

AUTOMATIC SOCIAL MEDIA APP CREATION

A Project Report

Submitted by

AKASH KUMAR	JEC17CS012
DIVYA PETER	JEC17CS044
ELJO JOY	JEC17CS045
JACKSON JAMES	JEC17CS052

to

APJ Abdul Kalam Technological University
in partial fulfillment of the requirements for the award of the Degree of
Bachelor of Technology (B.Tech)
in
COMPUTER SCIENCE & ENGINEERING

Under the guidance of
MR. SHAIJU PAUL



CREATING TECHNOLOGY
LEADERS OF TOMORROW
ESTD 2002

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Jyothi Engineering College
NAAC Accredited College with NBA Accredited Programmes*

Approved by AICTE & affiliated to APJ Abdul Kalam Technological University

A CENTRE OF EXCELLENCE IN SCIENCE & TECHNOLOGY BY THE CATHOLIC ARCHDIOCESE OF TRICHUR

JYOTHI HILLS, VETTIKATTIRI P.O., CHERUTHURUTHY, THRISSUR, PIN-679531 PH : +91- 4884-259000, 274423 FAX : 04884-274777

NBA accredited B.Tech Programmes in Computer Science & Engineering, Electronics & Communication Engineering, Electrical & Electronics Engineering and Mechanical Engineering valid for the academic years 2016-2022. NBA accredited B.Tech Programme in Civil Engineering valid for the academic years 2019-2022.



June 2021

DECLARATION

We the undersigned hereby declare that the project report “Automatic Social Media App Creation”, submitted for partial fulfillment of the requirements for the award of degree of Bachelor of Technology of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by us under supervision of Mr. Shaiju Paul. This submission represents our ideas in our own words and where ideas or words of others have been included, we have adequately and accurately cited and referenced the original sources. We also declare that we have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in this submission. We understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously used by anybody as a basis for the award of any degree, diploma or similar title of any other University.

Name of Students	Signature
AKASH KUMAR (JEC17CS012)	
DIVYA PETER (JEC17CS044)	
ELJO JOY (JEC17CS045)	
JACKSON JAMES (JEC17CS052)	

Place:

Date:

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



CREATING TECHNOLOGY
LEADERS OF TOMORROW
ESTD 2002

CERTIFICATE

This is to certify that the report entitled “ **AUTOMATIC SOCIAL MEDIA APP CREATION** ” submitted by **AKASH KUMAR(JEC17CS012)**, **DIVYA PETER(JEC17CS044)**, **ELJO JOY(JEC17CS045)**, and **JACKSON JAMES(JEC17CS052)** to the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the Degree in Bachelor of Technology in **Computer Science & Engineering** is a bonafide record of the project work carried out by them under my/our guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.

Dr. Saju P John

Professor
Internal Supervisor & Head of the Department

ACKNOWLEDGEMENT

We take this opportunity to thank everyone who helped us profusely, for the successful completion of our project work. With prayers, we thank **God Almighty** for his grace and blessings, for without his unseen guidance, this project would have remained only in our dreams.

We thank the **Management** of Jyothi Engineering College and our Principal, **Dr. Sunny Joseph Kalayathankal** for providing all the facilities to carry out this project work. We are grateful to the Head of the Department **Dr. Saju P John** for his valuable suggestions and encouragement to carry out this project work. Our sincere thanks to **Dr. Vinith R**, former Head of the Department for permitting us to make use of the facilities available in the department to carry out the project successfully.

We would like to express our whole hearted gratitude to the project guide **Mr. Shaiju Paul** for his encouragement, support and guidance in the right direction during the entire project work.

We thank our Project Coordinators **Mr. Unnikrishnan P & Ms. Namitha T N** for their constant encouragement during the entire project work. We extend our gratefulness to all teaching and non teaching staff members who directly or indirectly involved in the successful completion of this project work.

Finally, we take this opportunity to express our gratitude to the parents for their love, care and support and also to our friends who have been constant sources of support and inspiration for completing this project work.

AKASH KUMAR (JEC17CS012)

DIVYA PETER (JEC17CS044)

ELJO JOY (JEC17CS045)

JACKSON JAMES (JEC17CS052)

VISION OF THE INSTITUTE

Creating eminent and ethical leaders through quality professional education with emphasis on holistic excellence.

MISSION OF THE INSTITUTE

- To emerge as an institution par excellence of global standards by imparting quality Engineering and other professional programmes with state-of-the-art facilities.
- To equip the students with appropriate skills for a meaningful career in the global scenario.
- To inculcate ethical values among students and ignite their passion for holistic excellence through social initiatives.
- To participate in the development of society through technology incubation, entrepreneurship and industry interaction.

VISION OF THE DEPARTMENT

Creating eminent and ethical leaders in the domain of computational sciences through quality professional education with a focus on holistic learning and excellence.

MISSION OF THE DEPARTMENT

- To create technically competent and ethically conscious graduates in the field of Computer Science & Engineering by encouraging holistic learning and excellence.
- To prepare students for careers in Industry, Academia and the Government.
- To instill Entrepreneurial Orientation and research motivation among the students of the department.
- To emerge as a leader in education in the region by encouraging teaching, learning, industry and societal connect

PROGRAMME EDUCATIONAL OBJECTIVES

PEO 1: The graduates shall have sound knowledge of Mathematics, Science, Engineering and Management to be able to offer practical software and hardware solutions for the problems of industry and society at large.

PEO 2: The graduates shall be able to establish themselves as practising professionals, researchers or Entrepreneurs in computer science or allied areas and shall also be able to pursue higher education in reputed institutes.

PEO 3: The graduates shall be able to communicate effectively and work in multidisciplinary teams with team spirit demonstrating value driven and ethical leadership.

PROGRAMME SPECIFIC OUTCOMES

Graduate possess -

- PSO 1:** An ability to apply knowledge of data structures and algorithms appropriate to computational problems.
- PSO 2:** An ability to apply knowledge of operating systems, programming languages, data management, or networking principles to computational assignments.
- PSO 3:** An ability to apply design, development, maintenance or evaluation of software engineering principles in the construction of computer and software systems of varying complexity and quality.
- PSO 4:** An ability to understand concepts involved in modeling and design of computer science applications in a way that demonstrates comprehension of the fundamentals and trade-offs involved in design choices.

PROGRAMME OUTCOMES

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

COURSE OUTCOMES

COs	Description
C410.1	The students will be able to analyse a current topic of professional interest and present it before an audience.
C410.2	Students will be able to identify an engineering problem, analyse it and propose a work plan to solve it.
C410.3	Students will have gained thorough knowledge in design, implementations and execution of Computer science related projects.
C410.4	Students will have attained the practical knowledge of what they learned in theory subjects.
C410.5	Students will become familiar with usage of modern tools.
C410.6	Students will have ability to plan and work in a team.

CO MAPPING TO POs

COs	POs											
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
C410.1	3	2	3	3	3	3	2	3	3	2	3	2
C410.2	2	3	3	3	3	3	2	3	2	3	3	3
C410.3	3	2	2	3	3	3	3	3	2	3	3	3
C410.4	3	3	2	3	3	3	3	2	3	3	3	3
C410.5	2	3	3	3	2	3	3	2	2	3	2	2
C410.6	3	2	3	2	2	3	2	3	2	3	2	2
Average	2.67	2.5	2.67	2.83	2.67	3	2.5	2.67	2.33	2.83	2.67	2.67

CO MAPPING TO PSOs

COs	PSOs			
	PSO1	PSO2	PSO3	PSO4
C410.1	3	3	1	3
C410.2	2	3	2	3
C410.3	3	1	3	2
C410.4	2	1	3	2
C410.5	3	1	2	2
C410.6	3	2	3	3
Average	2.67	1.833	2.33	2.5

ABSTRACT

In the early 2000s, the primary purpose of mobile phones was to communicate by calling or texting an interlocutor. However, nowadays the use of mobile phones is different with the launch of the iPhone in 2007 and the Apple App Store in 2008, mobile phones have seen the rise of new functionalities and seem to have become vital for a large part of its users. Since then, mobile phones have become tools that have changed our world, allowing users to entertain themselves learn, and search for information faster and more efficiently. Most of these features have become obtainable thanks to applications, either directly included when purchasing the mobile phone, or downloadable from application stores such as the Apple App Store or Google Play Store. Our project is a platform for generating applications for educational institutions. It will reduce the time taken and the cost needed for creating a social media application. And also limited expense is needed for its maintenance.

CONTENTS

List of Figures	xii
List of Abbrevations	xiv
1 Introduction	1
1.1 Overview	1
1.2 Objectives	2
1.3 Data Description	2
1.4 Organization of the project	2
2 Literature Survey	3
2.1 Simulating User Interactions: A Model and Tool for Semi-realistic Load Testing of Social App Backend Web Services	3
2.1.1 Related Works	3
2.2 An Empirical Evaluation of the User Interface Energy Consumption of React Native and Flutter	5
2.2.1 Three most common used frameworks	6
2.2.2 Testing Devices	7
2.3 Application of Low-Cost Methodologies for Mobile Phone App Development	8
2.3.1 Low-Cost Mobile Phone App Development	9
2.4 Efficient Way Of Web Development Using Python And Flask	11
2.4.1 Technological Advantages	12
2.5 An Introduction to Hybrid Platform Mobile Application Development . . .	13
2.5.1 Approach to Build a Hybrid Platform Mobile Application	14
3 METHODOLOGY	16
3.1 Modules	16
3.1.1 Web Module	16
3.1.2 Data Acquisition Module	16
3.1.3 Automation Module	16
3.1.4 Application Module	16
3.2 System Requirements and Specifications	17
3.2.1 External interface Requirements	17

3.2.2	Functional Requirements	17
3.3	Data Flow Diagrams	20
3.3.1	Data Flow Diagram- Level 0	20
3.3.2	Data Flow Diagram- Level 1	20
3.3.3	Data Flow Diagram- Level 2	21
3.4	UML	23
3.5	Architecture[7]	24
3.6	IMPLEMENTATION	25
3.6.1	Website	25
3.6.2	Python Script	27
3.6.3	Flutter Application	28
4	Results & Discussion	30
5	Conclusion & Future Scope	44
5.1	Conclusion	44
5.2	Future Scope	44
	Appendix	45
A	Code	46

LIST OF FIGURES

Figure No.	Title	Page No.
2.1	UML class diagram displaying the Java classes of the proposed load testing tool	4
2.2	Baseline App, React Native App, Flutter App	7
2.3	Energy use in different devices	8
2.4	Development of Web-based mobile phone using online app builder	10
2.5	Login Code Using Python	11
2.6	Hybrid Application	15
3.1	DFD- Level 0	20
3.2	DFD- Level 1	20
3.3	DFD- Level 2.1	21
3.4	DFD- Level 2.2	21
3.5	DFD- Level 2.3	22
3.6	Use Case Diagram	23
3.7	Structure Chart	24
3.8	PHP Code for collecting college details	25
3.9	PHP Code for collecting image and logo	26
3.10	Excel file to fill	26
3.11	PHP code which calls main python script	27
3.12	Fire-base function	27
3.13	Python script for icon generation and initiation of APP compilation	28
3.14	Dart file named 'constants'	28
3.15	Dependencies used in flutter application	29
3.16	Pages	29
4.1	Page to enter the college details for social media application	30
4.2	Upload college logo and image	31
4.3	Download the excel file, fill it and upload here	31

4.4	Click Generate Application for creating APK	32
4.5	For downloading the APK file	32
4.6	Real-time database	33
4.7	Fire-store database	33
4.8	Login Page	34
4.9	Announcements can be read here	35
4.10	Announcements can be made here using password for admin	36
A.1	Code for home page	46
A.2	Code for Announcement	47

LIST OF ABBREVIATIONS

APP	: Application
API	: Application Programming Interface
DFD	: Data Flow Diagram
SRS	: System Requirements And Specifications
UML	: Unified Modeling Language
OSN	: Online Social Network
PaaS	: Platform as a Service
UI	: User Interface
iOS	: iPhone Operating System
FR	: Functional Requirements
SDLC	: System Development life Cycle
CSS	: Cascading Style Sheets
HTML	: Hypertext Markup Language
APK	: Android Application Package
GCP	: Google Cloud Platform

CHAPTER 1

INTRODUCTION

1.1 Overview

Smartphone technologies are evolving rapidly and, not surprisingly, Mobile applications are as well. In recent years, all the institutions will have their own mobile application for smoother process. Many institutions are currently hiring professionals or software developing companies for develop an application. This process is so costly. And also the average time for developing such an application will take about 4-6 months.

This project aims at reducing the time and cost for developing an application for their institution. This a closed social media application which can be generated instantaneously from the website which we developed, so others who are not permissible by the institution cannot invade directly in to this application. So it become more secure to handle. Students and faculty members can communicate each other, also get important announcements from the authority and websites related to the institution can be accessed through this single app.

And the client will be able to go through the whole process easily. Even a person with less knowledge about mobile application will be able to use our project. The automation system in project can also be used in many other fields. But our project is mainly focusing on institutional purposes only.

1.2 Objectives

The main objective of this project is to access the application easily to every colleges which reduces the time taken and the cost needed for creating a social media application. Also limited expense is needed for its maintenance

1.3 Data Description

There are no machine learning algorithms included in the project therefore there is no need of any specific data-set .The only data that is used in the project is accessed from the client at the initial stage.This data includes all the information about the client institution which are name of the institution,admin details,images of institution and other relevant information.

1.4 Organization of the project

The report is organised as follow:

- Chapter 1: Introduction- Gives an introduction to "College social media app and its automation".
- Chapter 2: Literature Survey- Summarizes the various existing techniques that helps in achieving the desired result.
- Chapter 3: Methodology- Discusses about the methods which are used in this project.
- Chapter 4: Results & Discussions- Discuss about the results of the project.
- Chapter 5: Conclusion & Future Scope- Concludes with the future scope of implementation.
- References- Includes the references for the project.

CHAPTER 2

LITERATURE SURVEY

2.1 Simulating User Interactions: A Model and Tool for Semi-realistic Load Testing of Social App Backend Web Services

Many mobile apps today support interactions between [1]their users and/or the provider within the app. Therefore, these apps commonly call a web service backend system hosted by the app provider. For the implementation of such service backends, load tests are required to ensure their performance and scalability. However, existing tools like JMeter are not able to simulate “out of the box” a load distribution with the complex time evolution of heterogeneous, real and interacting users of a social app, which e.g. would be necessary to detect critical performance bottlenecks. Therefore, in this paper a probabilistic model for simulating interacting users of a social app is proposed and evaluated by implementing it in a prototype load testing tool and using it to test a backend of new real-world social app currently under development.

2.1.1 Related Works

In many cases, server-side components of distributed applications will be deployed in the cloud, typically using the Platform-as-a-Service (PaaS) model (Mell and Grance, 2011). Consequently, synthetic workload generation and performance modeling for cloud environments became a major research focus recently. But since elasticity is a crucial feature of cloud environments, the simulation of workload variations over time has been considered. While approaches for synthetic workload generation for existing OSN applications have been proposed, no solution exists yet for performance testing new OSN applications still under development, for which the user roles and behavior are still unknown.

In this case, only stochastic or probabilistic models could be used to model the expected user behavior, based on general observations about OSN users’ behavior. In general, the activity of users in OSN has been found to depend on the factors: intrinsic interest, breaking news, which could be used to define a generic probabilistic model of users’ behavior. In addition, users’ activities will in general follow an overall periodic time dependency due to day and night, weekends and weekdays etc.

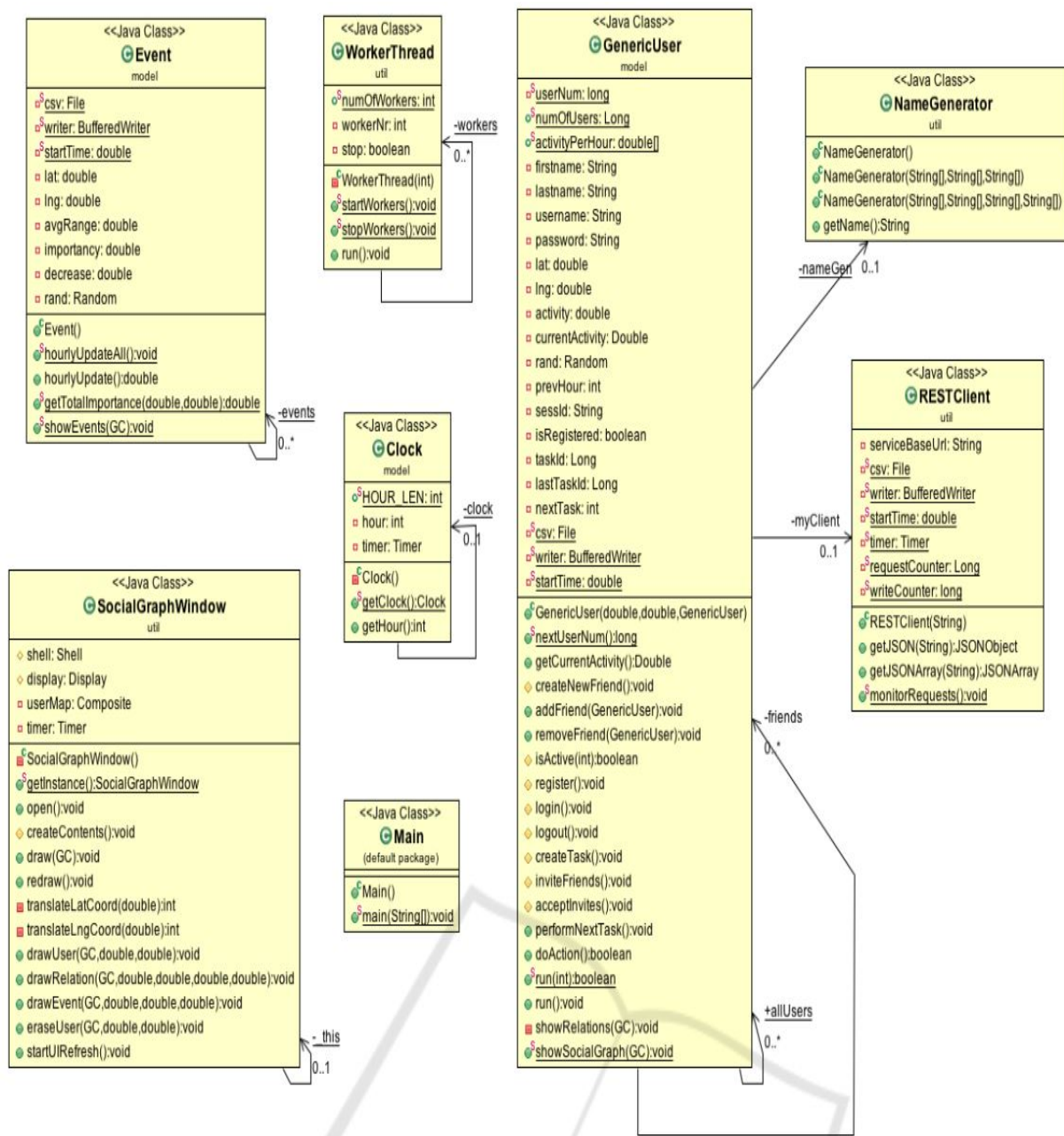


Figure 2.1: UML class diagram displaying the Java classes of the proposed load testing tool

Since no approach exists so far combining all these aspects, in this paper a model and software tool for the synthetic creation of workloads for OSN applications are proposed and evaluated, which combine the following features:

1. Virtual users modeled by finite state machines with probabilistic transitions between the states, where the transitions refer to different user actions,
2. Users' activity depends on their intrinsic interest, external events in their area and their friends' activity,
3. Users are spatially distributed,
4. External events are localized and have a certain spatial impact range.

