1. INTRODUCTION

In institutions, there are still many time-consuming tasks and problems that can be overthrown with the help of technology. As the students have to spend a lot of time going through all these processes and the physical availability of faculty is always required. In order to conquer this problem, we introduce our Student's Cell. Using this system student's can get information, seek help in emergency situation, contribute their innovative ideas in the institution and also can chat with the admin. This application consists of two main modules such as USER and ADMIN . The main purpose is to create an android application which will manage the working of these different modules for the student's.

The student's can register through this app, and the admin will just have to acknowledge the form. This process won't require the physical presence of students as well as admin. This process can help manage all the data in a more organized manner and make it easier. The main lead for using this system is that the faculty/admin need not be physically present at their respective places and students can express their feelings easily, also can gather information's about their needs. The system can be accessed irrespective of time or place. The proposed software will also reduce the fearsome in students and can help them with various problems that are related to institution.

1.1 PROJECT OVERVIEW

In this modern age, student's management has become very essential component in education. With the help of College Student's and Admin through android application we can gather all the useful information and helps needed for students. Our System manages the two most important

module such as user(student) module and admin module. Admin plays a major role in this app.

Student's can get into the application and ask for the needful help or provide with the needful information about the institution. The main Ideas we would like to introduce and implement in our institution through the cell are Fund Raising, Requirements, Academic Calendar, Idea Corner, Chat Box and Notifications for student's. Also, the system allows the Students to update their profile. Admin can log into the admin side and can go through every details of the system. An admin can approve any registration, fund details, requirement details, idea implementation as well as remove them. In this system, admin can verify, add queries and also can view every details. Admin is the bridge between students and Students Cell. Information in the cell can be given not only by the admin but also by the Users, under one condition that is: Admin has to approve the users information/issue only then it will be published in the cell.

1.2 BACKGROUND STUDY

As the name "STUDENTS CELL" suggest the application will manage the different functions which are necessary for students. Education system in India has become so advanced due to the development of the technology. But still there is no platform for student's to express the problems of students in an institution. Everyone have their own creative ideas and thoughts which they would like to execute in their institution,

in case of emergency fund raising situation there are chances of manipulation done by the authorities and also very difficult to track each event date that held in institution. Thus we implement this application in order to overcome these difficulties which may arise at different levels in an organization.

2.1 ARCHITECTURE OF EXISTING SYSTEM

In the existing system all the works are done manually. Students have no place to express their feelings, sometimes it is very difficult to get information and needful help during the time. All the records are maintained on paper file. In this system it is very difficult to find any information about various cultural/technical/sports which is supposed to be held in soon. And it is very difficult to maintain the funding during emergency situations and maintenance of the institution in proper way.

2.2 DRAWBACKS OF EXISTING SYSTEM

- Lack of accountability.
- Less secure.
- Users won't be notified their approval information.
- Without admin, system will crash down.
- Without the participation of student's, the system is incomplete.

Due to all these reasons, we are moving for the proposed system.

3.1 OBJECTIVES

- Students cell helps students to interact with the institution in easy and comfortable way.
- Students cell helps to promote creative thinking skills among students.
- Students cell acts as a platform to raise their problems, discontent and dispute which they face in that institution.
- Students can get instant information about the events which has held in that institution through the academic calender.
- Students can ask for emergency funding issues at anytime from anywhere.
- A chatbot will always be present to respond to the queries asked by the users.
- Notifications regarding the institution and university will be published in the application.

3.2 METHADOLOGY

SPIRAL MODEL

The spiral model is a systems development lifecycle method used for risk management that combines the iterative development process model with elements of the Waterfall model.

It looks like a spiral with many loops the exact number of loops of spiral model is unknown and can vary from project to project each loop of spiral model is called phase of software development process and each phase is divided into four sections:

1. Determining objectives

In this phase, requirements are gathering from students and objectives are identified and analysed at the start of very phase. Once the system or product's objectives, alternatives, and constraints are understood, second phase is performed.

2. Identify and resolve risk

In this phase, all the possible solution for problem are evaluated to get the best possible solution. Then risk associated with the solution is evaluated and get a best possible strategy to solve that. At the end of this phase, prototype is built for the best possible solution.

3. Developement and testing

During this phase, all the implemented features are developed and verified thorough testing. At the end of the this phase, the next version of the software is available.

4. Plan for the next phase

During this phase, the students evaluate the so far developed version of the software. It includes risk identification and monitoring like cost overrun or schedule slippage and after that planning of the next phase is started.

ADVANTAGES:

- The spiral model is perfect for large projects and complex in nature.
- This model has an risk handling ability
- Change in requirements by customers at last phases can handled accurately.

DISADVANTAGES:

- It is very expensive and time taking
- It is not suitable for a simpler and smaller project because of multiple phases.
- It requires more documentation as compared to other models.

3.3 ENVIRONMENT

The minimum requirements for the working of the system are as follows:

- HARDWARE REQUIREMENTS FOR DEVELOPER

- Device name: LAPTOP-HP15s
- Processor : AMD Ryzen3 z
- Memory : 8.00 GB (7.34 GB usable)
- System type : 64-bit operating system, x64-based processor
- Memory: 4 GB
- Hard disk: 1 TB
- Monitor: 18.5” LED
- Keyboard: 104 keys
- Android phone: 4 GB

- SOFTWARE SPECIFICATION

- Front end: Android IDE(Java code)
- Language: Java
- Platform: Android
- Back end: PHP, MySQL
- Operating system: Microsoft Windows 11

➢ FEATURES OF ANDROID STUDIO

- ❖ Android studio is the official integrated development environment for Google's android operating system builds on Jet Brains IntelliJ IDEA software and designed specially for android development. It is available for download for on windows, macOS and Linux based operating system.
- ❖ Android studio offers advanced code completion, refactoring, and code analysis. The Android Emulator installs and starts your apps faster than a real device and allows you to prototype and test your app on various Android device configurations.
- ❖ Android Studio's Apply Changes features lets you push code and resource changes to your running app without restarting your app and, in some cases, without restarting the current activity.
- ❖ We can also simulate a variety of hardware features such as GPS location, network latency, motion sensors, and multi-touch input.
- ❖ Android Studio includes project and code templates that make it easy to add well-established patterns such as a navigation drawer and view pager.
- ❖ Android Studio provides a robust static analysis framework and includes over 365 different lint checks across the entirely your app. Additionally it provides several quicks fixes that help you address issues in various categories, such as performance, security, and correctness, with a single click.
- ❖ When working with XML layout files, Android Studio provides a drag and drop visual editor that makes it easier than ever to create a new layout.
- ❖ Android Studio offers build automation, dependency management, and customizable build configurations. You can configure your

project to include local and hosted libraries, and define build variants that include different code and resources, and apply different code shrinking and app signing configurations.

- When working with XML layout files, Android Studio provides a drag and drop visual editor that makes it easier than ever to create a new layout.

FEATURES OF XAMPP

- XAMPP is a free and open-source cross-platform web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server. Maria DB database, and interpreters for scripts written in the PHP and Perl programming languages. Since most actual web server deployments use the same components as XAMPP, it makes transitioning from a local test server to a live server possible. The most obvious characteristic of XAMPP is the case at which a WAMP webserver stack can be deployed and instantiated. Later some common packaged applications that could be easily installed were provided by Bitnami.
- XAMPP is regularly updated to the latest releases of Apache, Maria DB, PHP and Perl. It also comes with a number of other modules including OpenSSL, phpMyAdmin, MediaWiki, Joomla, WordPress and more. Self- contained, multiple instances of XAMPP can exist on a single computer, and any given instance can be copied from one computer to another. XAMPP has the ability to serve web pages on the World Wide Web. A special tool is provided to password-protect the most important parts of the package. XAMPP also provides support for creating and manipulating databases in Maria DB and SQLite among others. Once XAMPP is installed, it is possible to treat a localhost like a remote host by connecting using an FTP client.

