed to be user-friendly, with a syntax similar to other popular programming languages like Java and JavaScript. Adding new features to the app can further enhance the user experience, increase engagement, and set it apart from competitors. Continuous assessment of user needs and feedback is essential in identifying areas for improvement and ensuring that the app remains relevant and effective. Maintenance is a critical aspect of mobile app development to ensure the app's ongoing usability, stability, and security. Developers need to address various aspects of maintenance, including bug fixes, performance optimization, update compatibility, feature updates, and security updates. Optimizing app performance is of utmost importance to ensure that users can navigate the app smoothly and complete their desired actions efficiently. Regular performance optimization ensures that the app runs smoothly, quickly, and efficiently, regardless of the device used to access it.

Competing Interest

All authors have no conflicts of interest.

References

[1] Dhiman, R.; Basral, A.; Jaswanti, D. (2019). Developing a New Android App for College Management System. *International Journal of Computing, Communications and Networking (IJCCN), 08*, 88–96. http://doi.org/10.30534/ijccn/2019/017812019.

[2] Deshmukh, R.; Rajbhar, V.; Sankhe, M.; Kahlon, R.K. (2020). Android Application for College Events. *International Research Journal of Engineering and Technology, 07*, 213–220.

[3] Avhad, S.; Bade, T.; Wasnik, A.P. (2020). Design and Implementation of College Student Information using Android Application. *International Research Journal of Engineering and Technology, 07*, 155–168.

[4] Vinmathi, M.S.; Theerkasri, M.; Monica, K.; Kavitha, S. (2020). Development of a Feature Rich Android Application for College Management. *International Research Journal of Engineering and Technology, 07*, 97–114.

[5] Sai, S.A. (2021). College Management Android System using Firebase. *International Research Journal of Engineering and Technology, 08*, 61–68.

[6] Lad, P.; Karpe, A.; Konadkar, Y.; Manivannan, P. (2021). A Mobile App to Help the College Students based on Cloud Computing. *International Research Journal of Engineering and Technology, 08*, 186–194.

[7] Shwethashree, A.; Rohith, B.R.; Harshitha, K.G. (2022). College Management Android Application. *International Research Journal of Modernization in Engineering Technology and Science*, 04, 73–79.

[8] Thombre, D.V.; Rathour, A.; Jadhav, P.; Vichare, A.; Sheoran, A. (2022). College Administration & Management System. *International Research Journal of Modernization in Engineering Technology and Science, 04*, 108–114.

[9] Basatwar, R.; Patil, A.; Konadkar, Y.; Taiwade, R. (2022). College Management System. *International Journal of Advanced Research in Science, Communication and Technology, 02*, 102–111.

[10] Pabba, P.; Viswadha, B.; Srivani, P.; Rajashree, P.; Kumar, S.V. (2022). Mobile Application for College Event Management. *International Research Journal of Engineering and Technology, 09*, 41–52.