In conclusion, in this paper a probabilistic model for simulating interacting users of a social app has been proposed and evaluated by implementing it in a prototype load testing tool. The proposed approach is capable not only of simulating the users' activity depending on their interest, external events in their area and their friends' activity, but also takes into account the spatial distribution of users and external events, which are localized and have a certain spatial impact range within a virtual world.

While the evaluation is still preliminary, the results obtained so far are promising. Already with the prototype tool some serious design flaws of a service backend of a new, real-world social app currently under development were detected. However, further research is needed to continue the evaluation of the proposed approach in lab and field tests with respect to its possible applications, ease of use, performance and adaptability.

2.2 An Empirical Evaluation of the User Interface Energy Consumption of React Native and Flutter

Energy efficiency is a growing area of concern for mobile developers, as good battery life is highly desired by end users of mobile devices. While many developers work to increase their app's energy efficiency during development, there is not much information available about the energy efficiency [4] of the different app frameworks on the market. As the choice of a framework must be made before the start of development, and cannot be easily changed later on, information about these frameworks is crucial to allow developers to optimize their apps for efficiency. A comparison between the energy use of the React Native and Flutter frameworks while performing User Interface tasks to the native Android API can be done. certain UI actions are consistently more or power-hungry than average, and the energy use tendencies of these actions tended to be consistent between different frameworks and devices. measuring overall energy use between separate test runs is inconsistent, and further research is necessary to identify the best method to isolate the energy use of a single app.

2.2.1 Three most common used frameworks

There are a number of Android app frameworks available for use, and these frameworks use a variety of techniques to display the user interface of an app. Three most common frameworks are Android API, React Native and Flutter.

1. **Android API** : The Android API uses a collection of elements combined by a developer into Activities, which form the basis of an app. Each screen of an app can be defined by an Activity, but can also be created by combining independent fragments into an activity, and these fragments can allow an app to change its UI without transitioning between Activities. Android apps using the Android ADK are written in Java or Kotlin, and executed by the Android Run Time (ART) on all Android versions 5 or higher. ART has been shown by Chen and Zong to be significantly more energy efficient than the older Dalvik Java Virtual Machine used in older versions of Android, and all tests of the baseline app will be performed on devices using this version of the JVM.
2. **React Native** : React Native has been shown to perform equal or better to a native Android app in terms of response time, with near equal results in memory usage and frame rate consistency, but with higher CPU usage. To render the UI, React Native uses "native bridges" to convert UI elements written in JSX to those exposed by the Android API. Unlike the Android API, React Native uses JavaScript for the app logic and user interface, using the JavaScriptCore JS engine, using JIT compilation on our Android test devices.
3. **Flutter** : Flutter uses the Dart programming language, and application code is Ahead-Of-Time compiled and built into an APK using the Android NDK. In contrast to React Native, Flutter does not use the UI elements built into Android, but instead renders the UI independently of the underlying system in a single view. This core difference could affect the energy efficiency of the app, as inefficiencies in the Android API could be avoided by the independent rendering, or vice versa.

Energy consumption of the different frameworks is compared by implementing a similar app in each framework. Automated testing is performed on each app to exercise the UI elements. Energy consumption of the phone with Android's built-in battery statistic services is also measured.

The testing app is created with a set of ui elements. Only elements that are provided by all of the frameworks are used in order to simplify the app and make a fair comparison between the app frameworks being tested. Animations and different transitions between screens are also included. Some of the tested features are:

1. Text Lists

2. Image Lists
3. Modal Dialogs
4. Buttons
5. Linear Animations

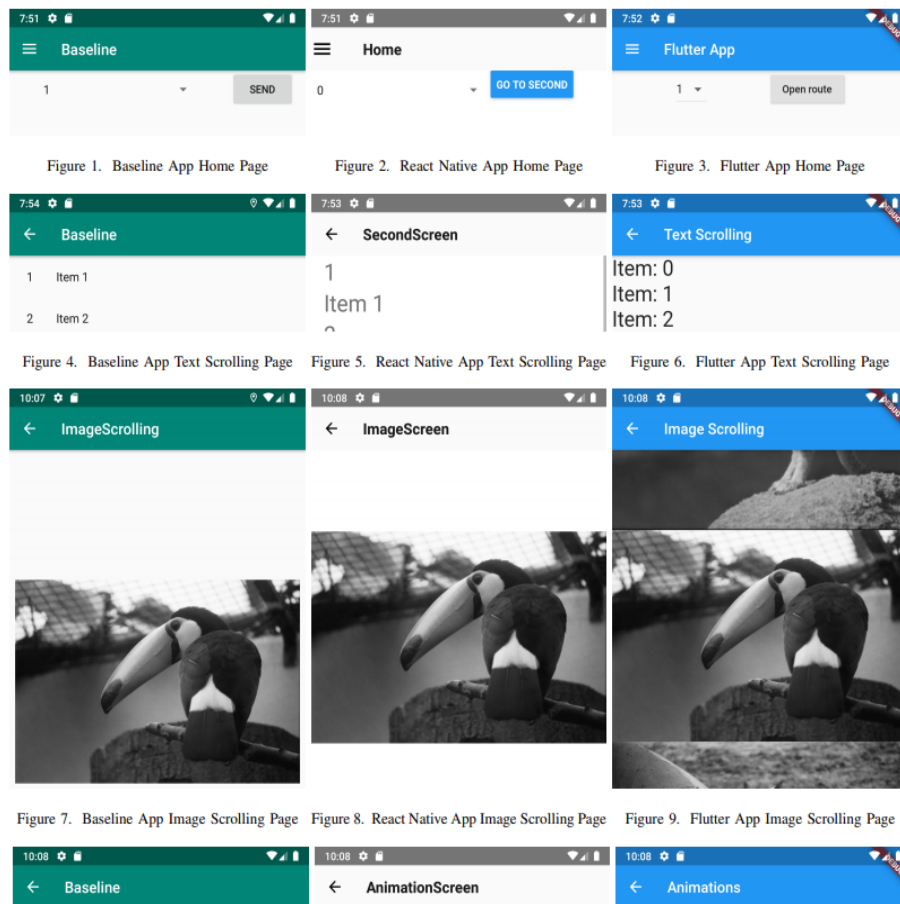


Figure 2.2: Baseline App, React Native App, Flutter App

2.2.2 Testing Devices

A small selection of devices were taken to verify differences measured between apps across different Android versions and device hardware.

1. Nokia 7 plus
2. Google Pixel XL
3. Motorola G5 Plus

Three of these devices were used to test each of the different frameworks and each device resulted in different test outcomes.

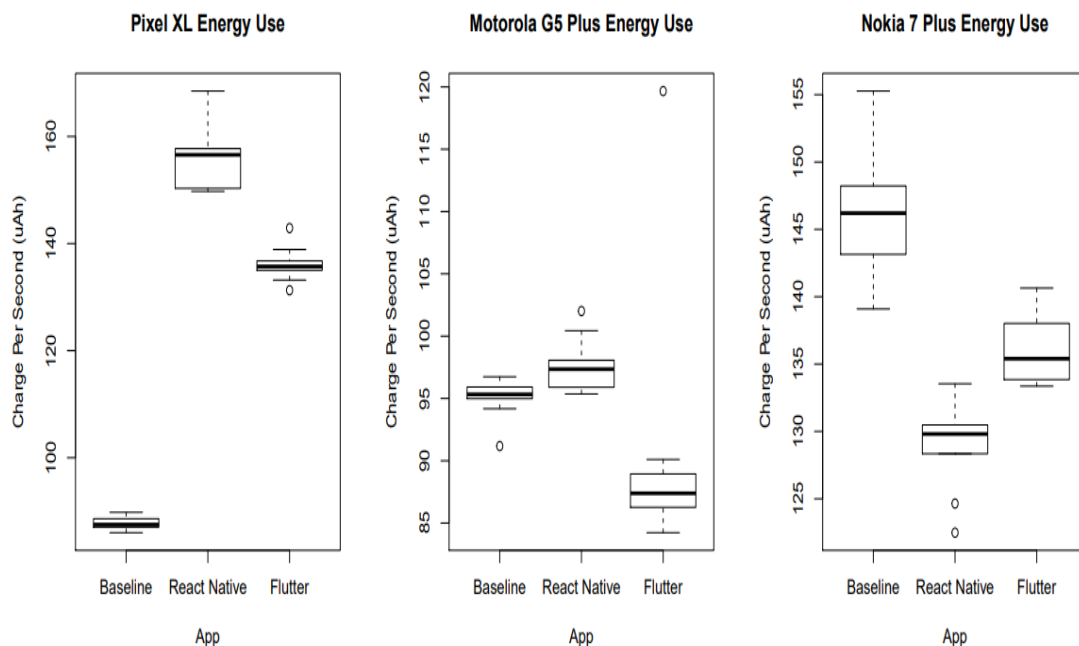


Figure 2.3: Energy use in different devices

2.3 Application of Low-Cost Methodologies for Mobile Phone App Development

The usage of mobile phones and mobile phone apps in the recent decade has indeed become more prevalent. Previous research has highlighted a method of using just the Internet browser and a text editor to create an app, but this does not eliminate the challenges faced by clinicians. The aims of this research article are to:

1. highlight a low-cost methodology that clinicians without technical knowledge could use to develop educational apps;
2. clarify the respective costs involved in the process of development;
3. illustrate how limitations pertaining to dissemination could be addressed.[2]

The worldwide prevalence of mobile phones has made them a very useful and innovative platform for the provision of patient care, as well as in helping clinicians with management decisions. Over the past few years, mobile phones have advanced drastically in terms of both their functionality and design, and they are currently more than what used to be a simple call and messaging device. They have literally been transformed into sophisticated personal mini-computers.

Along with the enhancements in mobile phone functionality, there has been an increased number of educational mobile phone apps made available for users to download and install.

However, it is a well-known fact that anyone could publish a medical app, and the app stores do not routinely do a rigorous review of the accuracy of the content of the mobile app prior to publication. Although mobile phones have been used by the majority of interns on a daily basis in performing their job, there still needs to be more guidance and advice with regards to the accuracy and the credibility of the information provided within the apps. Despite the high usage of mobile phone and its apps, the development of mobile phone apps by professionals will incur a huge cost. Very often, clinicians and researchers have to wait for and hope to be successful in securing grants to finance the developmental costs. Apart from the concerns about the high cost associated with development, another concern lies with the fact pertaining to how evidence-based apps are. Most of the current apps available have been developed by external vendors and developers, and are lacking inputs from clinicians who have a vast amount of knowledge and expertise in their specialized fields. Recent studies have highlighted the need for clinicians to be more involved in the mobile phone app development process and a research article highlighted a simple methodology of creating an application using just an Internet browser and a text editor. The methodology shared seemingly seemed to help overcome the fears of clinicians, but the methodology shared previously does require clinicians to have some fundamental technological knowledge. Also, the previous methodology shared does not enable clinicians to include more multimedia features in the app.

2.3.1 Low-Cost Mobile Phone App Development

There are various online Web-based mobile phone app builders such as Conduit Mobile and IbuildApp. The advantages of using online Web-based mobile app builders are that its graphic user interface will help in the immediate integration of text-based content, videos, questionnaires, and other multimedia features. These multimedia features include built-in photo-taking capabilities as well as e-commerce capabilities. An overview of the features that could be integrated using an online application builder is exemplified (Figures 2.3). Integration of content was simple. All users have had to do is select the appropriate interactive feature and then key in the relevant information. Subsequently, as the online app builder allows for the generation of an application programming interface (API) for download

In order to be published on the mobile app stores, creator has to set up a developer account and upload the API, along with images of the app. Publication to the iTunes store was similar, though the cost of the developer account was vastly different. The costs were US 25 per annum for an account with Android Play, but US 300 per annum for an account with the iTunes store.

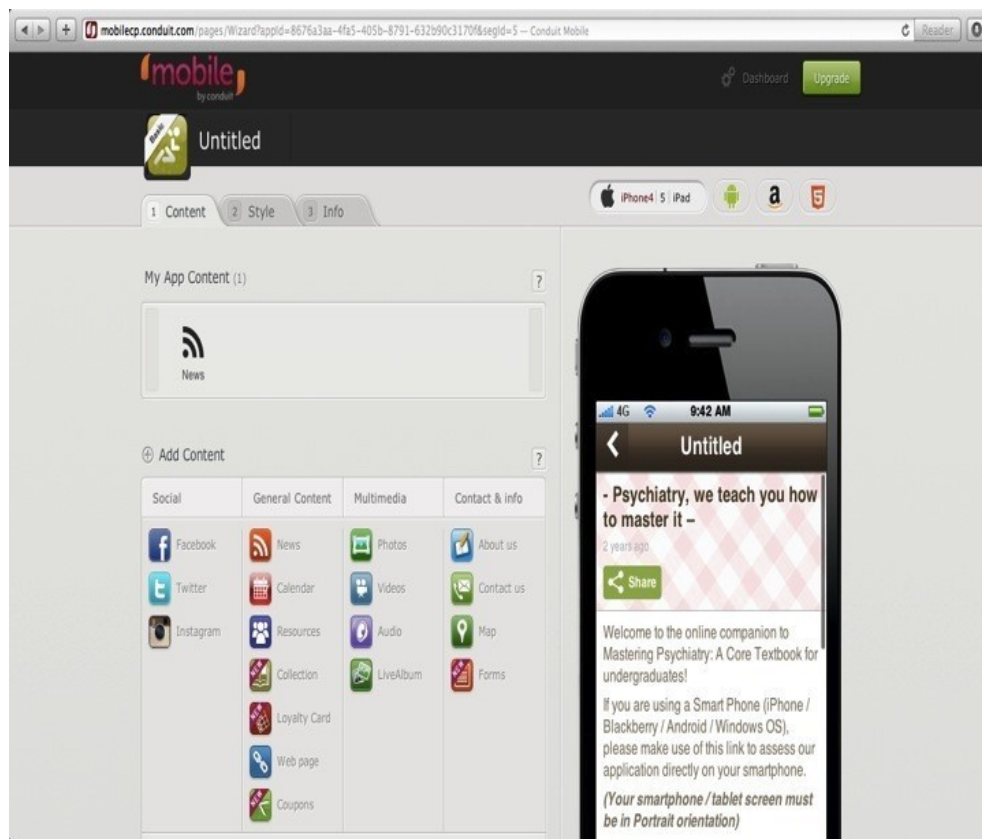


Figure 2.4: Development of Web-based mobile phone using online app builder

Nevertheless, several limitations remain in the current study. The authors have not formally evaluated the perspectives of non-Asian users of self-created mobile apps. The authors have addressed the previous limitations pertaining to the difficulties associated with the access of the mobile app, by making the app available on one of the app stores. However, the authors have not mentioned their experience with making the mobile app available on other stores, such as iTunes. It should be noted that different app stores have different criteria for acceptance of mobile apps and there might be a chance one store would accept and another might reject the proposal. This would limit the distribution of the mobile app across multiple computing platforms. Another limitation would be the lack of conducting focus groups before deciding what would be the most appropriate contents to be integrated within the respective mobile apps.

This is one of the few studies that has demonstrated that clinically relevant content for mobile phones could be developed by clinicians and clinical teachers using both low-cost, and non-technical methodologies. The results obtained have demonstrated that these Web-based low-cost mobile apps are applicable in real life

2.4 Efficient Way Of Web Development Using Python And Flask

Web is the most frequently used networking aid which satisfies the requirements of all types of users; it provides a solution for any type of problem definition. While developing a web portal the appearance of web portal makes a development more critical. The good appearance of a web can easily attract more number of visitors which is a success of web portal. For designing and developing such well structured and with the good appearance of web we have to choose a proper technology. The technological needs of satisfying a good web portal can be fulfilled by "python" and "flask".