➢ FEATURES OF MYSQL

- MySQL is a relational database management system (RDBMS) based on the SQL (Structured Query Language) queries. It is one of the most popular languages for accessing and managing the records in the table. MySQL is open-source and free software under the GNU license. Oracle Company supports it.
- The following are the most important features of MySQL
- Relational Database Management System (RDBMS): MySQL is a relational database management system. This database language is based on the SQL queries to access and manage the records of the table.
- Easy to use: MySQL is easy to use. We have to get only the basic knowledge of SQL. We can build and interact with MySQL by using only a few simple SQL statements.
- It is secure: MySQL consists of a solid data security layer that protects sensitive data from intruders. Also, passwords are encrypted in MySQL.
- Client/ Server Architecture: MySQL follows the working of a client/server architecture. There is a database server (MySQL) and arbitrarily many clients (application programs), which communicate with the server; that is, they can query data, save changes, etc.
- Free to download: MySQL is free to use so that we can download it from MySQL
- Official website without any cost.

➢ FEATURES OF JAVA

- ❖ Java is a widely used programming language expressly designed for use in the distributed environment of the internet. It is the most popular programming language for Android smartphone applications and is also among the most favoured for the development of edge devices and the internet of things.
- ❖ The Java platform is designed with multithreading capabilities built into the language. That means you can build applications with many concurrent threads of activity, resulting in highly interactive and responsive applications
- ❖ Java programming language is easy to learn and also easy to read and write.
- ❖ Java code is compiled into intermediate format (byte code), which can be executed on any systems for which Java virtual machine is ported. That means you can write a Java program once and run it on Windows, Mac, Linux or Solaris without re-compiling.
- ❖ The Integrated Development Environment (IDE) is one of Java's most intriguing features. The Java IDE is a collection of automation tools, editors, and debuggers.
- ❖ With automatic garbage collection and simple memory management model (no pointers like C/C++), plus language features like generics, try-with-resources
- ❖ Since Java SE version 8 (JDK 8), Java is updated with functional programming feature like functional interfaces and Lambda Expressions. This increases the flexibility of Java.
- ❖ Unlike C++ which is semi object-oriented, Java is a fully object-oriented programming language. It has all OOP features such as abstraction, inheritance and polymorphism.

- Java is similar to C/C++ but it removes the drawbacks and complexities of C/C++ like pointers and multiple inheritances. So if you have background in C/C++, you will find Java familiar and easy to learn.

FEATURES OF PHP

- PHP is an open-source, interpreted, and object-oriented scripting language that can be executed at the server-side. PHP is a general-purpose scripting language geared toward web development (an application that executes on the server and generates the dynamic page.).PHP was created by Rasmus Lerdorf in 1994 but appeared in the market in 1995. PHP 7.4.0 is the latest version of PHP, which was released on 28 November.
- PHP is very popular language because of its simplicity and open source.
- Platform Independent: PHP is available for WINDOWS, MAC, LINUX & UNIX operating system. A PHP application developed in one OS can be easily executed in other OS also.
- Open Source: PHP source code and software freely available on the web. We can develop all the versions of PHP according to your requirement without paying any cost. All its components are free to download and use.
- Error Reporting: PHP has predefined error reporting constants to generate an error notice or warning at runtime. E.g., E_ERROR, E_WARNING, E_STRICT, E_PARSE.
- Performance: PHP script is executed much faster than those scripts which are written in other languages such as JSP and ASP. PHP uses its own memory, so the server workload and loading time is

automatically reduced, which results in faster processing speed and better performance.

- Embedded: PHP code can be easily embedded within HTML tags and script.
- Web servers Support: PHP is compatible with almost all local servers used today like Apache, Netscape, Microsoft IIS, etc.
- Database Support: PHP supports all the leading databases such as MySQL, ODBC etc.
- Loosely Typed Language: PHP allows us to use a variable without declaring its datatype. It will be taken automatically at the time of execution based on the type of data it contains on its value.
- Familiarity with syntax: PHP has easily understandable syntax. Programmers are comfortable coding with it.

4. SYSTEM ANALYSES

System analysis is the process of gathering and interpreting facts, diagnosing problems and using the facts to improve the system. Analysis is a detailed study of various operations performed by a system and their relationship within and outside of the system. This involves gathering and using structured tools for analysis.

System analysis is the way of studying a system with an eye on solving its problems using computers. To analyse a system one has to study the working of the system in detail. The system analyst has to understand the functioning and concept of the system in detail, before the appropriate computer based system that will meet all the requirements of the existing system. The various techniques used in the study of the present system are:

- Observation: We had observed several times that so many students are facing difficulties to approach a particular place, to track a particular event etc. So as the initial step of our project we figured various fields were we should implement our project.
- Discussion: As part of our project we conducted a direct discussion with students and teachers about their needs and essential requirements to be implemented in our organization. That was the phase in which we realized the actual need of our android application in each institutions.
- Information collection: We visited various department tutors students, NSS coordinators, HOD's of various departments and collected information about activities and programs which they had conducted and planning to conduct in the institution.

4.1 PRELIMINARY INVESTIGATION

Preliminary investigation basically refers to the collection of information that guides the managements of an organization to evaluate the merits and demerits of the projects request and make an informed judgment about the feasibility of the proposed system. This sort of investigation provide us with a through picture of the kind of software and hardware requirements which are host feasible for the system, plus the environment in which the entire project has to be installed and made optional.

4.2 FEASIBILITYSTUDY

Feasibility study is a procedure that identifies, describes and evaluates a candidate's system and selects the best system for the job. An estimate is made of whether the identified user needs may be satisfied using current software and hardware technologies. The study will decide if the proposed system will be cost effective from business constraints. The key considerations involved in the feasibility analysis are economic, technical, behavioural and operational.

- **Economic Feasibility**

The economic analysis is to determine the benefits and saving that are expected from a candidate system and compare them with costs. The system is economically feasible, has the organization possesses the hardware and software resources required for the functioning of the system. Any additional resources, if required, can also be easily acquired.

- **Technical Feasibility**

It centers on the existing computer system and to what extent it can support the proposed addition. Since the minimum requirement of the system like IIS the server and browser on the client, are met by any average user.

- **Operational Feasibility**

System operation is the longest phase in the development life cycle of a system. So, operational feasibility should be given much importance. The users of the system don't need to go through training on the system. All they are expected to know to operate the system is the basic net surfing knowledge. It has a user-friendly interface.

4.3 PROPOSED SYSTEM

The proposed system is an android application with Android at the front end and PHP-MySQL at the back end. Students cell project is a professional service that uses specialized, project management techniques to oversee the project's time, cost and quality, security is an important aspect in the project and is maintained using a username and password which is unique for each user. The student details and other details that a specific user gives to the application is stored in the database. It allows the data to be stored in a secure and safe way. Data loss can be avoided to a certain extent. During this project, a system for payment is introduced.

In this modern age, student's management has become very essential component in education. With the help of College Student's and Admin android application we can gather all the useful information and help needed to a student. Our System manages the three most important module such as user/student, admin.

We Create login for the user based on their access rights. Users and admin can login by marking a secure username and password by themselves Then they can get into the application and ask for the needful help or provide with the needful information.The system is given a unique access to every user who are having username and password which is necessary access to the system.

The requirement of the user is to:

- Students Cell: Student Registration and Login- Here to access the system, students need to first register themselves into the system. Registered students can access the system by logging into it with a login page where the registered user/student can enter username and password to access. Every users are treated equally with the system.

The requirement of the admin is to:

- Admin Login: The admin can login to the system with a specific username and password . An admin can approve any registration as well as remove them. Not only registration, admin has the power to approve or delete each and every fields.

The main Ideas we would like to introduce and implement in our institution through the cell are:

User Modules of the System

- Fund Raising Module : - Members of the institution may be facing any serious financial issues as well as collecting fund for the college programs by students/NSS, at times of emergency we can seek help from other members by informing them about the seriousness of the situation.
- Requirements Module : - Here, we can ask for the essential maintenance works and new requirements which are essential for the members of the institution
- Notifications Module : - Here, we will be providing all the notification related to educational , arts , sports , NSS programs which are being conducted in the campus or related to student's in the campus . Example : university examination time table , results , blood donation camps etc.

- Academic Calendar Module : -Each event will be marked in the academic calendar . By clicking the specific date and we can know about the event that is upcoming or which has been already finished. By clicking the finished events we can get images/videos related to that events and information about event.
- Chat Box Module : - User can chat with admin through private chat. Privacy is ensured while chatting. Students can get information and needful help from the admin.
- Idea corner : -Many student's will be having their own perspective view towards each situations. So all of them will be having different ideas towards each Issue or new innovative ideas which can be implemented in the institution. So we are providing them a platform to express their idea's which can be implemented to gain more entertainment/information.

Information in the cell can be given not only by the admin but also by the Users, under one condition that is: Admin has to approve the users information/issue only then it will be published in the cell.

Advantages:

➤ Manual processes can be eliminated.

➤ Less time needed to prepare records.

➤ Admin can control the system effectively.

➤ Users can get the details of Requirements, Academic Events and Fund necessity in one platform.

5. SYSTEM DESIGN

5.1 DESIGN OF PROPOSED SYSTEM

System design transforms a logical representation what the system is required to do into the physical specification. The specifications are converted into a physical reality during the development. Design forms a blue print of the system and adds how the components relate to catch other. The design phase proceeds accordingly to an ordinary sequence of steps, beginning with review and assigning of task and ending with package design. Design phase is the lifecycle phase in which the detailed design of the system selected in the study phase is accomplished. A smooth transition from the study phase to design is necessary because the design phase continues the activities in the earlier phase. Simplicity is the most important criteria of design phase. The most creative and challenging phase of the life cycle is system design. The term design describes the final system and the process by which it is developed. The first step in design is to determine how the output is to be produced and in what format. Second the formats of input screen are to be determined. The input data and the master files have to be designed to meet the requirements of the proposed output. In institutions, there are still many time-consuming tasks and problems that can be overthrown with the help of technology. As the students have to spend a lot of time going through all these processes and the physical availability of faculty is always required. In order to conquer this problem, we introduce our Student's Cell. This application consists of two main modules such as USER and ADMIN. The main purpose is to create an android application which will manage the working of these different modules. The completion of the registration process is quite hectic as the student has to fill the form and then physically

submit it to the admin, which consumes a lot of time. This app will make this process work in an easy way.

The students can register through this app, and the admin will just have to acknowledge the form. This process won't require the physical presence of students as well as admin. This process can help manage all the data in a more organized manner. The main lead for using this system is that the faculty/admin need not be physically present at their respective places and students can express their feelings easily also can gather information's about their needs. The system can be accessed irrespective of time or place. The admin just has to approve the application, and student's can login into app.

Even if multiple students apply at the same time there won't be a problem for managing those students. Admin plays a major role in this project. Student's can interact with admin using chat box. All the record stores in database. The proposed software will also reduce the fearsome in students and can help them with various problems that are related to institution.

System design goes through two phases of development

- Logical design
- Physical design

1. LOGICAL DESIGN

The part of the design process that is independent of any specific hardware or software platform is referring to a logical design. During the local design, all functional features of the system chosen for development in analysis phase are described independently of any computer platform.

2. PHYSICAL DESIGN

Physical design is the part of the design in which the logical specification of the system from logical design are transferred into technology-specific details from which all programming and system construction can be accomplished. The system performs information output.

5.2 DATABASE DESIGN

The objective of database design is to provide auxiliary storage and to contribute to be overall efficiency of the computer program component one auxiliary storage medium must provide efficient access to the data. The concept behind a database is an integrated collection of data and provides centralized access to the data from program.

A database is a collection of logically related data stored with minimum redundancy to serve many users quickly and efficiently. Relationships are established between the data items and unnecessary data items are removed. Normalization is done to get an internal consistency of the data items and to have minimum redundancy and maximum stability. While designing database, several objectives must be considered.

- Accuracy and integrity
- Privacy and security
- Performance

5.3 NORMALIZATION

The term normalization refers is a technique of separating redundant fields and breaking up large tables into smaller ones. After the conceptual level, the next level process of database designed to organize the database structure into a good shaped is called normalization. Normalization is adapted to overcome drawbacks like

- Loss of information
- Inconsistency
- Reparation of data

Different normal forms:

- First Normal Form (1 NF)
- Second Normal Form (2 NF)

First Normal Form (1NF)

A relation is said to be in 1NF if the constraints that contain primary key only.

Second Normal Form (2 NF)

The concept of 2NF is that every table has a primary key and other fields depends only on the primary key.

5.4 DATABASE DESCRIPTION

TABLE: USER REGISRATION

DESCRIPTION: To store user information.

SL NO.	FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
1.	id	int	11	Primary Key
2.	first_name	varchar	30	Not null
3.	last_name	varchar	30	Not null
4.	phone	varchar	10	Not null
5.	email	varchar	30	Not null
6.	user_name	varchar	20	Not null
7.	password	varchar	10	Not null
8.	gender	varchar	6	Not null
9.	status	varchar	10	Not null

TABLE: CHATBOT

DESCRIPTION: To set chatbot related queries and replies.

SL NO.	FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
1.	id	int	11	Primary key
2.	queries	Varchar	500	Not null
3.	replies	varchar	500	Not null

TABLE: EVENTS

DESCRIPTION: To store calendar events .

SL NO.	FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
1.	id	int	11	Primary Key
2.	title	Varchar	255	Not null
3.	start_date	date	50	Not null
4.	end_date	date	50	Not null
5.	created	date		Not null
6.	status	tinyint	1	Not null

TABLE: IDEA

DESCRIPTION: To store the creative ideas of users.

SL NO.	FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
1.	id	Int	100	Primary Key
2.	user_id	int	100	Not null
3.	name	varchar	100	Not null
4.	department	varchar	100	Not null

5.	idea	varchar	700	Not null
6.	status	varchar	10	Not null

TABLE: NOTIFICATION

DESCRIPTION: To store the newly published notifications under the university and institutions.

SL NO.	FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
1.	id	Int	10	Primary Key
2.	admin_id	Varchar	100	Not null
3.	eventname	Varchar	100	Not null
4.	date	Varchar	100	Not null
5.	place	Varchar	10	Not null
6.	time	Int	100	Not null
7.	description	Varchar	1000	Not null
8.	image	Int	255	Not null
9.	status	Varchar	10	Not null

TABLE: FUND

DESCRIPTION: TO store fund raise related informations.

SL NO.	FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
1.	id	int	11	Primary Key
2.	date	varchar	10	Not null
3.	fundraise	varchar	200	Not null
4.	name	varchar	20	Not null
5.	age	int	200	Not null
6.	acc_no	varchar	20	Not null
7.	ifsc_code	varchar	20	Not null
8.	branchname	varchar	30	Not null
9.	deadline	varchar	10	Not null
10.	image	varchar	1000	Not null
11.	status	varchar	10	Not null

TABLE: REQUIREMENTS

DESCRIPTION: To store the requirements raised by te users

SL NO.	FIELD NAME	DATATYPE	SIZE	CONSTRAINTS
1.	id	Int	11	Primary Key
2.	user_id	Int	10	Not null
3.	name	Varchar	20	Not null
4.	department	Varchar	30	Not null
5.	description	Varchar	100	Not null
6.	image	Varchar	200	Not null
7.	status	Varchar	10	Not null

5.5 MODULE DESIGN

- Admin Module
- User Module

ADMIN

In this module admin will approve the valid queries and information related to the users and events or else will directly delete the unwanted details from being uploaded on the application. He or she can control the overall working of the application by using this module. The admin can view all the processes which are being listed on the application and can

add events into the application. Admin controls the overall working of the application.

USER

In this module the user can register himself by filling the registration form provided in the application. After he or she can login with the help of the username and password which has been already registered. The user can ask for the essential requirements and maintenance works, the user can ask for emergency fund raising issues, can get helped with the known queries from the chatbot, view and add events in the calender and can also come forward with the creative ideas which he or she would like to implement in the organization.