Python is a general purpose, high level programming language that focuses on the code readability, for web development lines of code will be fewer than other languages. It is possible for Python because of large standard libraries which make the web development code simple and short.

```
@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        email = request.form.get('username', None)
        password = request.form.get('password', None)
        profile = Profile.authenticate(email, password)
        if profile and profile.id:
            set_session_and_login(profile)
            flash('Successfully logged in.', category='success')
            return redirect('/stream/me')
        return render_template('/generic/main/login.html',
                               menu=MenuView(None))
    def set_session_and_login(profile):
        session['user'] = str(profile.id)
        event=LoginEvent(user=session['user'],
                          url=request.url,
                          ip_address=request.remote_addr)
        event.save()
    @app.route('/logout', methods=['GET', 'POST'])
    @login_required
    def logout():
        if hasattr(g, 'user'):
            g.user = None
            session.clear()
        return redirect(url_for('home'))
```

Figure 2.5: Login Code Using Python

These libraries have pre-coded functions provided by Python community which can be easily downloaded and can be used as per the development needs. Initially Python was designed for web servers to deal with the incoming traffic on the server.[3]

Flask is a micro framework of Python which provides the basic functionality of web framework and allows more plug-ins to be added so the functionality and feature set can be extended to a new level. Flask is called as micro framework of Python because it makes the core functionality simple but extensible in terms of development. It can also be used to save time building web applications.

Flask uses Jinja Template Engine and the Werkzeug WSGI Toolkit. Flask structure is categories into two parts “Static files Template files”, template file have all the Jinja templates including HTML pages, where as static file have all static codes needed for website such as CSS code, JavaScript code and Image files.

2.4.1 Technological Advantages

- **Extensible**

Extensibility in web development is a principle rule designed as a system’s ability to have new functionality extended, in which the system’s internal structure and data flow are minimally or not affected, particularly that recompiling or changing the original source code is unnecessary when changing a system’s behavior, Because systems are long lived and will be modified for new features and added functionalities demanded by users.

- **Robust**

Robustness is the ability of system to cope with errors during execution. Robustness is also used as the ability of an algorithm to continue operating despite abnormalities in input, calculations, etc. Robustness can encompass many areas of web development.

- **Open Source**

Python Flask are an open source languages in which the source code is available to the general public for use and/or modification from its original design. Open-source code is typically a collaborative effort where other developers can improve upon the source code and share the changes within the community so that other members can help improve it further.

Flask is a lightweight web application framework written in python and baseband on the WSGI toolkit and jinja2 template engine. Flask takes the flexible python programming language and provides a simple template for web development. Once imported into python, Flask can be used to save time building web applications. It keeps the core simple but extensible. It has no database abstraction layer, form validation, or any other components. Flask supports

extensions. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and more.

Great thing about Python and Flask is that it is portable and interactive language for web development including speedy prototyping and dynamic semantics potential. In python it is also possible to bind new modules to Python to extend its core functionality. Many of great websites are moving to Python because its robustness. Python provides easy way to use standard libraries. Python works with almost all databases, powerful text processing and document processing facilities, and plays well with other web technologies.

As we all know that web development is a complex process of structuring content with dynamic data transactions. For maintaining such complexity technologies such as python Jinja Flask are more useful. Such technologies are also helps to create more user friendly interface for data fetch from WWW. This paper can be conclude as Python can be used for making web more powerful, fast and efficient with the help of Flask Template Engine.

2.5 An Introduction to Hybrid Platform Mobile Application Development

The mobile applications development industry has been developing at a rapid pace. The various operating systems available in the market are diverse and this proves to be a hindrance to application developers while developing a single application for all the operating systems. Hybrid platform mobile applications help in cost cutting and saving time as well as providing components for easier development of applications which provide a native feel to the user. This paper aims to help developers make the right choice in order to build an application as well as give vital information about hybrid platform mobile application approaches and their advantages and disadvantages.

Native applications are developed using the native programming languages of the devices for which it needs to be developed. For example, The native programming language for iOS is objective C and the new swift and for android, applications are built using JAVA programming language. Native apps provide with the best usability, features and best overall user experience. The characteristic that any native application would possess are:

1. **Multi touch:** Allows the user to double tap, pinch-spread and other complex UI gestures.
2. **Fast graphic API:** Any native application provides with the fastest graphics depending on the inbuilt characteristics of various devices.
3. **Fluid animation:** It is very essential for providing a good gaming experience on the device. It is also necessary for highly interactive reporting or compound computational algorithms for transforming audio and videos.

4. **Built-in components:** Inbuilt application such as camera, address book, photo gallery, Geo-location and other features can provide their services to mobile apps.
5. **Ease of use:** The user interface of native applications is easily understandable by users which enables them to seamlessly interact with the application.

Applications have rapidly caught up in recent times ever since app stores were made for the mobile phones developers have become more and more interested in creating apps for different operating system and platforms. But the problem was that the same application had to be created for different operating systems which led to wastage of resources and time. Therefore, Hybrid platforms for coding using HTML, CSS and JavaScript were created which led to a new beginning of mobile applications. Hybrid platform apps are developed using technologies such as: HTML, CSS and JavaScript. It is then placed into a native container such as Adobe PhoneGap. These native containers run the application code and package it into an app. Therefore it provides us with various Mobile Application Development Platforms that can generate largely native apps from a single original codebase, which can be deployed across various mobile platforms.

2.5.1 Approach to Build a Hybrid Platform Mobile Application

The Hybrid Approach may be seen as a bridge between Web and Native approaches. A Hybrid application is built using HTML, CSS and JavaScript which are web technologies and are executed in the native Hybrid approach uses the browser engine of the device which renders and displays the HTML content in full screen Web view control. Even during the development of Hybrid applications on the desktop, we can view the application in the browser. The device capabilities are exposed to the hybrid application through an abstraction layer. The abstraction layer exposes the device capabilities as JavaScript Application Programming Interface. Hybrid approach can take the advantage of both browser engine and device capabilities. Hybrid approach can be used for both server backed and standalone applications. Unlike web applications Hybrid applications needs to be downloaded and installed on the mobile device.

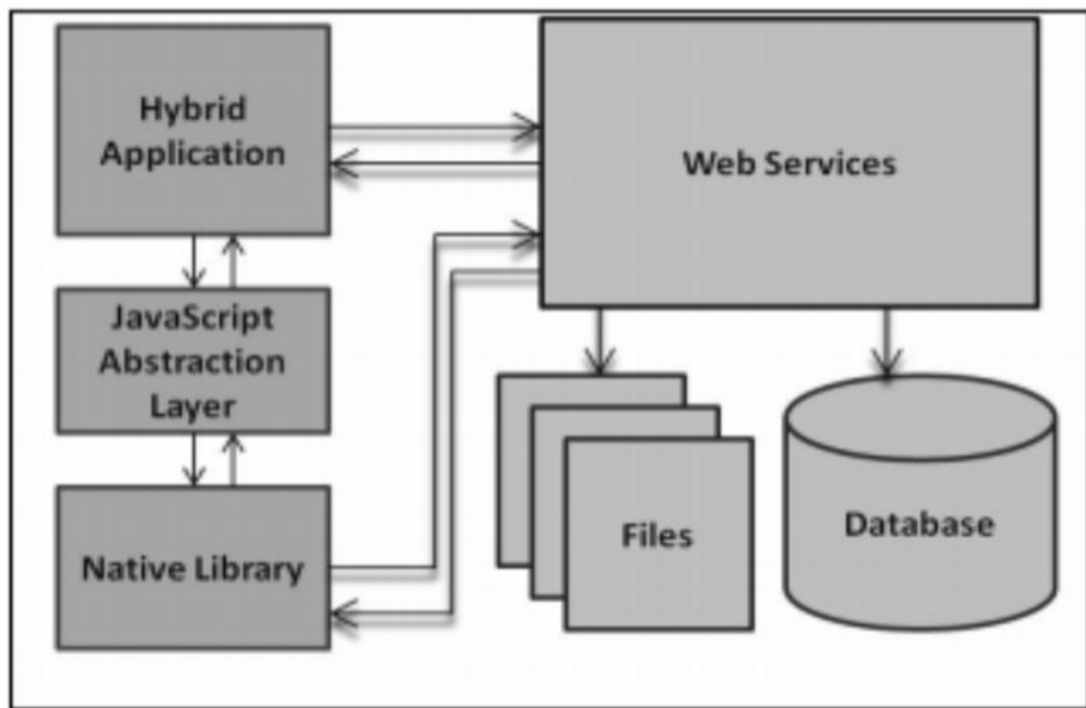


Figure 2.6: Hybrid Application

The field of mobile application development has developed rapidly in the last few years. Native mobile applications were the only type of applications which were common in the past. Native mobile applications without doubt provide the best user experience however, hybrid platform mobile applications will be preferred when the application is to be made for multiple platforms and time and cost are the primary factors. A developer needs to weigh the pros and cons of both hybrid and native applications and choose the right approach to building an application based on experience and the application's purpose.[5]

CHAPTER 3

METHODOLOGY

3.1 Modules

3.1.1 Web Module

In the initial stage the data for the application is collected from the website. The website contains a UI. The client has to fill the form in the UI in order to input the data. The website can send the input to the server. The web module also formats the given data in specific way for the server to read it easily. Web module can be built with common website building frameworks. The UI should be made in such a way that the user can easily understand. The collected data also includes images and login details. The web module will also provide the generated applications to the clients

3.1.2 Data Acquisition Module

The data from web module will be received by the server in a specific format. All the received data will be initially saved inside separate directories. There will be different directories for various data. It is the responsibility of the user not to make any mistakes in the uploaded data. If any mistakes occur, they will have to rerun the process from the beginning. All the data directories will be stored even after the whole process is completed, for any future needs.

3.1.3 Automation Module

This module is used to automate the app generation process. Usually flutter apps are generated in flutter environment. An environment for flutter app generation will be built in server. The flutter app template will contain all the necessary files for app generation. The Automation module will inject the acquired data into the template files. After all the necessary procedures "flutter run" command will be called in the flutter environment, and applications will be generated. These applications will be sent to the web module.

3.1.4 Application Module

After all the procedures the application will be available in the web module. Application can be installed on Android or iOS devices. The user can login to the application with the user id provided by the institution. The user is also capable of changing their password. Application will provide the services like:

1. Status update

2. Profile management
3. College updates
4. Chat service
5. Tasks management (ToDo)
6. Faculty details
7. Student complaint and suggestions

3.2 System Requirements and Specifications

This section contains more detailed information regarding the needed functionalities of our application, including interfaces, internal functions, and testing requirements.

3.2.1 External interface Requirements

User Interfaces

The first time user should see a log-in page when he/she opens the app. The user could be able to register for the app.

Software Interfaces

The mobile app needs to communicate with various databases that contains photos, videos, pdf etc. Since users are only viewing the content the communication only contains reading operations.

3.2.2 Functional Requirements

Automation of Application development

1. Functional requirement 1.1

- ID: FR1
- Title: Input data for website
- DESC: Metadata for social media app development
- RAT: In order to collect information for generating application
- DEP: None

2. Functional requirement 1.2

- ID: FR2
- Title: Automation
- DESC: Automatically generate the application based on the collected data.

- RAT: In order for a user to download the application
- DEP: FR1

User Interface - User

1. Functional requirement 2.1

- ID: FR3
- Title: User login
- DESC: The user should be able to login to the application using given login details.
- RAT: In order for a user to login to the application
- DEP: FR1

2. Functional requirement 2.2

- ID: FR4
- Title: Home screen
- DESC: The user should be directed to the home screen or the dashboard of the application immediately after login for the ease of accessing the features of the application.
- RAT: In order to help users find the resources.
- DEP: FR3

3. Functional requirement 2.3

- ID: FR5
- Title: Chat Option
- DESC: A module for exchanging messages
- RAT: In order to allow the user to communicate with each other.
- DEP: None

4. Functional requirement 2.4

- ID: FR6
- Title: Help options
- DESC: The user should be directed how to use the application. For that there should be good manuals or proper documentation that can be accessed from the home screen.
- RAT: In order to identify the various options available in the application
- DEP: FR3

User Interface - Admin

1. Functional requirement 3.1

- ID: FR7
- Title: Library Management
- DESC: Information regarding library usage
- RAT: Details of books borrowed- date, book name, student name, book number etc
- DEP: FR1

2. Functional requirement 3.2

- ID: FR8
- Title: Database Management
- DESC: Manipulating data regarding new users.
- RAT: Remove and update the metadata.
- DEP: FR1

Other Non-functional Requirements

1. **Performance Requirements** The performance of the application should be very good. There should not be any delays when switching between different pages and updating story.
2. **Safety Requirements** Trespassers cannot login to the application. Only certified members can access the application using login details like user-id and passwords. Users can change the password after first login in order to ensure the security.
3. **Security Requirements** Chats are end-to-end encrypted. The details of the user like name, class, etc. should be collected with the proper permission of the user. There will be an option for users to select the viewers of the story. Software Quality Attributes

- Adaptability

The application can be run on any android device and iOS devices.

- Reliability

The app would not crash [6]while using it except the error caused by external reasons like OS error.

- Availability

The application will be available for admin and he/she can provide application to the users.