User Modules of the System

- Fund Raising Module : - Members of the institution may be facing any serious financial issues as well as collecting fund for the college programs by students/NSS, at times of emergency we can seek help from other members by informing them about the seriousness of the situation.
- Requirements Module : - Here, we can ask for the essential maintenance works and new requirements which are essential for the members of the institution
- Notifications Module : - Here, we will be providing all the notification related to educational , arts , sports , NSS programs which are being conducted in the campus or related to student's in the campus . Example : university examination time table , results , blood donation camps etc.
- Academic Calendar Module : -Each event will be marked in the academic calendar . By clicking the specific date and we can know about the event that is upcoming or which has been already finished.

By clicking the finished events we can get images/videos related to that events and information about event.

- Chat Box Module : - User can chat with admin through private chat. Privacy is ensured while chatting. Students can get information and needful help from the admin.
- Idea corner : -Many student's will be having their own perspective view towards each situations. So all of them will be having different ideas towards each Issue or new innovative ideas which can be implemented in the institution. So we are providing them a platform to express their idea's which can be implemented to gain more entertainment/information.

5.6 DEVELOPMENT OF THE SYSTEM

The development of this system contains the following activities, which try to automate the entire process keeping in the view of database integration approach. User friendliness is provided in the application with various controls provided by system rich user interface. The system makes the overall project management much easier and flexible. It is much convenient too. The user information can be stored in centralized database which can be maintained by the system.

6. SYSTEM TESTING

Testing is the major quality measure employed during software development. After the coding phase, computer programs are available that can be executed for testing purposes. Testing not only has to recover errors introduced during coding, but also locate errors committed during the previous phases. Thus, the aim of testing is to verify the requirement design or coding error in the program. System

testing is an expensive but critical process that can take as much as fifty percent of the budget for program development. Consequential different level of testing is employed in fact a successful test is one that finds an error. The system performance criteria deal with turnaround time, backup, file protection and human factor. A test for the user acceptance should be carried out. The package developed was taken through different level of testing and required modification was made.

Type of Testing

System testing is the state of implementation which is aimed at ensuring that the system works accurately and efficiently before live operation commences. Testing is vital to the success of the system. The candidate system is subjected to a variety of test. A series of tests are performed for the proposed system before the system is ready for user

acceptance testing.

- Unit testing
- Integration testing
- Output testing
- User acceptance testing

Unit testing:

Unit testing focuses on the verification effort on the smallest unit of software design or the software component module. Using a component level design as a guide important control paths are tested to find out the errors within the boundary of the module.

Integration testing:

Integration testing is a systematic technique for constructing the program structure while at the same time conducting test to uncover errors associated with interfacing.

Output testing:

Here the output is tested to view whether that screen is what which is desired. It is also checked whether it is to the satisfaction of the user.

User acceptance testing:

User acceptance testing is done in presence of user. The user will have some constraints and if some constraints are absent it has to be added and then the user is satisfied. When this system was executed, we got a successful output with high accuracy as we intended.

7.SYSTEM IMPEMETATION

Implementation is most crucial stages in achieving system and giving the user confidence that new system workable and achievable each program is tested individually at the time for development using test data and was verified that these programs link together in the same way specified in the specification. The computer system and its environment are testing at the specification of the user.

IMPLEMENTATION OF SYSTEM INCLUDE:

- Tested the developed package with sample data.
- Based on the feedback from the user during the test run changes are made to the system.
- Ensuring that the user has understood and accepted the changes.
- Getting complete feedback during test run and ensuring that everything is perfect for the final changeover.

An implementation management team, which includes the head of each implementation team, a primary user of the system oversees the implementation preparation activities. Implementation is the stage of the project when the theoretical design is turned into a working system. The implementation stage is system project in its own right. It includes careful planning, investigation of current system and its constraint on implementation, design of method to achieve the changeover. Our application can be implemented in various educational institutions which has the assistance of an admin to organize the overall functionalities and performance of the system. The institutions can better function efficiently with the help of our application. Moreover the communication among students and organization can be made simpler.

8. CONCLUSION AND FUTURE ENHANCEMENT

As part of project a detailed study has been made about the designing and the project "STUDENTS CELL" has been developed as per the requirement specified by our team. The study has given insight to the current trends and methodologies, We have tried our level best to achieve all the goals and satisfaction. The project has been developed using php for backend and java for frontend. The complete system is thoroughly tested with the availability of our resources and throughout reports which are prepared manually. The project has to meet all the requirements that were collected during analysis and design phase.

Design procedure and output reports are presented in this report. However, our software has great advantage such as promoting the creative ideas of the students, helping them in their most necessary and helpless situations, a calender to remember their institutional events.

The project is developed in such a way that it is able to undergo future enhancement in a reliable, secured manner. The successful completion of this has expanded our boundaries of imaginations, invoked confidence, raised our creativity and has provided with knowledge and experience. We are planning to make our software much efficient in order to build better satisfaction among the users and institution. We can make updation in our app and add features like adding the teachers module, which helps teachers too become a part of our cell, academic notes can be provided in this application itself so backup of the notes will be always available in the application itself, a pshycological assistance for students who request for counselling inorder to understand the problems of the students and provide them with the necessary assistance.

9. BIBILIOGRAPHY

- https://www.irjet.net/archives/V5/i4/IRJET-V5I4137.pdf
- https://www.slideshare.net/irjetjournal/irjet-an-online-grievance-redressal-system
- (800) Making Weather App in Android Studio | Android Projects | GeeksforGeeks - YouTube
- https://youtu.be/PFvnSZHO3cI
- https://www.tutorialspoint.com/android/android_php_mysql.htm
- Headfirst Android Development
  Dawn Griffiths
  1st edition
  O'Reilly
- Android Programming for Beginners
  John Horton
  2nd edition
  Packt Publishing

10. APPENDIX

A. DATAFLOW DIAGRAM

A data flow diagram is used to define the flow of the data and the resource such as information. Data flow diagram are a way of expressing system requirements in a graphical manner. It has the purpose of classifying system requirements and identifying major transformation that will become program in the system design.so it is starting point of design phase that functional decomposes the requirement specification down in to the lowest level of details. The bubbles represent data transformation and the lines represent information flow in the system. Data flow diagrams are useful in understanding a system and can be effectively used for partitioning. The system may be an organization, a manual procedure, software system, mechanical system or any combination of these.

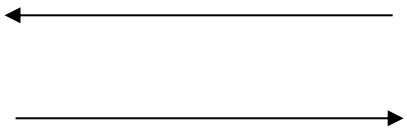
- **Rules for DFD:**

- Process should be named and numbered for easy reference. Each name should be representative of the process.
- The direction of flow is from top to bottom and left to right. That is information flow should be from source to destination.
- Numbering is given when a process is exploded into lower-level details.
- The names of data stores, sources and destination are written in capital letters.
- Process and data flow names have the first letter of each word capitalized. The data flow diagram is particularly design to aid communication. If it contains dozens of process data store it gets to unwieldy.

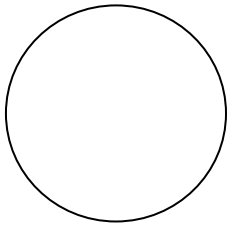
- The goal of DFD is to have a commonly understood model of a system. This diagram the basic of structure diagrams, and data dictionaries.

- **DFD Symbols:**

Data flow diagram is composed of the four bases shown below:-



Data Flow Movement of data between external entities, processes and data stores is represented with an arrow symbol, which indicates the direction of flow. This data could be electronic, written or verbal. Input and output data flows are labelled based on the type of data or its associated process or data store, and this name is written alongside the arrow.