3.3 Data Flow Diagrams

3.3.1 Data Flow Diagram- Level 0



Figure 3.1: DFD- Level 0

3.3.2 Data Flow Diagram- Level 1

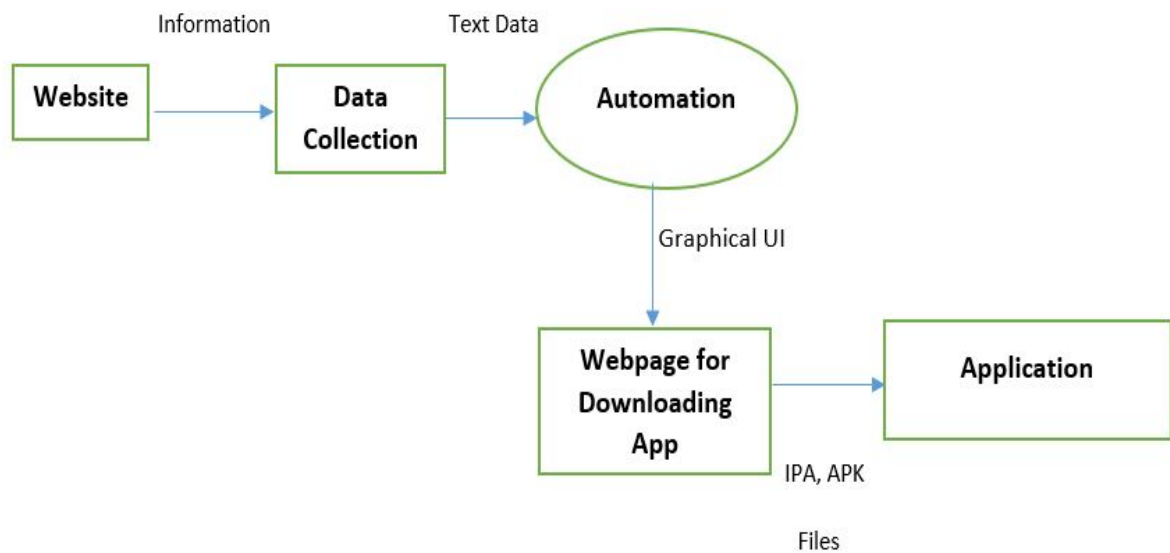


Figure 3.2: DFD- Level 1

3.3.3 Data Flow Diagram- Level 2

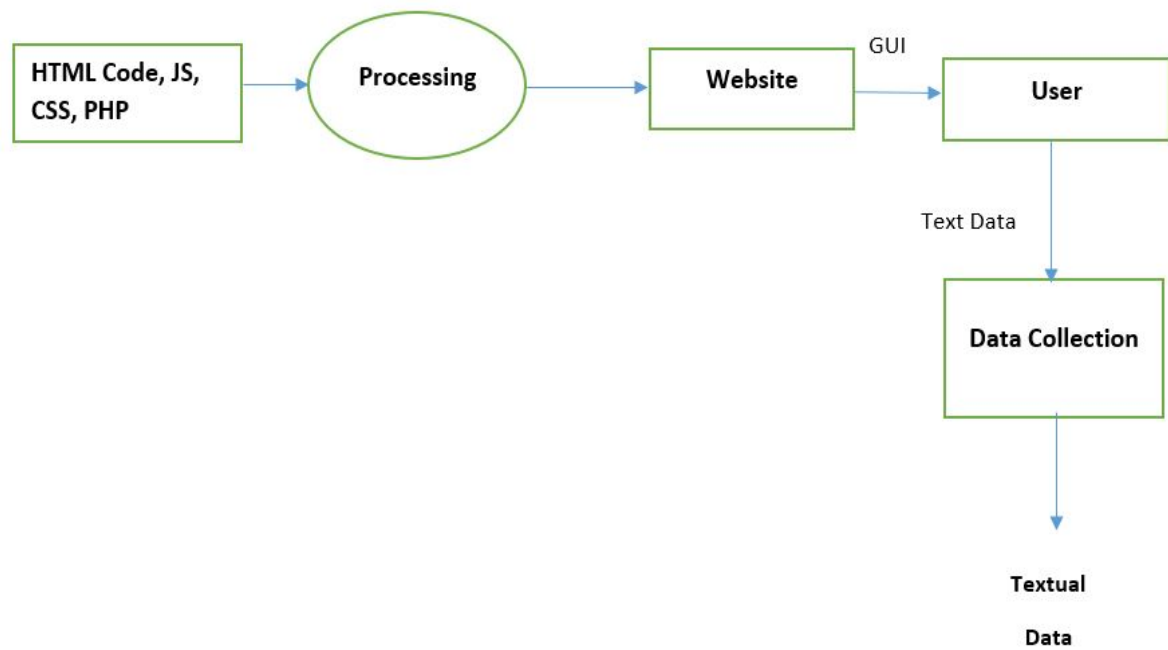


Figure 3.3: DFD- Level 2.1

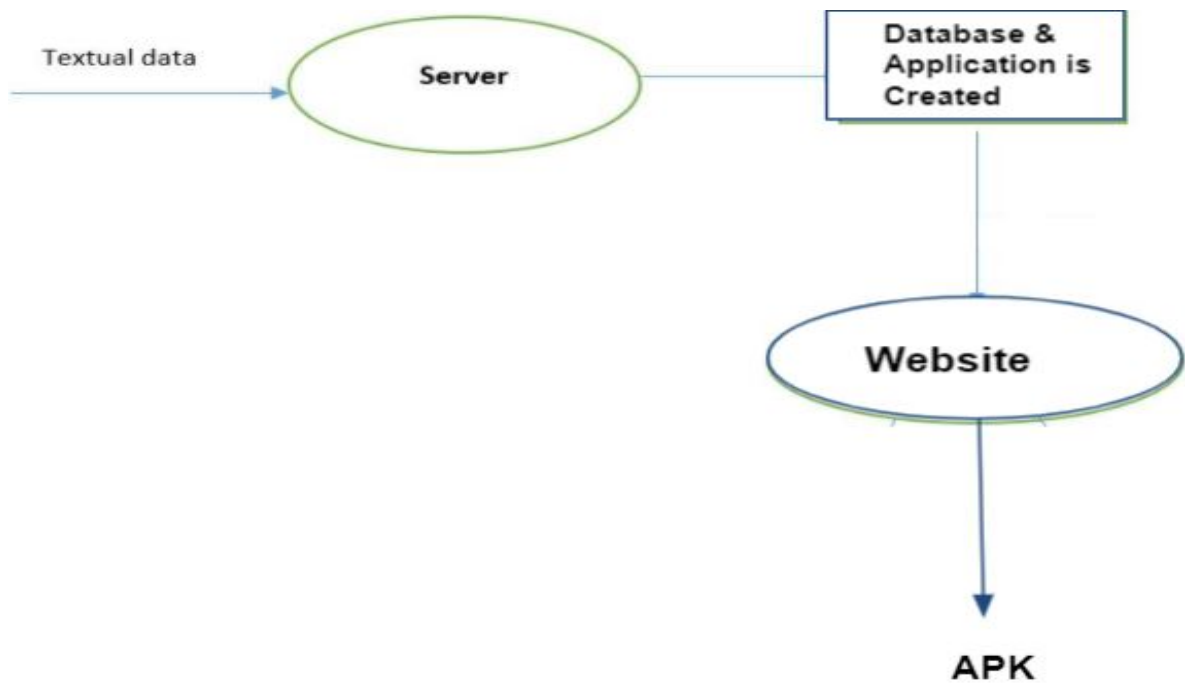


Figure 3.4: DFD- Level 2.2

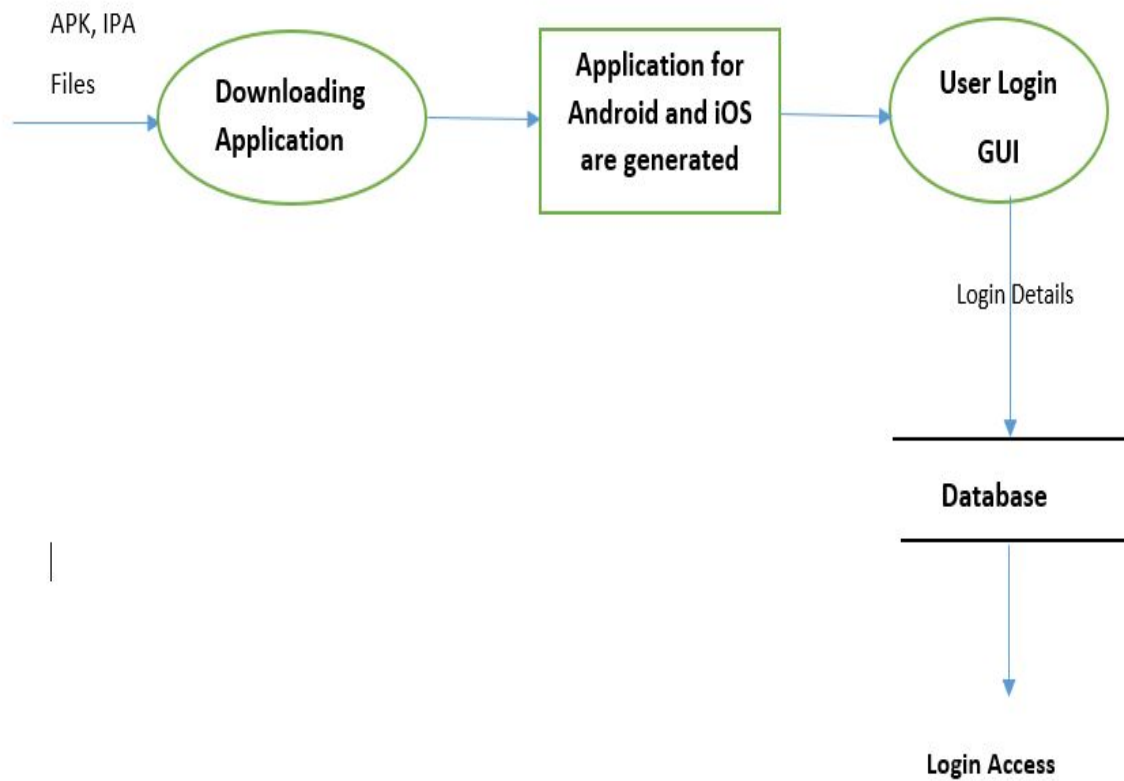


Figure 3.5: DFD- Level 2.3

3.4 UML

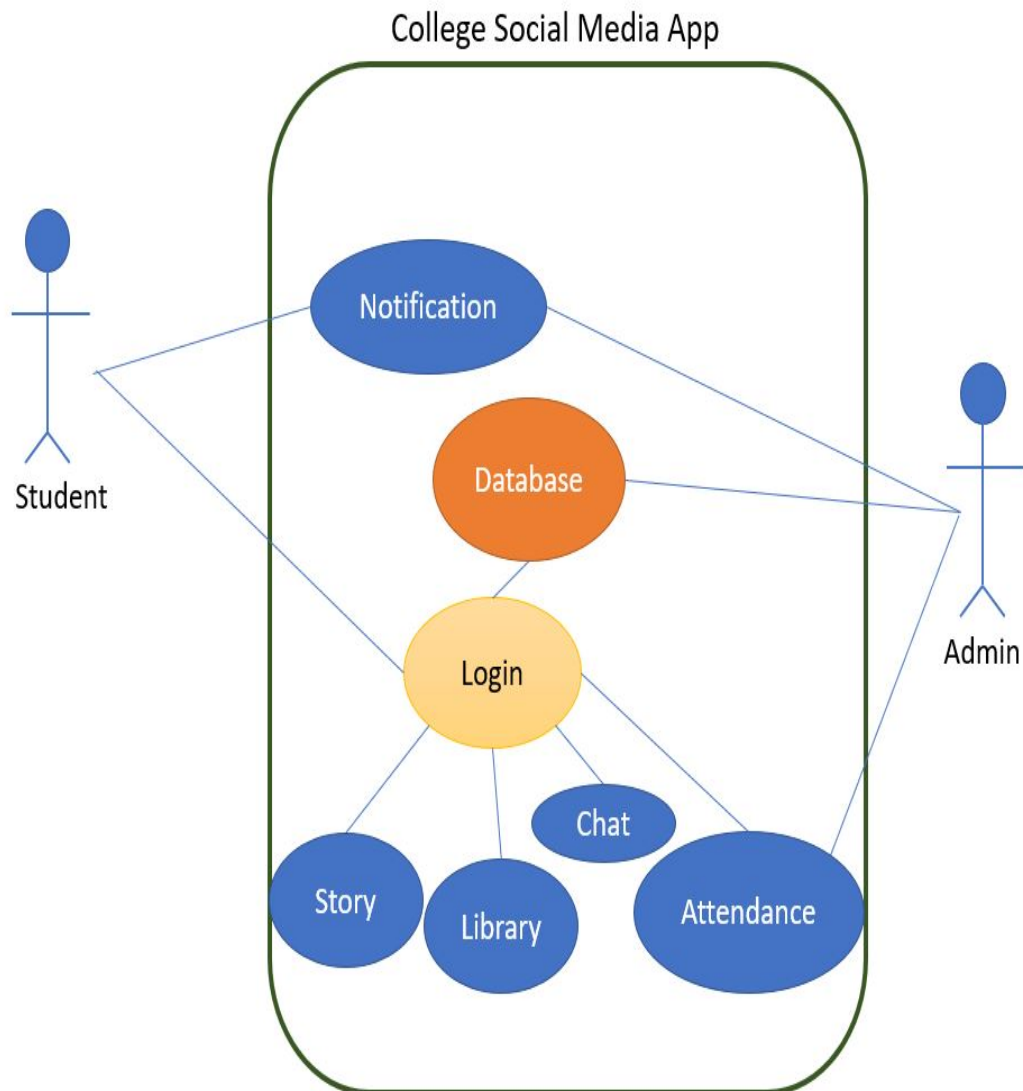


Figure 3.6: Use Case Diagram

3.5 Architecture[7]

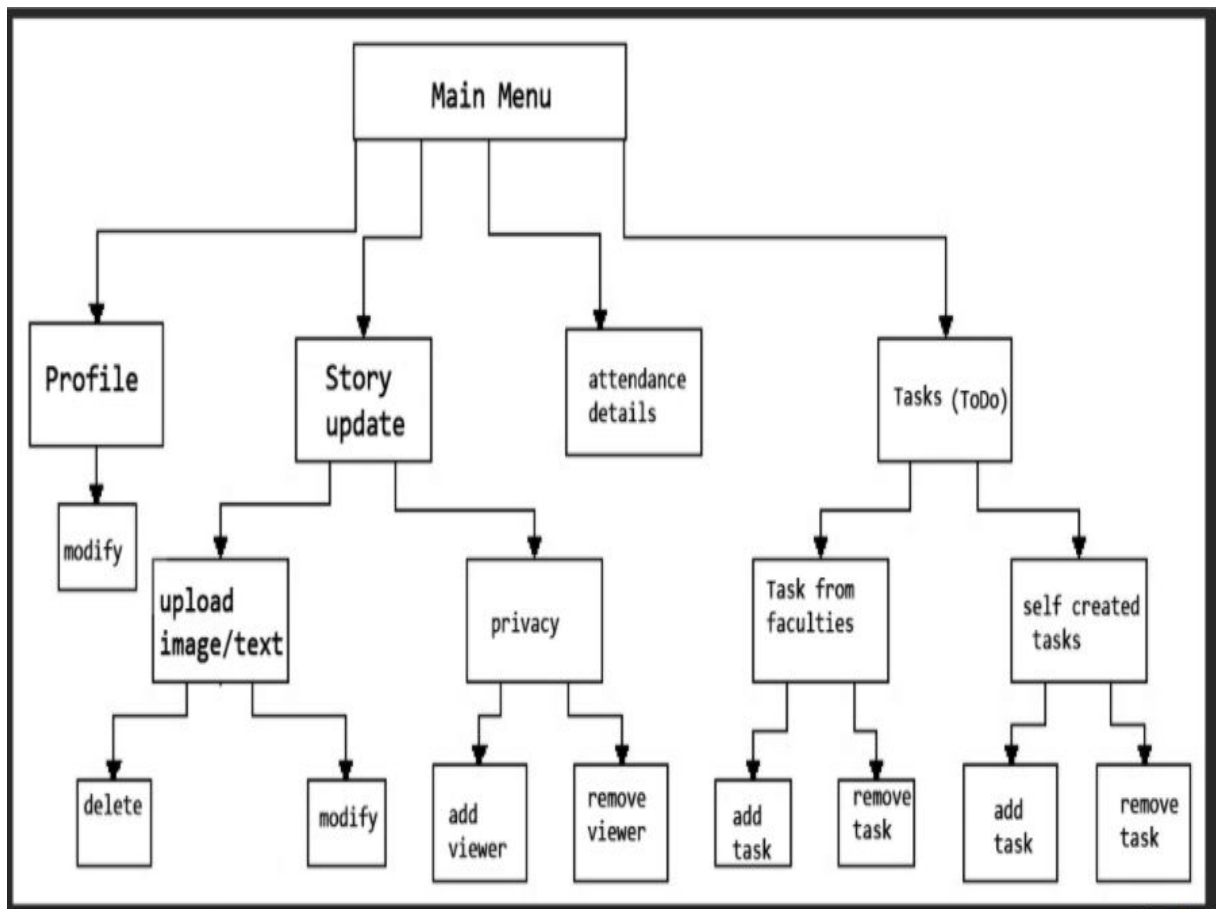


Figure 3.7: Structure Chart

3.6 IMPLEMENTATION

The implementation is done by three sections:

1. Website
2. Python Script
3. Flutter Application

3.6.1 Website

Website is created using HTML5 and PHP. It includes 6 web-pages. Home page contains a form which collects basic details of the institution. The details are collected by PHP and it is written to a text file in a line by line format.

```
<?php
    extract($_REQUEST);
    $file=fopen("details.txt","w");
    fwrite($file, $inst_name ."\n");
    fwrite($file, $app_name ."\n");
    fwrite($file, $details ."\n");
    fwrite($file, $reason ."\n");
    fwrite($file, $library ."\n");
    fwrite($file, $website ."\n");
    fwrite($file, $adminpass ."\n");
    fclose($file);
    $command = escapeshellcmd('python C:\xampp\htdocs\python\clean.py');
    $output = shell_exec($command);
    echo $output;
?>
```

Figure 3.8: PHP Code for collecting college details

Second web-page collects the college image and logo. These images are moved into specific directories.

```

<?php
if(isset($_FILES['image'])){
    $errors= array();
    $file_name = $_FILES['image']['name'];
    $file_size =$_FILES['image']['size'];
    $file_tmp =$_FILES['image']['tmp_name'];
    $file_type=$_FILES['image']['type'];
    $tmp=explode('.', $file_name);
    $file_ext=strtolower(end($tmp));
    $file_namew='logo.jpg';
    $extensions= array("jpeg","jpg","png");
    if(in_array($file_ext,$extensions)=== false){
        $errors[]="extension not allowed, please choose a JPEG or PNG file.";
    }
    if($file_size > 2097152){
        $errors[]='File size must be excately 2 MB';
    }
    if(empty($errors)==true){
        move_uploaded_file($file_tmp,"images/".$file_namew);
        echo "Success";
    }else{
        print_r($errors);
    }
}

```

Figure 3.9: PHP Code for collecting image and logo

Third web-page accepts an excel file. The excel file headers should not be changed for avoiding errors.

	A	B	C	D	E
1	regno	branch	email	fullname	rollno
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					

Figure 3.10: Excel file to fill

The python script for further processing is called in the last web-page.

```
<?php
extract($_REQUEST);
$command = escapeshellcmd('python C:\xampp\htdocs\python\main.py');
$output = shell_exec($command);
echo "done";
?>
```

Figure 3.11: PHP code which calls main python script

3.6.2 Python Script

Python script controls the further process such as

1. Moving image files
2. Removing previously generated files
3. Reading the college details from text files
4. Creating fire-base database
5. Replacing flutter variables
6. Generate flutter application

The modules used for creating python script:

The data read from excel is written to fire-base using "fire-base function".

```
def firebase_function(firebase,college_appname,key):

    df2 = pd.read_excel(r'C:\xampp\htdocs\exceluploads\data.xlsx')
    firebase = firebase.FirebaseApplication
    ['https://collegeapp-e02e4-default-rtdb.firebaseio.com', None]
    for i in df2.index:
        a1=str(df2['regno'][i])
        a2=str(df2['branch'][i])
        a3=str(df2['email'][i])
        a4=str(df2['fullname'][i])
        a6=str(df2['rollno'][i])
        result = firebase.patch(str(college_appname)+'/'+a1,
                                {'branch': a2, 'email':a3 , 'fullName':a4,
                                 'rollNo':a6, 'signedin': 'F'})
        result2 = firebase.patch(str(college_appname)+'/'+'validation',
                                {'key': key})
```

Figure 3.12: Fire-base function

The script executes two types of flutter commands at the end. One for generating app icons and one for starting app compilation.

```
process1=subprocess.Popen(["powershell","flutter pub run flutter_launcher_icons:main"]
| | | ,stdout=subprocess.PIPE,);
result1=process1.communicate()[0]
print (result1)

process2=subprocess.Popen(["powershell","flutter build apk --debug"],stdout=subprocess.PIPE);
result2=process2.communicate()[0]
print (result2)
```

Figure 3.13: Python script for icon generation and initiation of APP compilation

3.6.3 Flutter Application

Common social media app structure is used in the flutter application. The dart file named "constants" is used for updating the variables like college name and college details.

```
String collegename = 'input01';
String appname = 'input02';
String about = 'input03';
String madebecause = 'input04';
String library = 'input05';
String collegeweb = 'input06';
String chatusers = appname + 'messages';
String messages = appname + 'messages';
String userdata = appname + 'users';
```

Figure 3.14: Dart file named 'constants'

21 different dependencies are used in flutter application. Upcoming updates in these dependencies may produce dependency errors which is common in all flutter applications.

```
dependencies:  
  flutter:  
    | sdk: flutter  
  cupertino_icons: ^1.0.0  
  firebase_core: ^0.3.4  
  comment_box: ^0.0.15  
  firebase_auth: ^0.8.4+5  
  google_sign_in: ^4.4.1+3  
  firebase_database: ^2.0.3  
  firebase_storage: ^2.1.0+1  
  cloud_firestore: ^0.9.13+1  
  firebase_messaging: ^4.0.0+3  
  shared_preferences: ^0.4.3  
  validate:  
  image_picker: ^0.6.6+5  
  cached_network_image: ^2.0.0-rc  
  url_launcher: ^4.2.0+1  
  timeago: ^2.0.10  
  path_provider: ^0.4.1  
  http: ^0.12.0  
  provider: ^5.0.0  
  intl:  
  webview_flutter: ^2.0.4  
  flutter_local_notifications: ^0.6.1
```

Figure 3.15: Dependencies used in flutter application

The pages present in flutter application are :

```
> about  
> activity  
> blog  
> developer  
> home  
> library  
> notifications  
> profile  
> saved_images
```

Figure 3.16: Pages

CHAPTER 4

RESULTS & DISCUSSION

We have described a method for automated generation of a social media application for educational institutions. Completed the designing of the social media application and the DFD, UML, ER etc required for developing such an application and the website for the automatic generation of the application.

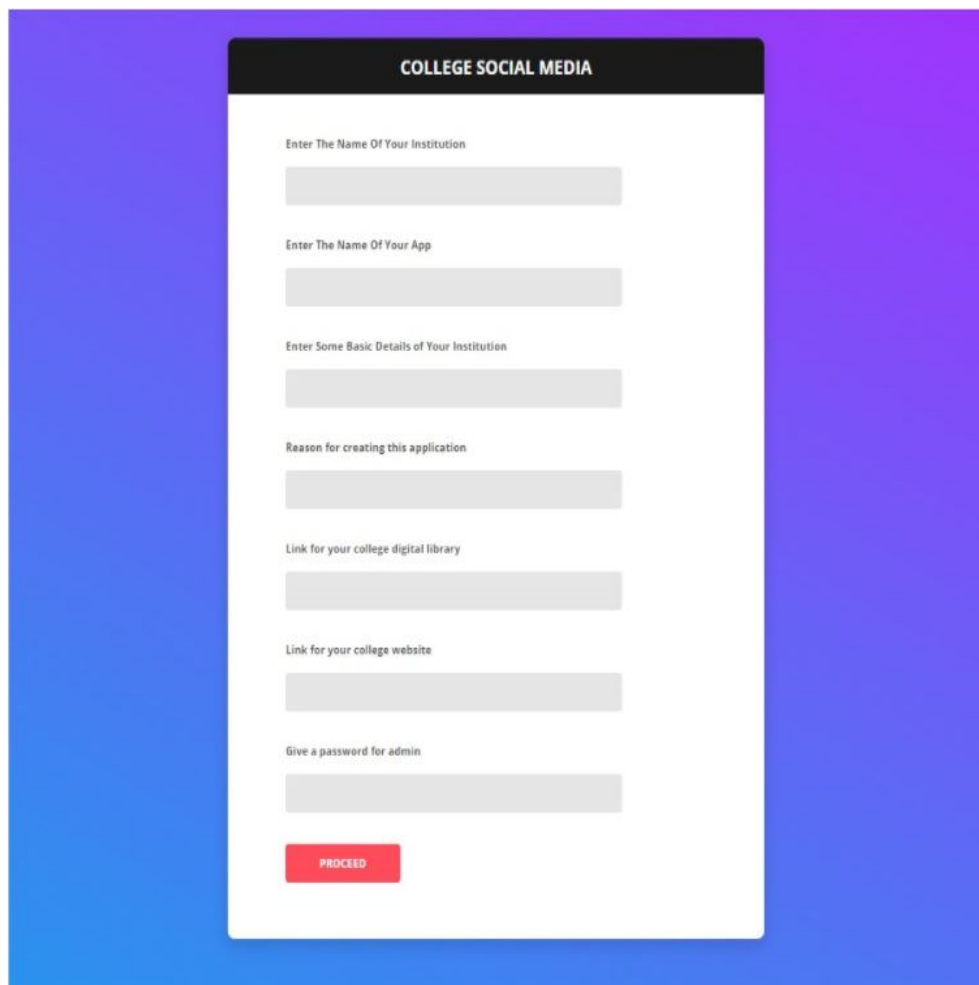
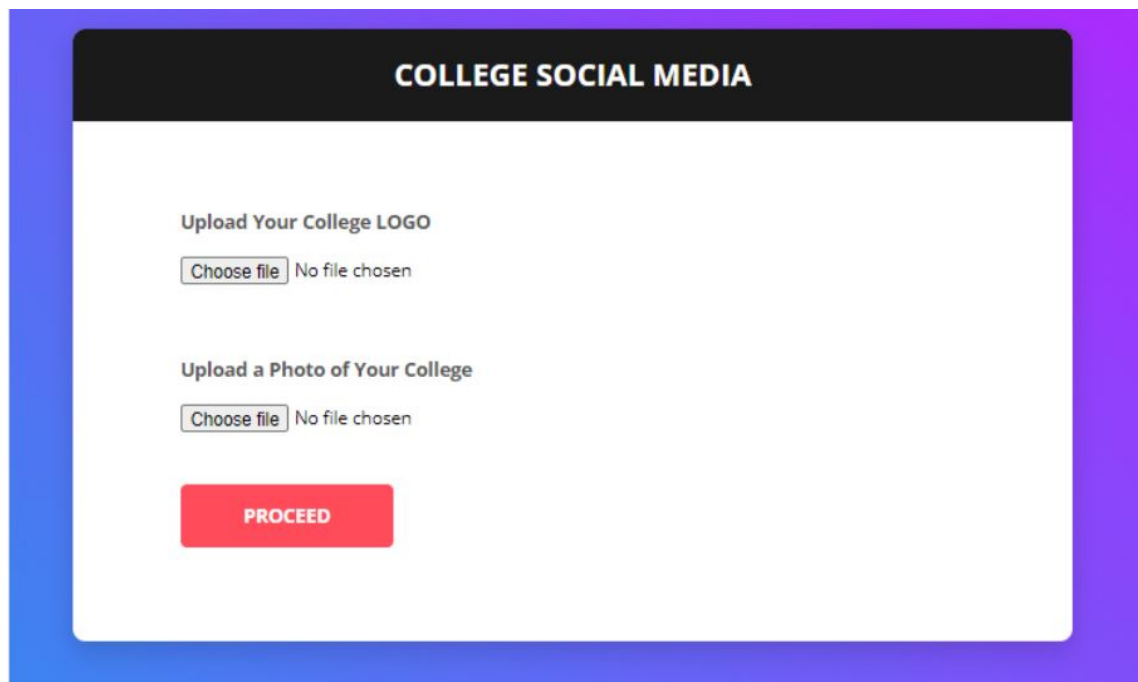
The image shows a web form titled "COLLEGE SOCIAL MEDIA" set against a blue-to-purple gradient background. The form itself is white with a black header bar containing the title. It contains several text input fields, each preceded by a label: "Enter The Name Of Your Institution", "Enter The Name Of Your App", "Enter Some Basic Details Of Your Institution", "Reason for creating this application", "Link for your college digital library", "Link for your college website", and "Give a password for admin". At the bottom of the form is a red button with the word "PROCEED" in white capital letters.

Figure 4.1: Page to enter the college details for social media application

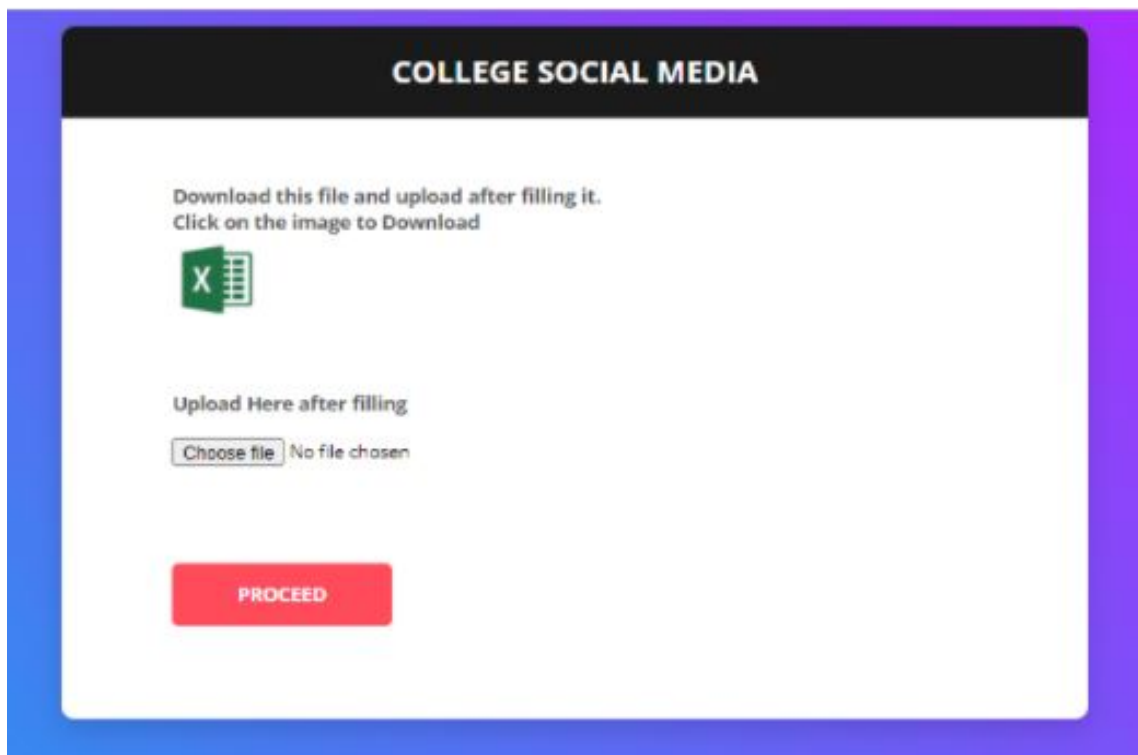
We built a website for making app using HTML, CSS, Java-script etc. From the website collect data for generating app, data like name of the institution, name of the app, basic details of the institution, reason for creating app, [6] link for institution website and institution's digital library, and one password specific for admin.



The screenshot shows a web form titled "COLLEGE SOCIAL MEDIA" with a black header. The form has a white background and is set against a blue and purple gradient border. It contains two upload sections: "Upload Your College LOGO" and "Upload a Photo of Your College". Each section has a "Choose file" button and the text "No file chosen". At the bottom, there is a red "PROCEED" button.

Figure 4.2: Upload college logo and image

Also user should upload excel spreadsheet and image and logo of the institution.



The screenshot shows the same "COLLEGE SOCIAL MEDIA" form. It now includes a section with the text "Download this file and upload after filling it. Click on the image to Download" above a green Excel icon. Below this is an "Upload Here after filling" section with a "Choose file" button and "No file chosen" text. The red "PROCEED" button remains at the bottom.

Figure 4.3: Download the excel file, fill it and upload here

By clicking the generate application button the automated process will begin in the background.

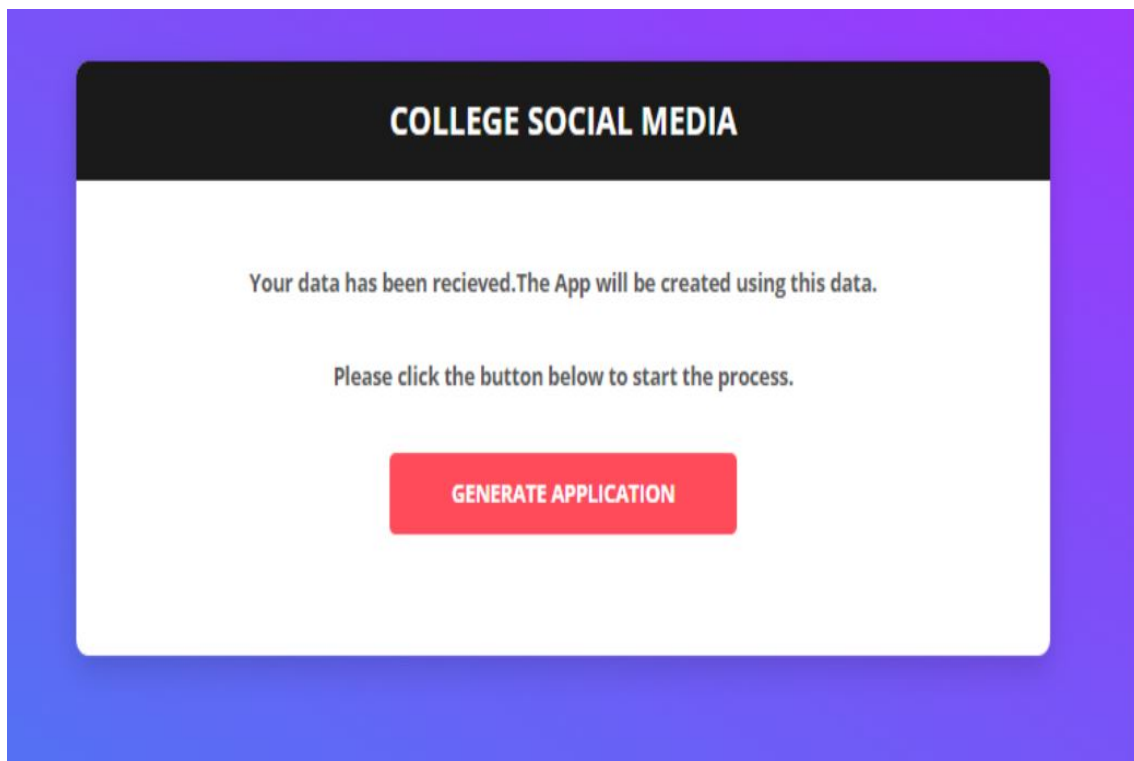


Figure 4.4: Click Generate Application for creating APK

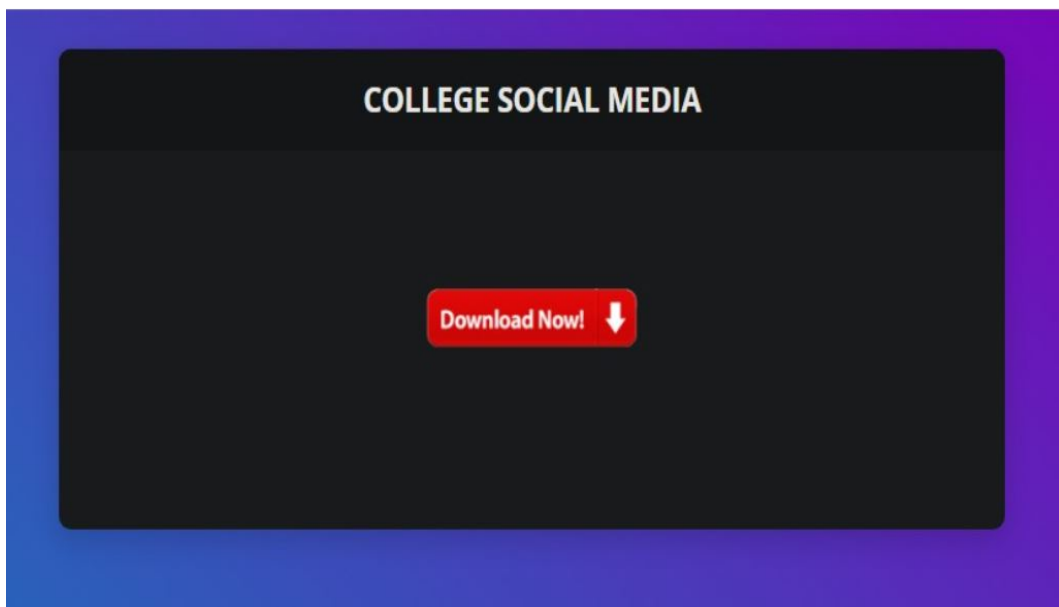


Figure 4.5: For downloading the APK file

Given details are stored into the fire-base database using python script. Finally app is generated by flutter service. APK can be downloaded by the user.