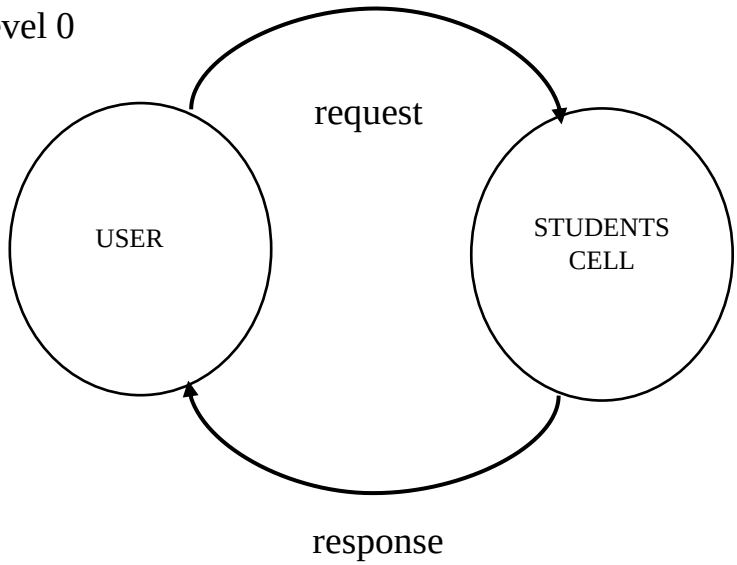
Process - An activity that changes or transforms data flows. Since they transform incoming data to outgoing data, all processes must have inputs and outputs on a DFD. This symbol is given a simple name based on its function, such as "Ship Order," rather than being labelled "process" on a diagram. In Gane-Sarson notation, a rectangular box is used and may be labelled with a reference number, location of where in the system the process occurs and a short title that describes its function. Processes are typically oriented from top to bottom and left to right on a data flow diagram.

External Entity - Also known as actors, sources or sinks, and terminators, external entities produce and consume data that flows between the entity and the system being diagrammed. These data flows are the inputs and outputs of the DFD. Since they are external to the system being analysed, these entities are typically placed at the boundaries of the diagram. They can represent another system or indicate a subsystem.

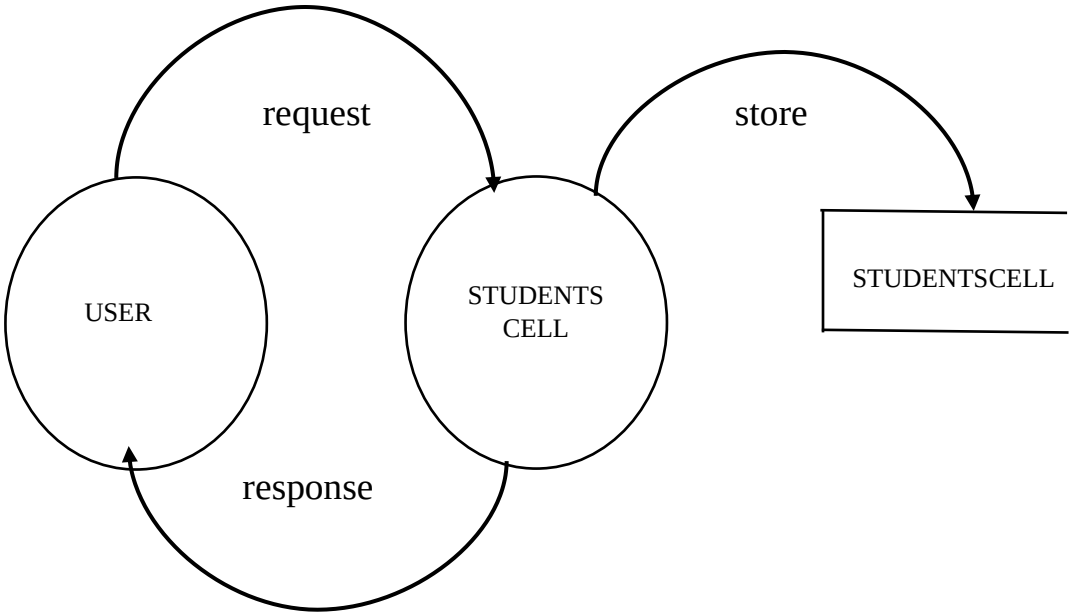
Data Store - A data store does not generate any operations but simply holds data for later access. Data stores could consist of files held long term or a batch of documents stored briefly while they wait to be processed. Input flows to a data store include information or operations that change the stored data. Output flows would be data retrieved from the store