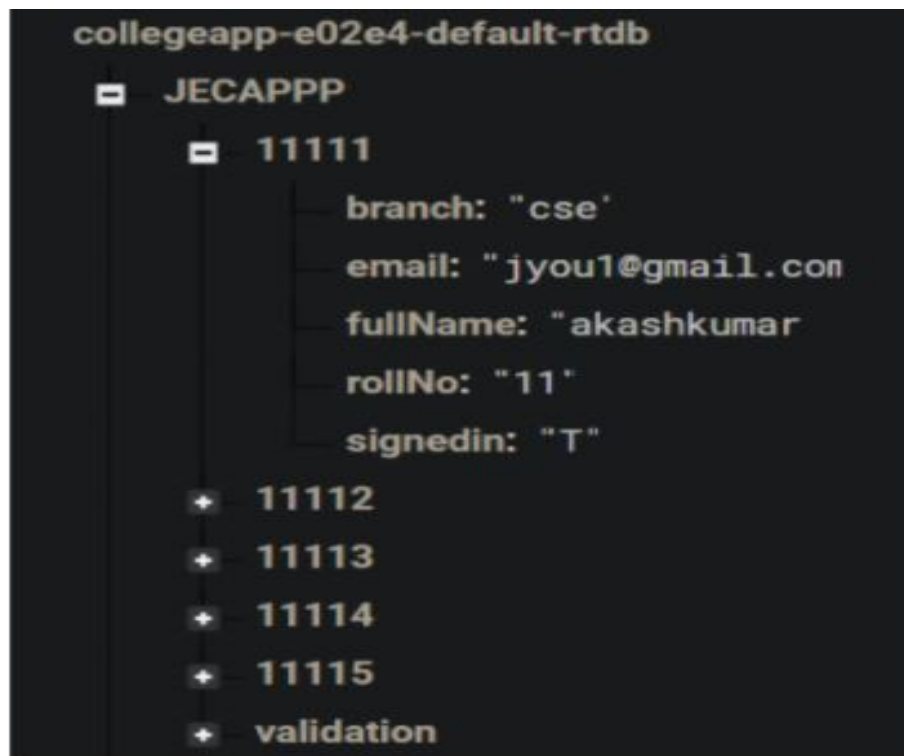


Figure 4.6: Real-time database

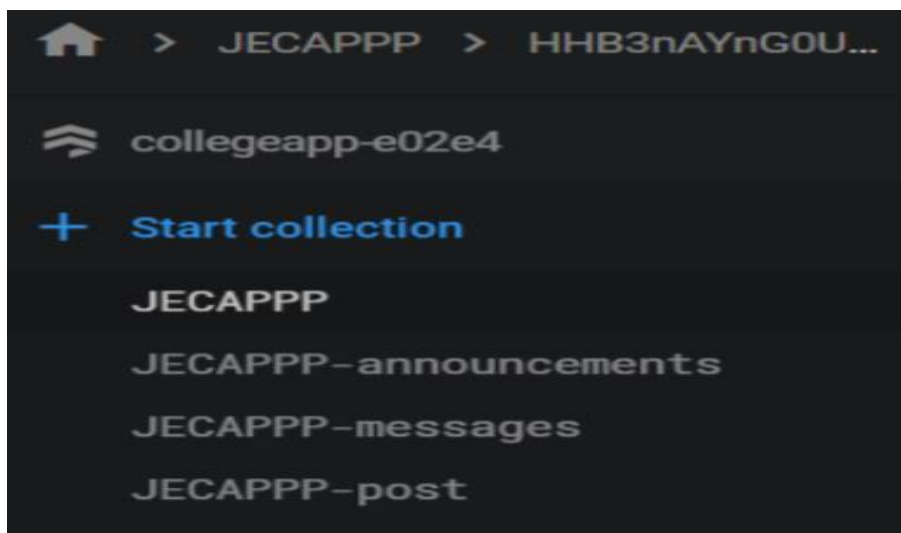


Figure 4.7: Fire-store database

User can login to the app by entering login details which are given in the excel file. Other features like blog, website, digital library, chat, profile, theme change etc. are available in this app. New user have to register by giving the password. Existing user can login by registered credentials.

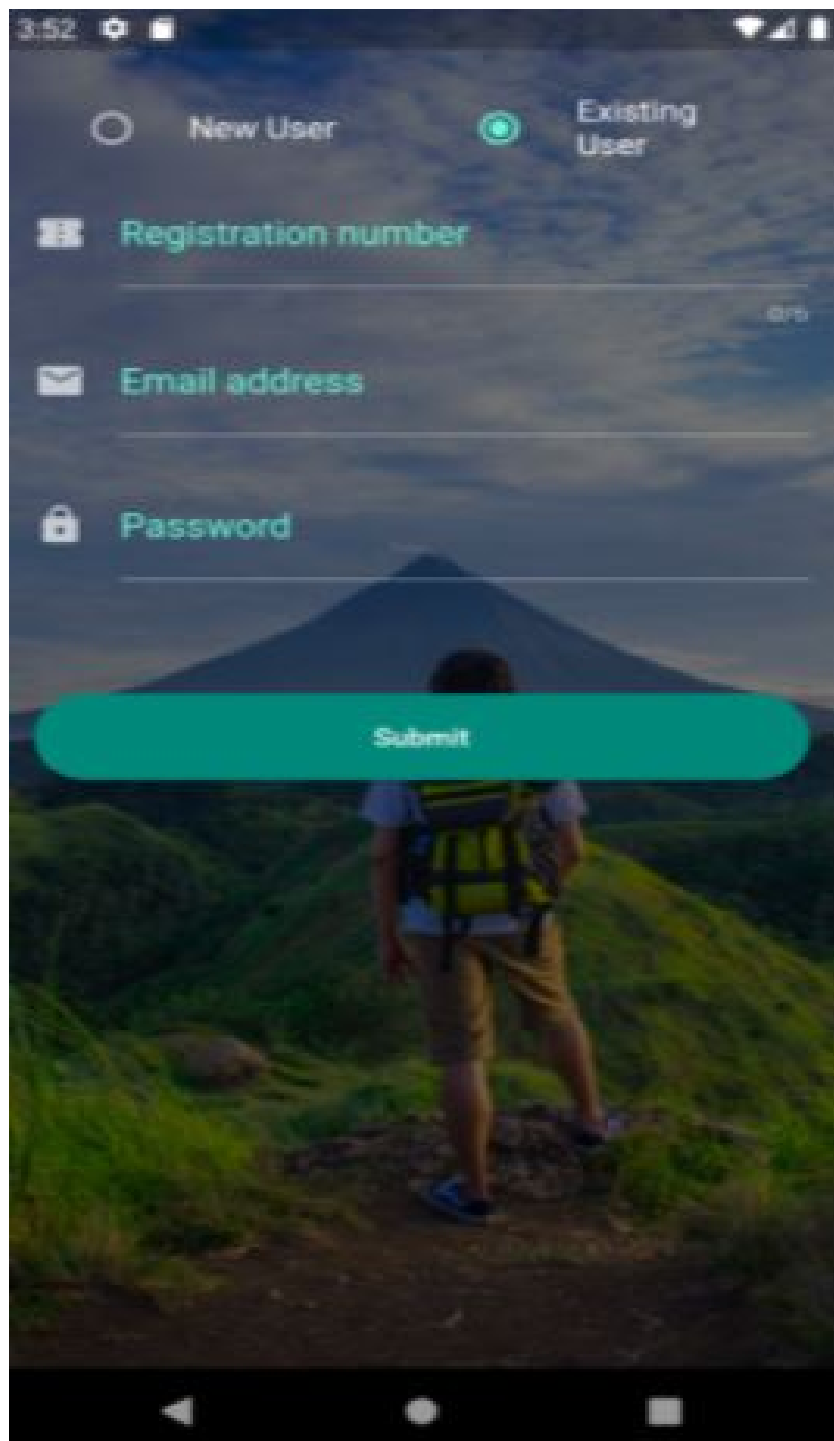


Figure 4.8: Login Page

Home Page displays the announcement page and announcement making page. Only those who have the admin password can make an announcement. Other can only view the announcement. The time of each announcements can also be seen in the page.

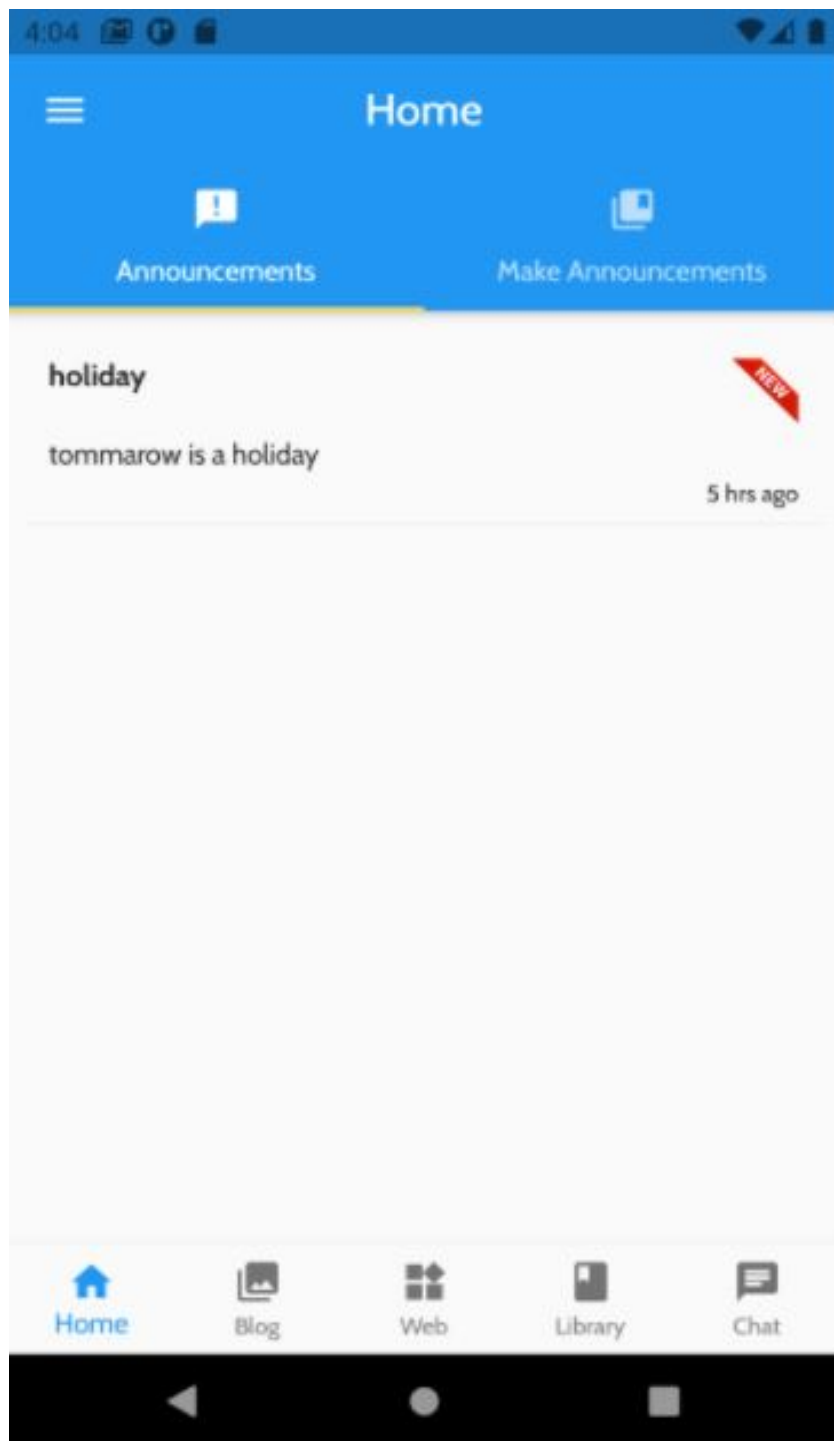


Figure 4.9: Announcements can be read here

A pop-up for adding the announcement will appear only if the password is correct.

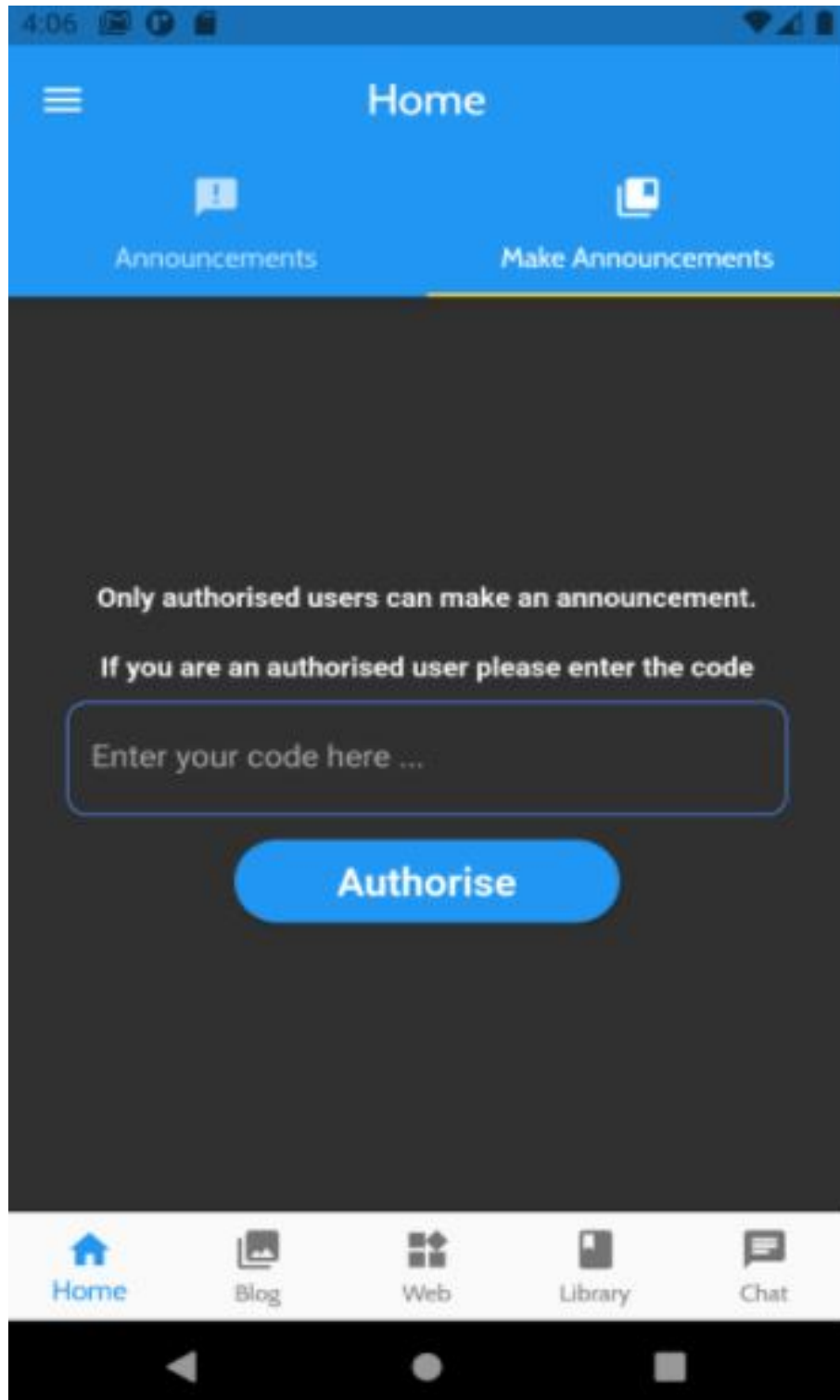
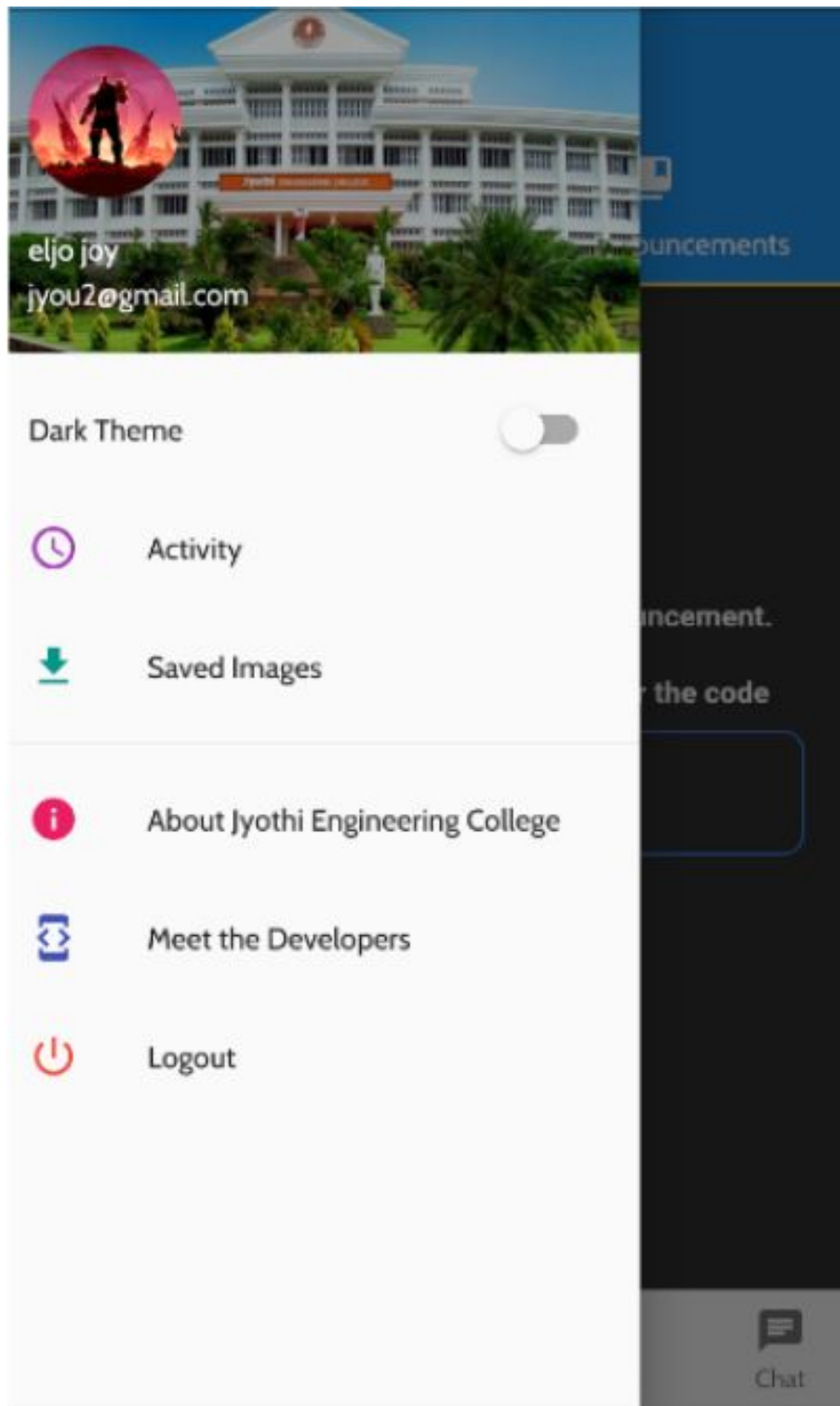


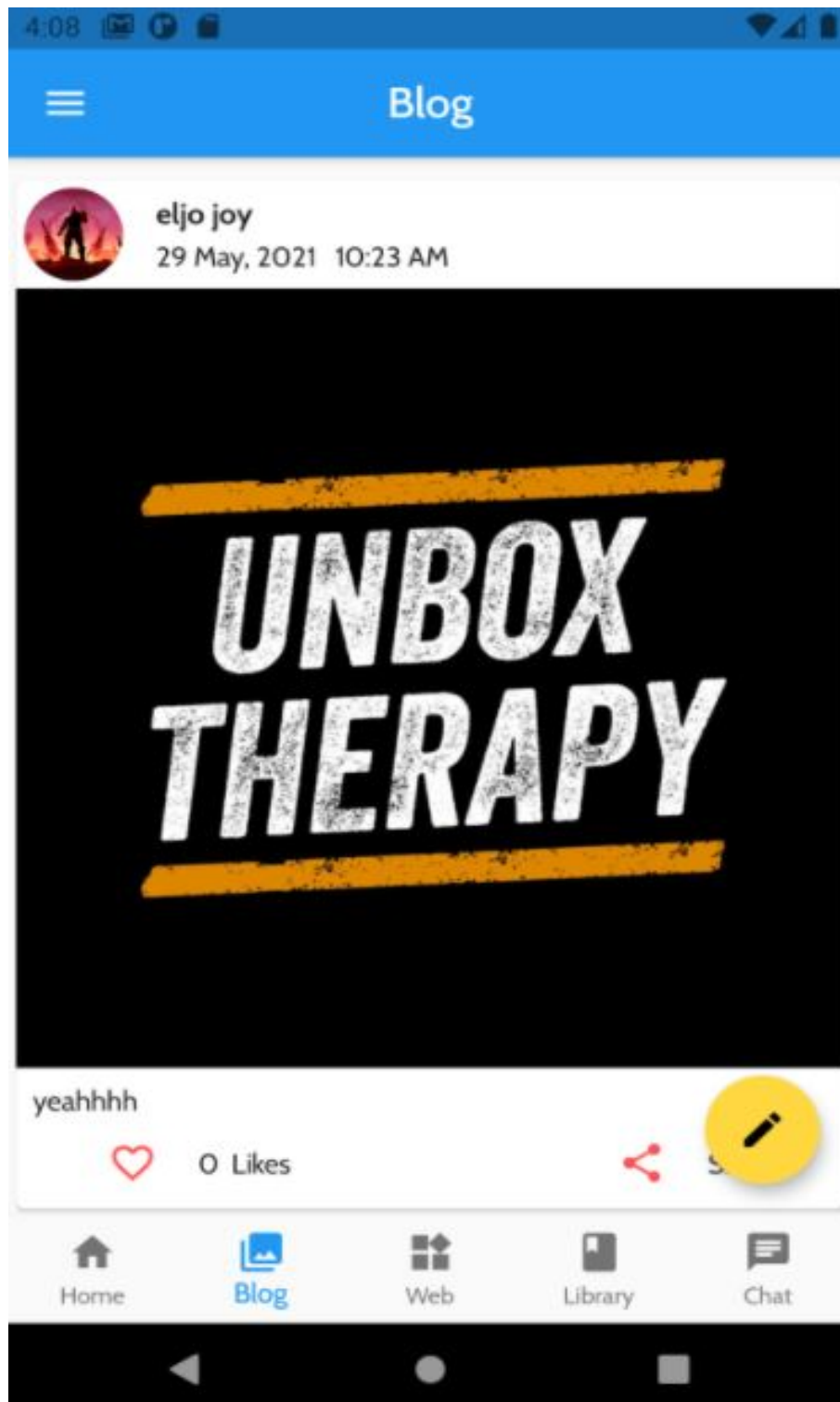
Figure 4.10: Announcements can be made here using password for admin

Sidebar Contents



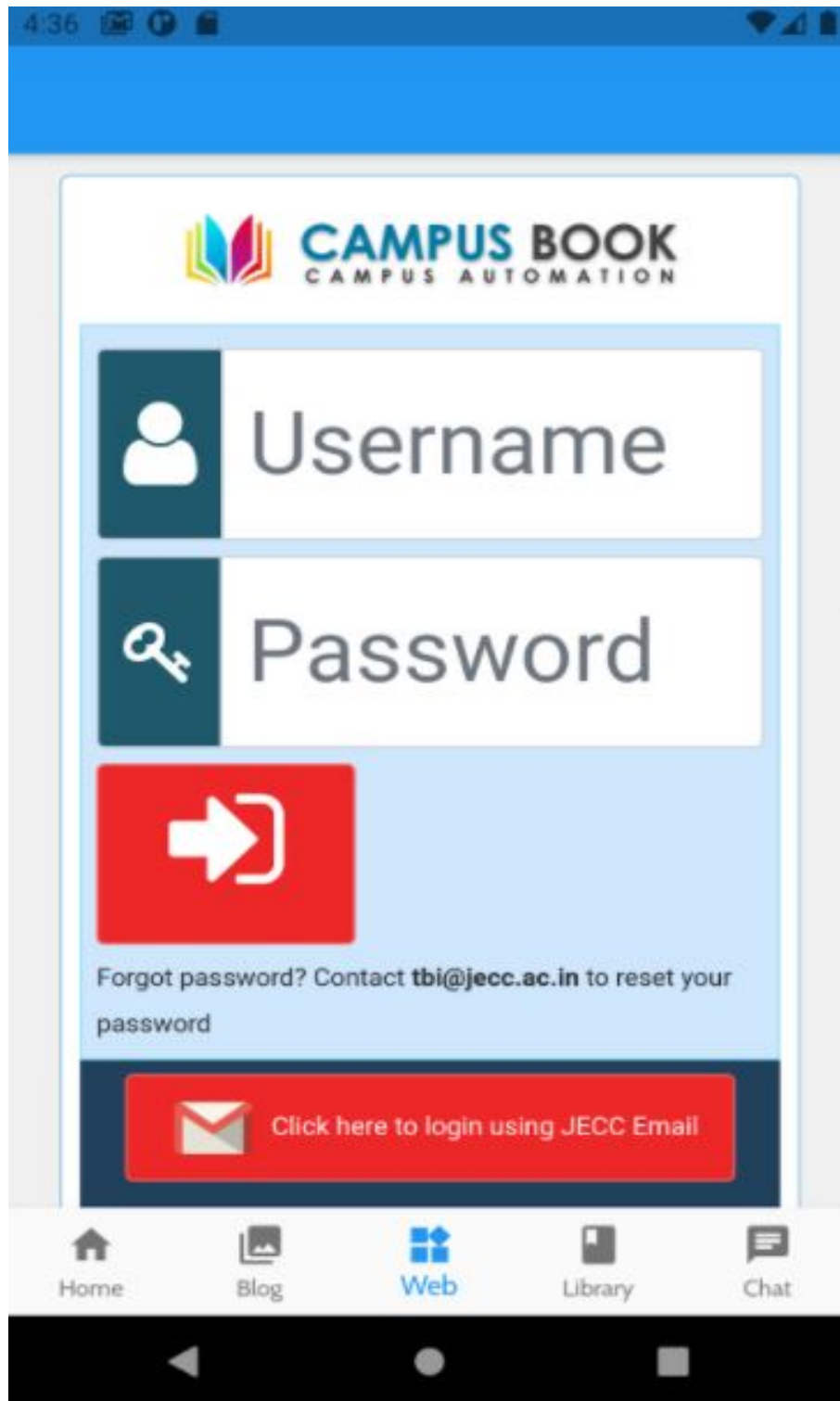
In this page we can change theme, saved images can be visit, description about the college and about developers can be seen here. In order to exit the app here logout option is provided.

Blog Page



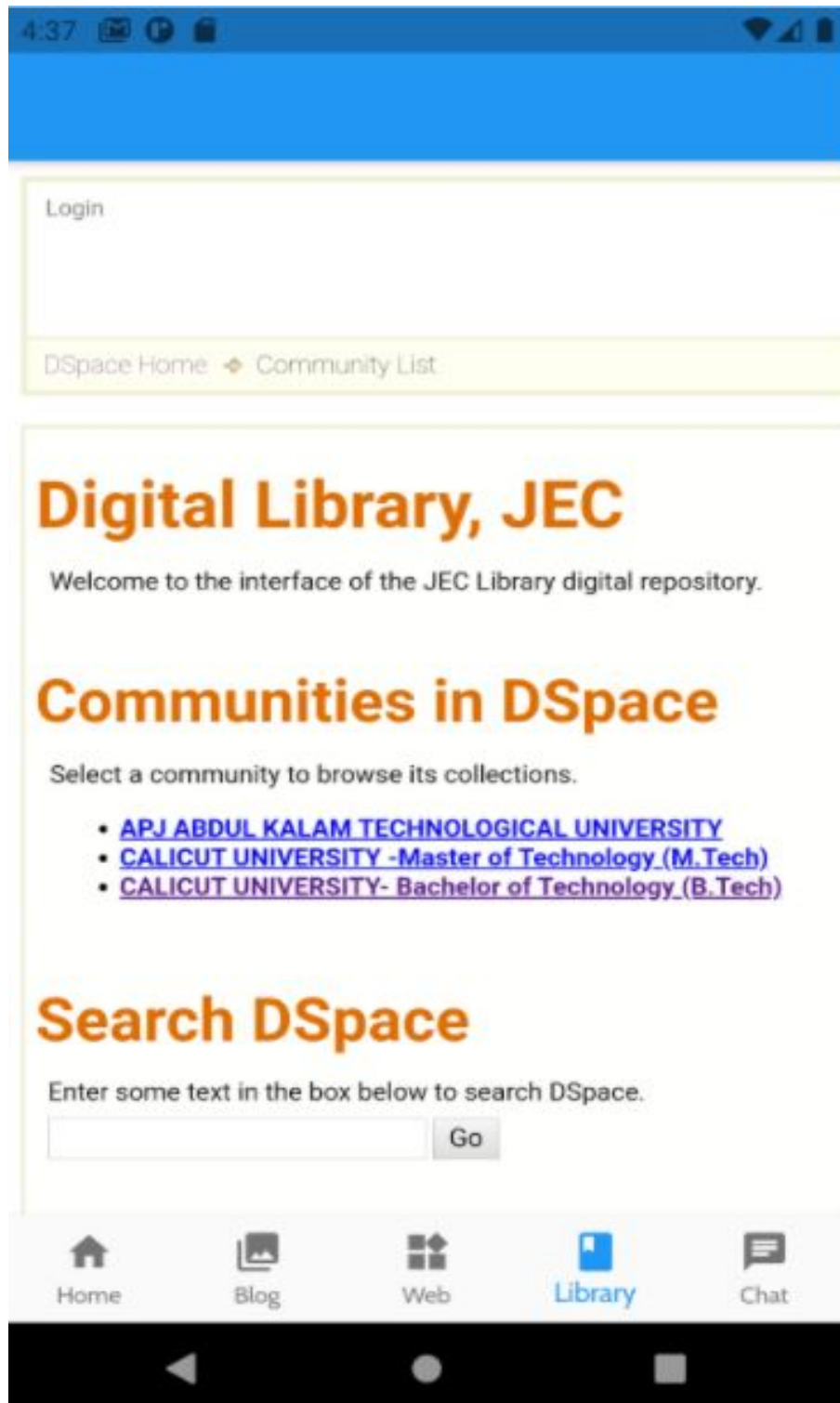
This is the output of the blog page. Here we can post images and text by clicking the pen icon which can be viewed by all others in the app. So others can comment, like and share etc. Time and date of the post is also shown here.

Institution Website



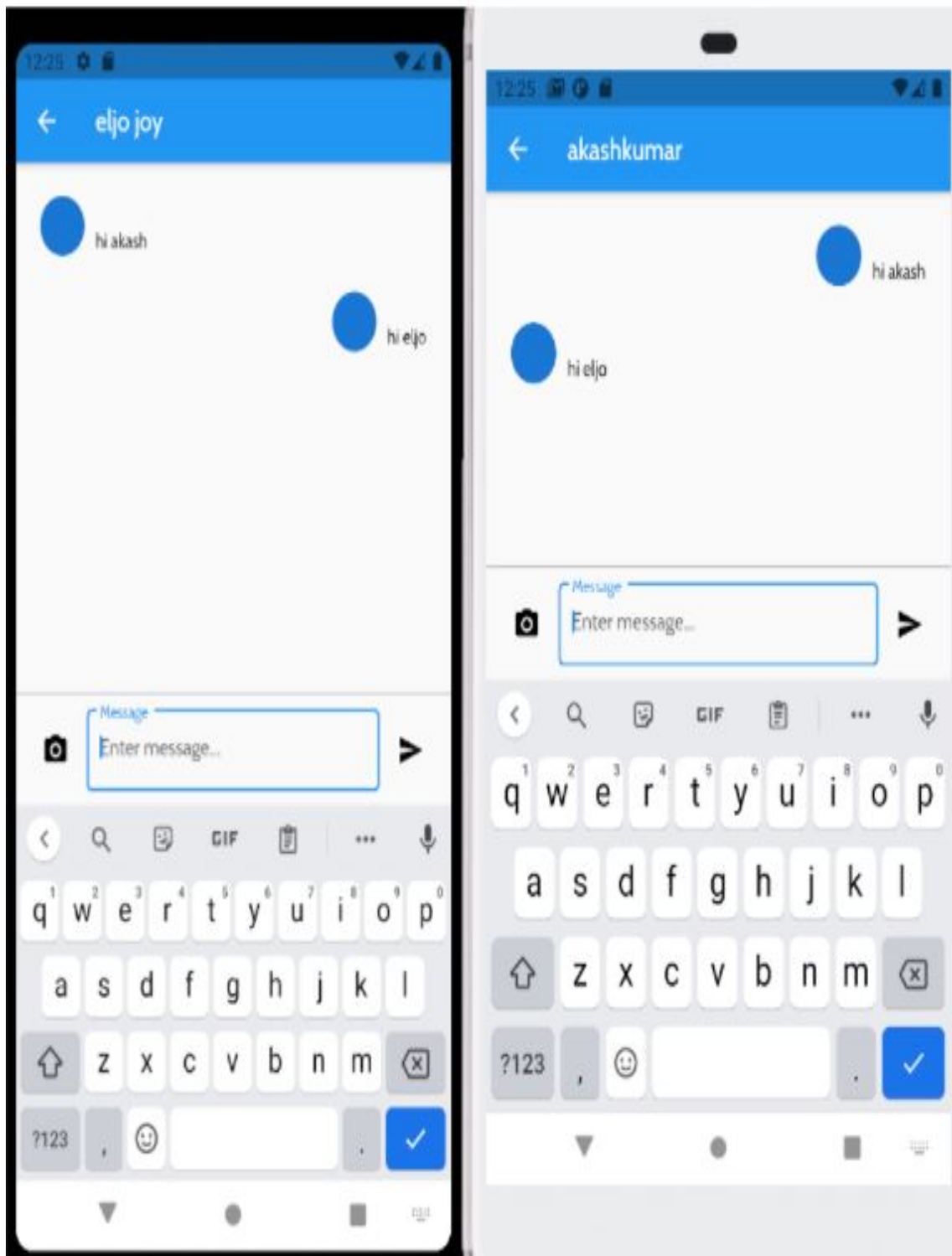
The page provides a web view of the college website link given at the beginning. Users can interact in the web view, just like the browser.

College Digital Library



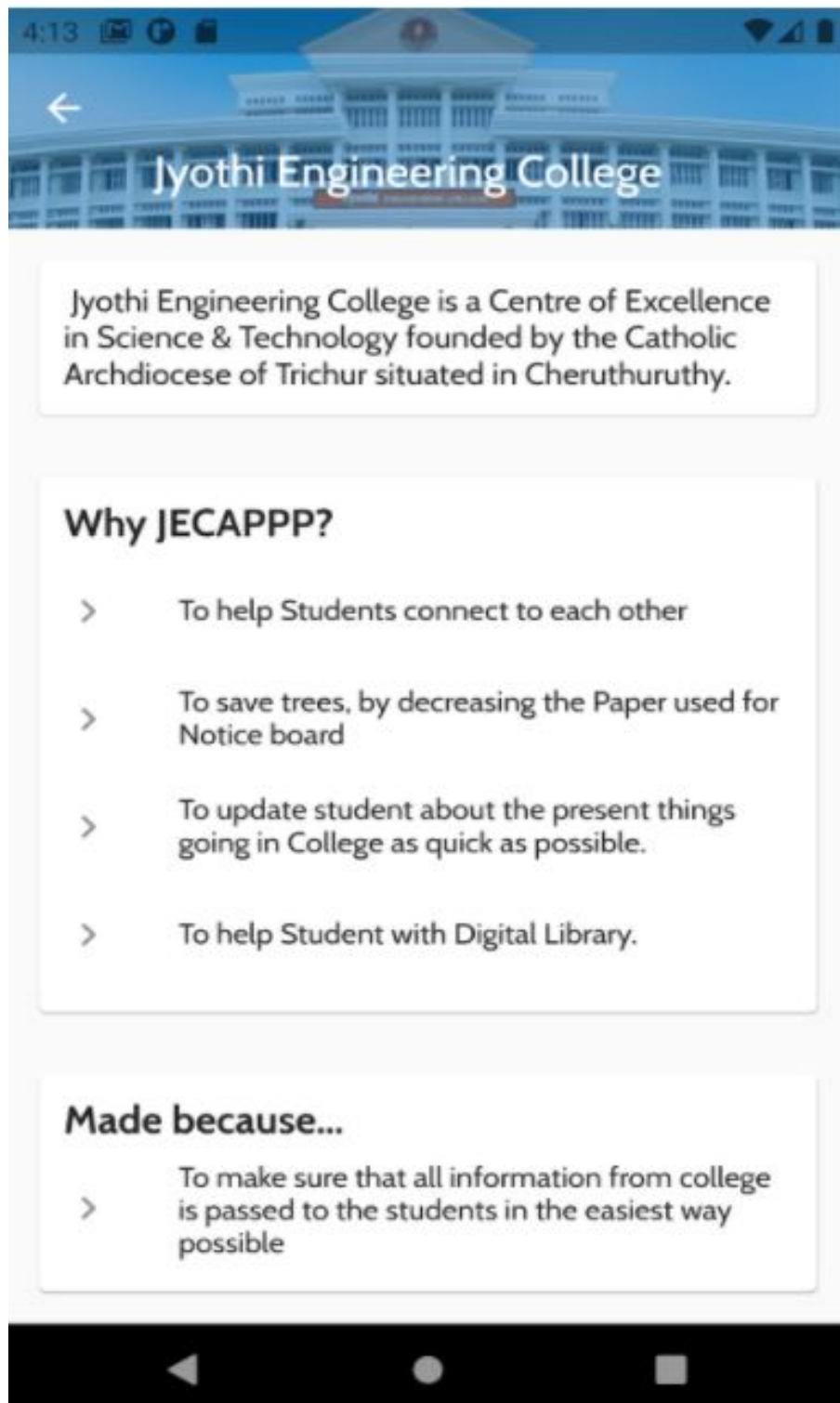
The page provides a web view of the college library link given at the beginning. Users can access their library details through this web view.