a) DATAFLOW DIAGRAMS

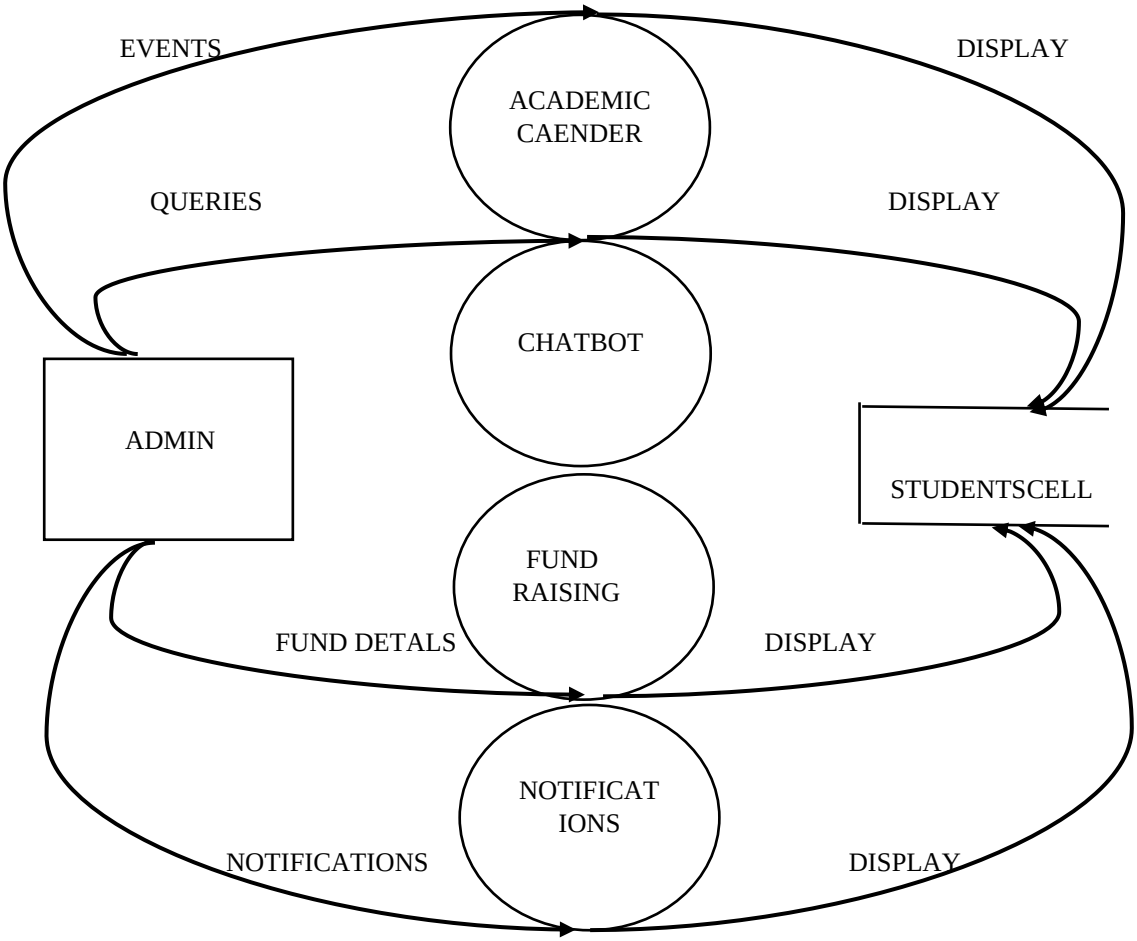
Level 0



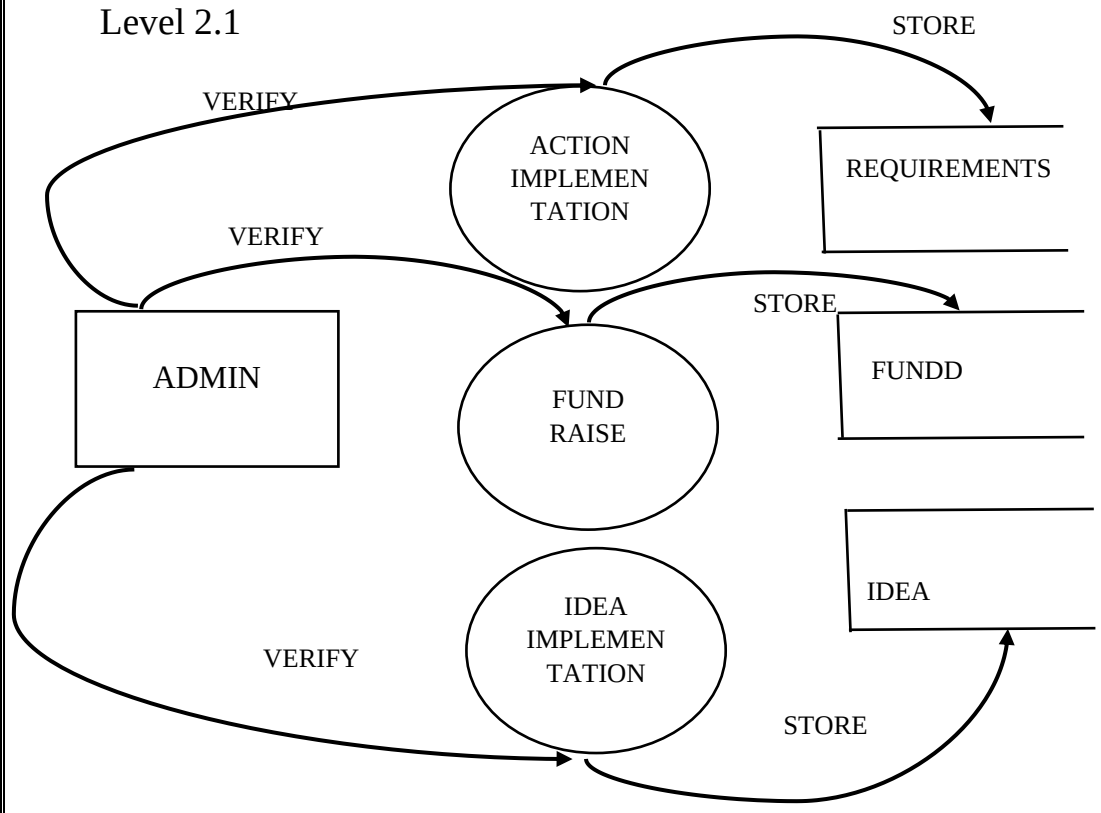
Level 1

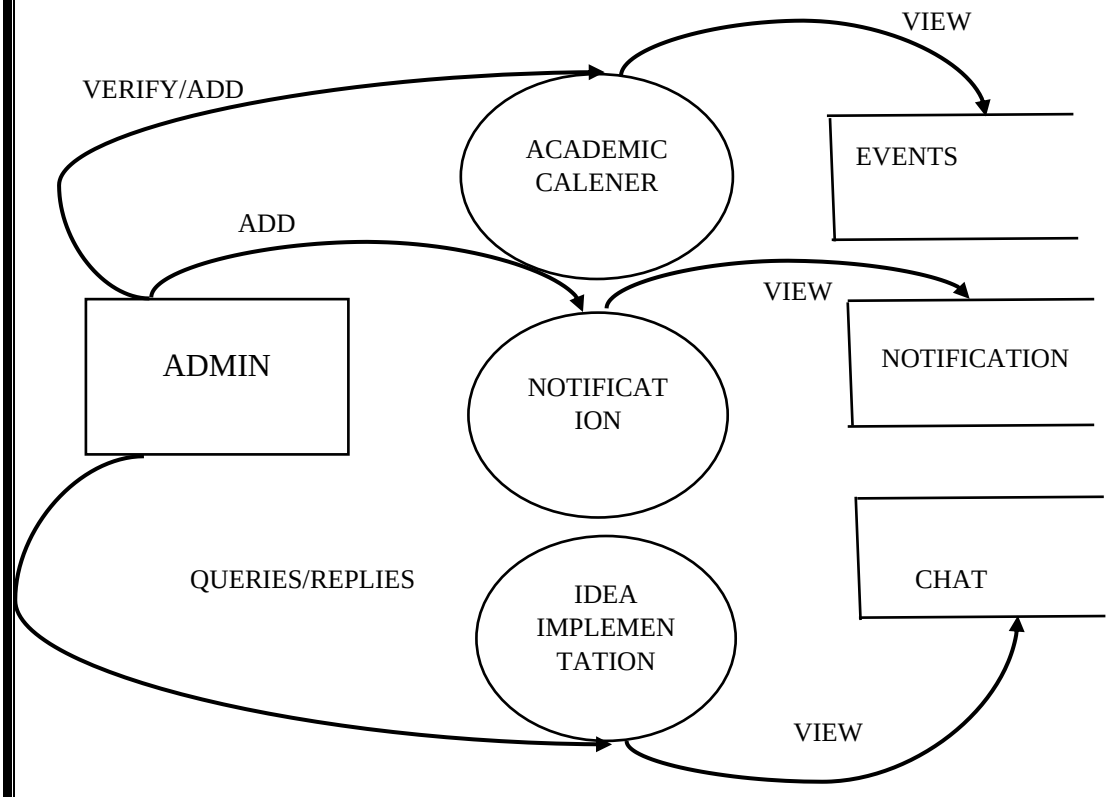


Level 2.0

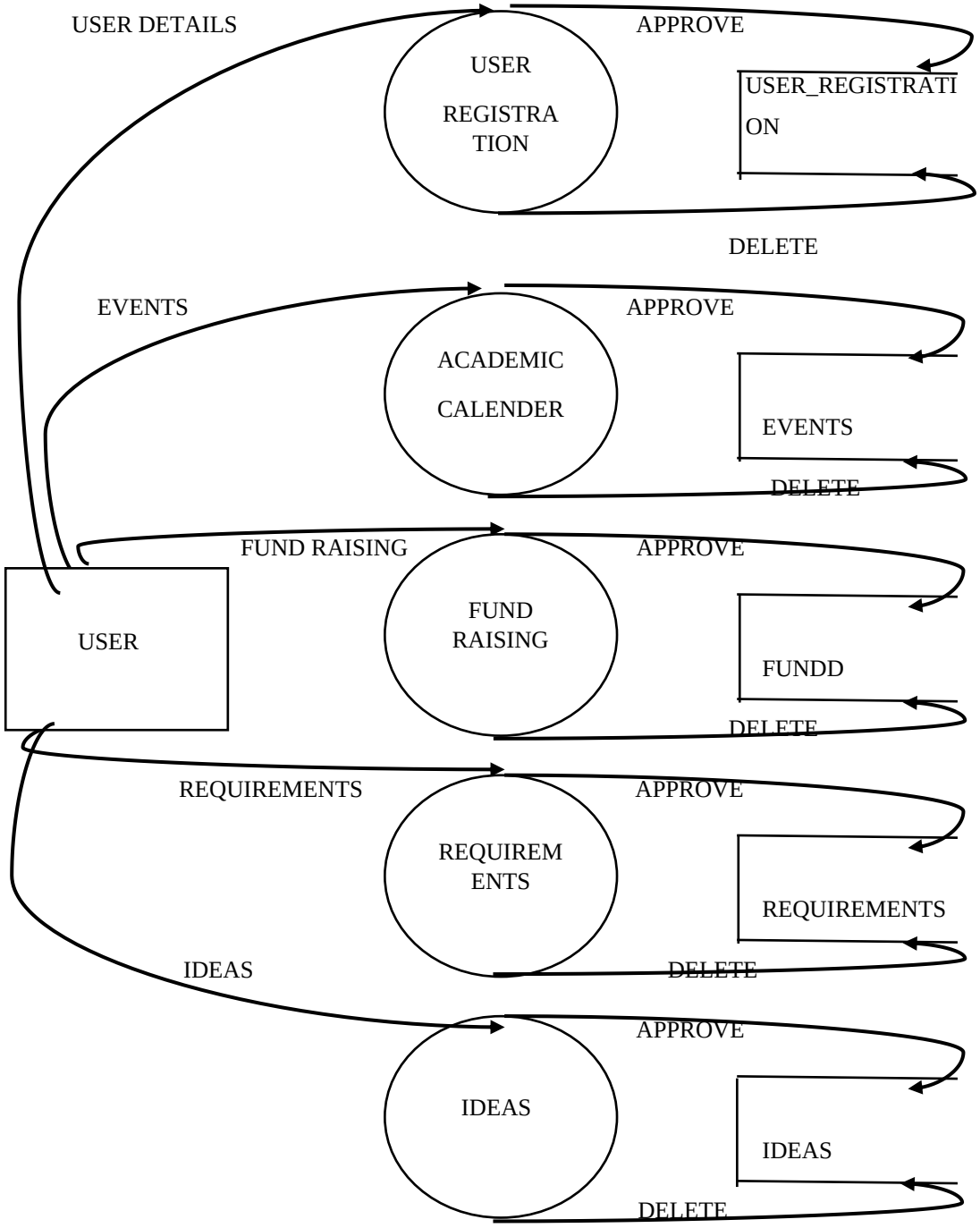


Level 2.1

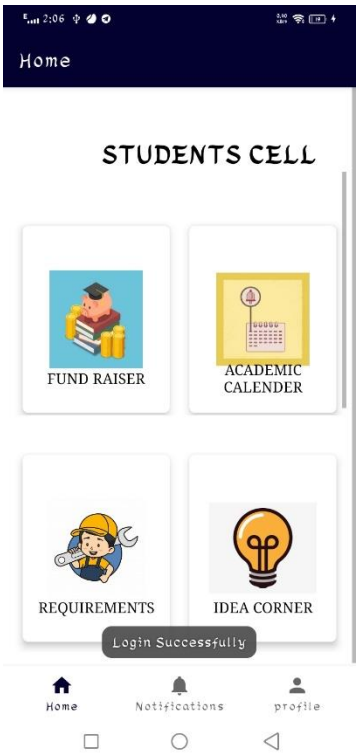

STORE
VERIFY
ACTION IMPLEMEN TATION
REQUIREMENTS
VERIFY
STORE
ADMIN
FUND RAISE
FUNDD
IDEA
IDEA IMPLEMEN TATION
VERIFY
STORE


VIEW
VERIFY/ADD
ACADEMIC CALENER
EVENTS
ADD
VIEW
ADMIN
NOTIFICAT ION
NOTIFICATION
CHAT
QUERIES/REPLIES
IDEA IMPLEMEN TATION
VIEW

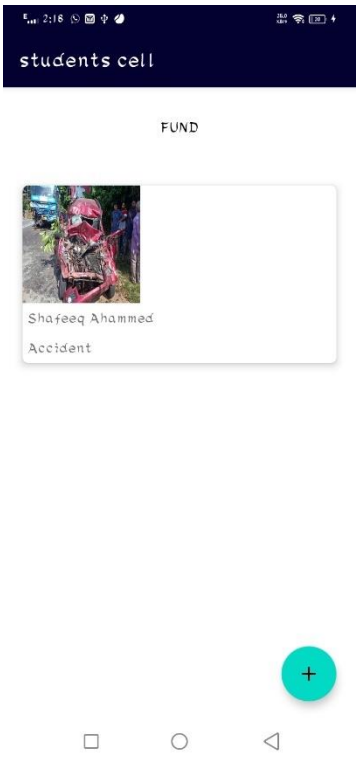
LEVEL 2.1

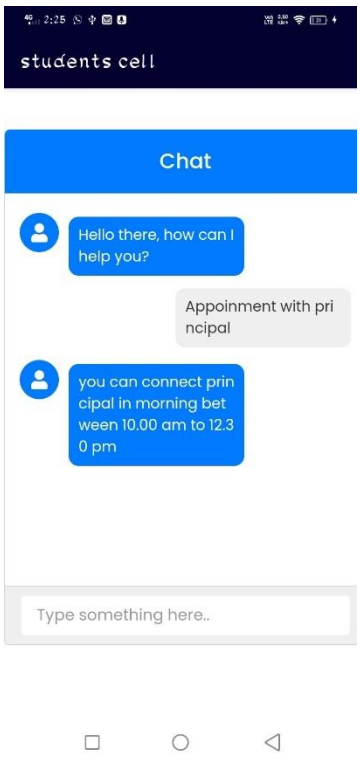

USER DETAILS
APPROVE
USER
REGISTRA
TION
USER_REGISTRATI
ON
DELETE
EVENTS
APPROVE
ACADEMIC
CALENDER
EVENTS
DELETE
FUND RAISING
APPROVE
USER
FUND
RAISING
FUNDD
DELETE
REQUIREMENTS
APPROVE
REQUIREM
ENTS
REQUIREMENTS
IDEAS
DELETE
APPROVE
IDEAS
IDEAS
DELETE

b) SCREENSHOTS

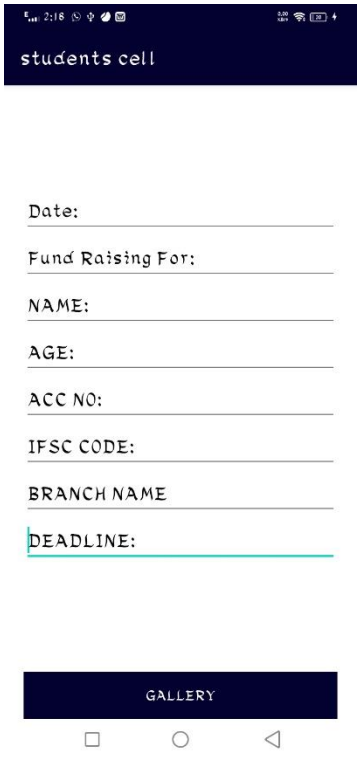