Chat Page



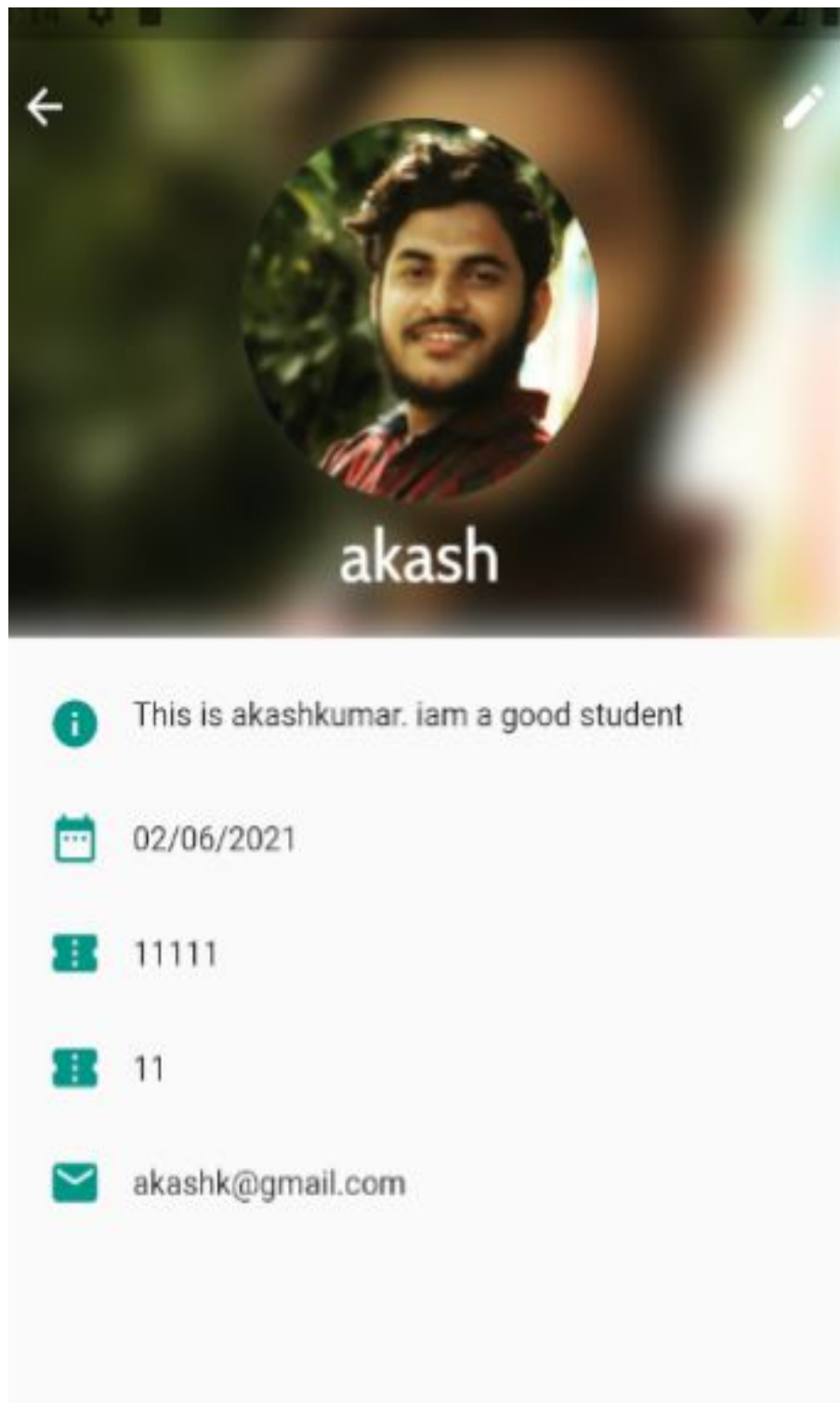
In this page users can communicate with other users. There will not be any latency between the communication. User can also share images.

About Page



About page displays the details of the college that was given at the beginning. This page will be same for all users.

Profile Page



This page displays the details of the user. User can also update their profile here.

CHAPTER 5

CONCLUSION & FUTURE SCOPE

5.1 Conclusion

Nowadays the interaction between students and faculties are diminishing. Social Media plays a crucial role in connecting people. An independent social media for institution will help to develop a better social environment and social skill among members of the institution. So this application will provide an environment for interaction and improve the communication between student-student and student-faculty relationships.

Our project aimed at reducing the time and cost for developing a social media application for an institution. We used python and flutter to develop an automatic application generating system based on the data collected from the client. As a result, application is formed and it contains features such as updating story, chat, accessing college updates from the admin, attendance details etc.

This is a closed social media application which can be generated instantaneously from the website which we developed, so others who are not permissible by the institution cannot invade directly in to this application. So it becomes more secure to handle. Students and faculty members can communicate each other, also get important announcements from the authority and websites related to the institution can be accessed through this single app. And the client will be able to go through the whole process easily. Even a person with less technical knowledge will be able to use our service and create a social media application for their institution. Updates can be done by the person in-charge of the institutions. Other interruptions such as advertisement can be avoided. So, there will be an increased privacy of user's data.

This service can be used for other institutions apart from educational by slight modifications. In real word application it is preferable to host the service in cloud servers such as AWS, GCP etc.

5.2 Future Scope

In future a separate faculty and student login page is planned and additional feature like story, comment, like etc can be added to the application to make it better. Also a separate announcement page for admins and a module in website for users to update database so that new members can access the application. Configuration of the app for iOS is also pending. Currently chat system provides basic chat option only but provision for upload photos, videos and documents in the chat system is being planned and need to be developed in future. These features will help to increase the usability of the service.

Bibliography

- [1] Erik Blokland. 2019. *"An empirical evaluation of the user interface energy consumption of react-native and flutter."*. 2019.
- [2] *"Structure chart."*Empirical study of drone sound detection in real-life environment with deep neural networks <https://www.google.com/url?sa=iurl=http>.
- [3] *"Er diagram for college management system."*. [https://www.google.com/url?sa=iurl=httpsJuhyun Kim, Cheonbok Park, Jinwoo Ahn, Youlim Ko, Junghyun Park, and John C Gallagher](https://www.google.com/url?sa=iurl=httpsJuhyun%20Kim,%20Cheonbok%20Park,%20Jinwoo%20Ahn,%20Youlim%20Ko,%20Junghyun%20Park,%20and%20John%20C%20Gallagher)
- [4] Philipp Brune. InWEBIST, pages 235–242, 2017. *"Simulating user interactions: A model and tool for semi-realistic load-testing of social app backend web services."*. Benjamin Knoedler, Reda Zemmari, and Wolfgang Koch.
- [5] Anmol Khandeparkar, Rashmi Gupta, and B Sindhya. *"An introduction to hybrid platformmobile application development."*. International Journal of Computer Applications, 118(15),2015.Xuejiao Li and Zixuan Zhou.
- [6] PS Lokhande, Fankar Aslam, Nabeel Hawa, Jumal Munir, and Murade Gulamgaus. *"Effi-cient way of web development using python and flask. 2015. "*.
- [7] Melvyn Zhang, Enquan Cheow, Cyrus SH Ho, Beng Yeong Ng, Roger Ho, and Christo-pher Cheng Soon Cheok. *"Application of low-cost methodologies for mobile phone appdevelopment."*. JMIR mHealth and uHealth, 2(4):e55, 2014.

APPENDIX A

SCREENSHOTS

Home Page

```
_getPostFromDatabase() async {
  Firestore.instance
    .document(widget.postReference)
    .get()
    .then((DocumentSnapshot documentSnapshot) {
      setState(() {
        _isLoading = true;
        postSnapshot = documentSnapshot;
      });
    });
});

@override
Widget build(BuildContext context) {
  final appState = AppState.of(context);
  return Scaffold(
    appBar: AppBar(
      title: Text('Your Post'),
    ), // AppBar
    body: (_isLoading)
      ? ListView(
        children: <Widget>[
          SinglePost(
            FUID: appState.firebaseUser.uid,
            document: postSnapshot,
          ), // SinglePost
        ], // <Widget>[]
      ) // ListView
      : Center(child: CircularProgressIndicator()),
  ); // Scaffold
}
```

Figure A.1: Code for home page

Announcement Page

```

@override
Widget build(BuildContext context) {
  final appState = AppState.of(context);
  return Container(
    padding: const EdgeInsets.all(8.0),
    child: new RefreshIndicator(
      color: appColors.refreshColor,
      onRefresh: _onRefresh,
      child: StreamBuilder(
        stream: Firestore.instance
          .collection(appname + '-announcements')
          .orderBy('timestamp', descending: true)
          .snapshots(),
        // .snapshots(),
        builder:
          (BuildContext context, AsyncSnapshot<QuerySnapshot> snaps
            if (!snapshot.hasData) {
              print("nop");
              return const Center(child: CircularProgressIndicator());
            } else {
              print(snapshot.data.documents);

              return ListView(
                padding: EdgeInsets.all(0.0),
                children:
                  snapshot.data.documents.map((DocumentSnapshot snaps
                    print("object");
                    if (snapshot['type'] == 'all' ||
                      snapshot['type'] == appState.user.type) {
                      print('heree');
                      return SingleAnnouncement(snapshot: snapshot);
                    }
                  )).toList(),
                ); // ListView
              }
            ), // StreamBuilder
          ), // RefreshIndicator
        ); // Container
      }
    }
  }

```

Figure A.2: Code for Announcement

Sidebar Contents

```
@override
Widget build(BuildContext context) {
  appState = AppState.of(context);
  return Drawer(
    child: ListView(
      padding: EdgeInsets.zero,
      children: <Widget>[
        UserAccountsDrawerHeader(
          accountEmail: Text(appState.firebaseUser.email),
          accountName: Text(appState.user.fullName),
          currentAccountPicture: GestureDetector(
            onTap: () => methods.sendTo(
              appState: appState,
              context: context,
              page: ProfilePage(),
            ),
            child: Hero(
              tag: 'profile-pic',
              child: CircleAvatar(
                backgroundImage:
                  CachedNetworkImageProvider(appState.user.photoUrl),
              ), // CircleAvatar
            ), // Hero
          ), // GestureDetector
          decoration: BoxDecoration(
            image: DecorationImage(
              image: AssetImage('assets/images/college.jpg'),
              fit: BoxFit.cover,
              colorFilter: ColorFilter.mode(Colors.black38, BlendMode),
            ), // DecorationImage
          ), // BoxDecoration
        ), // UserAccountsDrawerHeader
        ListTile(
          title: Text('Dark Theme'),
          trailing: Switch(
            value: appState.isDarkThemeEnabled,
            onChanged: (bool b) {
              methods.savePrefs(saveInName: 'isDarkThemeEnabled', dat
              appState.changeTheme();
            },
          ),
        ),
      ],
    ),
  );
}
```


Blog Page

```
@override
Widget build(BuildContext context) {
  final appState = AppState.of(context);
  return Scaffold(
    body: NestedScrollView(
      controller: _scrollController,
      headerSliverBuilder: (BuildContext context, bool isScrolled) {
        return <Widget>[
          SliverAppBar(
            leading: appMenuButton(widget.openDrawerFunction),
            centerTitle: true,
            title: const Text('Home', style: titleStyle),
            pinned: true,
            floating: true,
            forceElevated: true,
            bottom: TabBar(
              controller: _tabController,
              tabs: <Widget>[
                Tab(
                  text: 'Announcements',
                  icon: const Icon(Icons.announcement),
                ), // Tab
                Tab(
                  text: 'Make Announcements',
                  icon: const Icon(Icons.collections_bookmark),
                ), // Tab
              ], // <Widget>[]
            ), // TabBar
          ), // SliverAppBar
        ]; // <Widget>[]
      },
      body: TabBarView(
        controller: _tabController,
        children: <Widget>[
          AnnouncementsPage(),
          TestMe(),
        ], // <Widget>[]
      ), // TabBarView
    ), // NestedScrollView
  ); // Scaffold
}
```

College Digital Library

```
class LibraryPage extends StatefulWidget {
  LibraryPage({Key key, this.openDrawerFunction}) : super(key: key)
  final VoidCallback openDrawerFunction;
  final url = library;

  @override
  createState() => LibraryPagest(this.url);
}

class LibraryPagest extends State<LibraryPage> {
  @override
  var url;
  final _key = UniqueKey();
  LibraryPagest(this._url);
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(),
      body: Column(
        children: [
          Expanded(
            child: WebView(
              key: _key,
              javascriptMode: JavascriptMode.unrestricted,
              initialUrl: _url)) // WebView // Expanded
        ],
      )); // Column // Scaffold
  }
}
```

Chat Page

```

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text("All Users"),
    ), // AppBar
    body: userslist != null
      ? Container(
        child: ListView.builder(
          itemCount: usersList.length,
          itemBuilder: ((context, index) {
            return ListTile(
              //leading: CircleAvatar(
              //  backgroundImage:
              //    NetworkImage(photo[index].data['photoUrl'])
              // ),
              title: Text(usersList[index].data['name'],
                style: TextStyle(
                  color: Colors.black,
                  fontWeight: FontWeight.bold,
                )), // TextStyle // Text
              subtitle: Text(usersList[index].data['emailId'],
                style: TextStyle(
                  color: Colors.grey,
                )), // TextStyle // Text
              onTap: (() {
                Navigator.push(
                  context,
                  new MaterialPageRoute(
                    builder: (context) => ChatScreen(
                      name: usersList[index].data['name']
                      photoUrl: photo[index].data['photoU
                      receiverUid:
                        usersList[index].data['uid']
                    )),
                )),
            ); // ListTile
          )),
        ), // ListView.builder
      ) // Container
      : Center(
        child: CircularProgressIndicator(),
      ); // Center // Scaffold
}

```

About Page

```
@override
Widget build(BuildContext context) {
  appState = AppState.of(context);
  return Scaffold(
    body: CustomScrollView(
      slivers: <Widget>[
        SliverAppBar(
          pinned: true,
          expandedHeight: 200.0,
          flexibleSpace: FlexibleSpaceBar(
            title: Text(collegename),
            background: Image.asset(
              'assets/images/college.jpg',
              fit: BoxFit.cover,
              color: Colors.black26,
              colorBlendMode: BlendMode.darken,
            ), // Image.asset
          ), // FlexibleSpaceBar
        ), // SliverAppBar
        SliverList(
          delegate: SliverChildListDelegate(
            <Widget>[
              _infoCard(),
              _whyMyAppCard(),
              _madeBecause(),
              _gotoDevelopers(),
            ], // <Widget>[]
          ), // SliverChildListDelegate
        ), // SliverList
      ], // <Widget>[]
    ), // CustomScrollView
  ); // Scaffold
}
```

Profile Page

```
class _ProfilePageState extends State<ProfilePage> {}  
  
@override  
Widget build(BuildContext context) {  
  final appState = AppState.of(context);  
  return Scaffold(  
    body: CustomScrollView(  
      slivers: <Widget>[  
        SliverAppBar(  
          expandedHeight: 250.0,  
          floating: false,  
          pinned: false,  
          actions: <Widget>[  
            IconButton(  
              icon: const Icon(Icons.create),  
              tooltip: 'Edit Profile',  
              onPressed: () => Navigator.push(  
                context,  
                MaterialPageRoute(  
                  builder: (context) => AppState(  
                    firebaseUser: appState.firebaseUser,  
                    user: appState.user,  
                    child: EditProfilePage(),  
                  ), // AppState  
                ), // MaterialPageRoute  
              ),  
            ) // IconButton  
          ], // <Widget>[]  
          flexibleSpace: FlexibleSpaceBar(  
            centerTitle: true,  
            background: Hero(  
              tag: 'profile-pic',  
              child:  
                BackgroundImageWithCircularAvatar(appState.user.photo),  
            ), // Hero  
            title: Text((appState.user.displayName == "")  
              ? 'My Profile'  
              : appState.user.displayName), // Text  
          ), // FlexibleSpaceBar  
        ), // SliverAppBar  
      ],  
    ),  
  );  
}
```