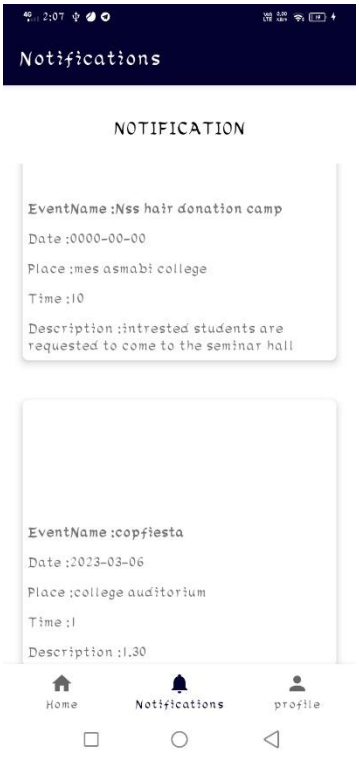
students cell
REQUIREMENTS
sabhari
Clean watertank
Lakshmi
lift is not working


students cell
FUND
Shafeeq Ahammed
Accident


students cell
Chat
Hello there, how can I help you?
Appoinment with principal
you can connect principal in morning between 10.00 am to 12.30 pm
Type something here..

students cell
Home
Add Event
Event CalendarL
<< Prev
Today
Next >>
Year
Month
Week
Day
February 2018
Academic Calendar
Sun... Mon... Tue... Wed... Thu... Frid... Sat...
All Events List
College
Bootstrap events 2018


students cell
Date:
Fund Raising For:
NAME:
AGE:
ACC NO:
IFSC CODE:
BRANCH NAME
DEADLINE:
GALLERY


Notifications
NOTIFICATION
EventName :Nss hair donation camp
Date :0000-00-00
Place :mes asmabi college
Time :10
Description :intrested students are requested to come to the seminar hall
EventName :copfiesta
Date :2023-03-06
Place :college auditorium
Time :1
Description :1.30
Home
Notifications
profile

Profile
PROFILE
sabhari
Sabari
sabhari@gmail.com
9956342786
SAVE
LOGOUT
Home
Notifications
profile

students cell
IDEA CORNER
isjdj
bahssks
jsjdjd
Appu
Electronics
Never
krishna
Bsc maths
college radio
athira
Bsc maths
college nursery
jasmin

c) CODES

REGISTRATION ACTIVITY

```
package com.example.studentscell;

import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.provider.VoicemailContract;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.Toast;

import com.android.volley.AuthFailureError;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONException;
import org.json.JSONObject;
```

```
import java.util.HashMap;
import java.util.Map;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

import javax.net.ssl.SSLEngineResult;

public class Reg extends AppCompatActivity {
   EditText
edt_fname,edt_lname,edt_phone,edt_uname,edt_password,edt_mail;
   Button sub;
   RadioGroup radio_gen;
   String url=config.baseUrl+"registration.php";
   String
fname,lname,phone,uname,password,mail,status,message,gender;
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_reg);
      edt_fname=findViewById(R.id.edt_fname);
      edt_lname=findViewById(R.id.edt_lname);
      edt_mail=findViewById(R.id.edt_mail);
      edt_phone=findViewById(R.id.edt_phone);
      edt_uname=findViewById(R.id.edt_uname);
      edt_password=findViewById(R.id.edt_password);
      sub=findViewById(R.id.sub1);
      radio_gen=findViewById(R.id.radio_gen);
      sub.setOnClickListener(new View.OnClickListener() {
```

```
        @Override
        public void onClick(View view) {
            userReg();
        }
    });

}

private void userReg() {
    fname = edt_fname.getText().toString();
    lname = edt_lname.getText().toString();
    phone = edt_phone.getText().toString();
    uname = edt_uname.getText().toString();
    password = edt_password.getText().toString();
    mail = edt_mail.getText().toString();

    int id=radio_gen.getCheckedRadioButtonId();
    RadioButton radioButton=radio_gen.findViewById(id);
    gender=radioButton.getText().toString();


    if (TextUtils.isEmpty(fname)) {
        edt_fname.setError("Required");
        edt_fname.requestFocus();
        return;
    } else if (TextUtils.isEmpty(lname)) {
        edt_lname.setError("Required");
        edt_lname.requestFocus();
        return;
```

```
    } else if (TextUtils.isEmpty(phone)) {
        edt_phone.setError("Required");
        edt_phone.requestFocus();
        return;
    } else if (TextUtils.isEmpty(uname)) {
        edt_uname.setError("Required");
        edt_uname.requestFocus();
        return;
    } else if (TextUtils.isEmpty(mail)) {
        edt_mail.setError("Required");
        edt_mail.requestFocus();
        return;
    } else if (TextUtils.isEmpty(password)) {
        edt_password.setError("Required");
        edt_password.requestFocus();
        return;
    }

    StringRequest stringRequest= new
StringRequest(Request.Method.POST, url,
            new Response.Listener<String>()
            {
                @Override
                public void onResponse(String response)
                {
                    Toast.makeText(Reg.this, response,
Toast.LENGTH_SHORT).show();
                    try {
                        JSONObject c = new JSONObject(response);
```

```
                status =c.getString("status");
                message=c.getString("message");



            }
            catch (JSONException e)
            {
                e.printStackTrace();
            }

            if ("0".equals(status))
            {
                Toast.makeText(Reg.this, "Registration failed",
Toast.LENGTH_SHORT).show();
            }

            else
            {
                Toast.makeText(Reg.this, "Registered Successfully",
Toast.LENGTH_SHORT).show();
                Intent i= new Intent(Reg.this,MainActivity.class);
                startActivity(i);
                finish();
            }

        }
    }, new Response.ErrorListener()
{
    @Override
```

```
        public void onErrorResponse(VolleyError error)
        {
            Toast.makeText(Reg.this,String.valueOf(error),
Toast.LENGTH_SHORT).show();
        }
      }
      )
      {
        @Nullable
        @Override
        protected Map<String, String> getParams() throws
AuthFailureError {
            Map<String,String> params=new HashMap<>();
            params.put("first_name",fname);
            params.put("last_name",lname);
            params.put("phone",phone) ;
            params.put("email",mail) ;
            params.put("user_name",uname);
            params.put("password",password);
            params.put("gender",gender);

            return params;
        }
      };
      RequestQueue requestQueue= Volley.newRequestQueue(this);
      requestQueue.add(stringRequest);

  }
```

```
    public static boolean isPhoneValid(String s) {
        Pattern p = Pattern.compile("(0/91)?[6-9][0-9]{9}");
        Matcher m = p.matcher(s);
        return (m.find() && m.group().equals(s));
    }

    public static boolean isEmailValid(String email) {
        String emailRegex = "^[a-zA-Z0-9_+&*-]+(?:\\."+
                "[a-zA-Z0-9_+&*-]+)*@" +
                "(?:[a-zA-Z0-9-]+\\.)+[a-z" +
                "A-Z]{2,7}$";

        Pattern pat = Pattern.compile(emailRegex);
        return pat.matcher(email).matches();
    }


}
```

MAIN ACTIVITY

```
 package com.example.studentscell;


import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.util.Log;
import android.view.View;
```

```
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;

import com.android.volley.AuthFailureError;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONException;
import org.json.JSONObject;

import java.util.HashMap;
import java.util.Map;

public class MainActivity extends AppCompatActivity {
    EditText edt_username,edt_pname;
    Button button_submit;
    String username,pname;
    String url=config.baseUrl+"login.php";
    String status,message;
    String id,fname,lname,phone,email,uname,password,gender;
```

```
TextView txt;



@Override
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_main);
   edt_username=findViewById(R.id.edt_username);
   edt_pname=findViewById(R.id.edt_pname);
   button_submit=findViewById(R.id.button_submit);
   txt=findViewById(R.id.txt);
   txt.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View view) {
         Intent intent=new Intent(MainActivity.this,Reg.class);
         startActivity(intent);
      }
   });

   button_submit.setOnClickListener(new View.OnClickListener()

   {
      @Override
      public void onClick(View view) {

         UserReg();
      }
   });
}
```

```
    private void UserReg() {
        username=edt_username.getText().toString();
        pname=edt_pname.getText().toString();
        if(TextUtils.isEmpty(username))
        {
            edt_username.setError("Required");
            edt_username.requestFocus();
            return;
        }
        else if(TextUtils.isEmpty(pname))
        {
            edt_pname.setError("Required");
            edt_pname.requestFocus();
            return;
        }
        StringRequest stringRequest= new
StringRequest(Request.Method.POST, url,
                new Response.Listener<String>()
                {
                    @Override
                    public void onResponse(String response)
                    {
                        Log.d("response>>",response);
//                              Toast.makeText(MainActivity.this, response,
Toast.LENGTH_SHORT).show();
                        try {
                            JSONObject c = new JSONObject(response);
                            status=c.getString("status");
```

```
                message=c.getString("message");
                id=c.getString("id");
                fname=c.getString("first_name");
                lname=c.getString("last_name");
                phone=c.getString("phone");
                email=c.getString("email");
                uname=c.getString("username");
                password=c.getString("password");
                gender=c.getString("gender");


            }
            catch (JSONException e)
            {
                e.printStackTrace();
            }

            if ("0".equals(status))
            {
                Toast.makeText(MainActivity.this, "Login failed",
Toast.LENGTH_SHORT).show();
            }

            else
            {
                Toast.makeText(MainActivity.this, "Login
Successfully", Toast.LENGTH_SHORT).show();
```

```
                    new
SessionManager(MainActivity.this).createLoginSession(id,fname,lname,
email,phone,uname,password,gender);
                    Intent          i=          new
Intent(MainActivity.this,User_homepage.class);
                    startActivity(i);
                    finish();
                }

            }
        }, new Response.ErrorListener()
    {
        @Override
        public void onErrorResponse(VolleyError error)
        {
            Toast.makeText(MainActivity.this,String.valueOf(error),
Toast.LENGTH_SHORT).show();
        }
    }
    )
    {
        @Nullable
        @Override
        protected Map<String, String> getParams() throws
AuthFailureError
        {
            Map<String,String> params=new HashMap<>();
            params.put("username",username);
            params.put("password",pname);
```

```
            return params;
        }
    };
    RequestQueue requestQueue= Volley.newRequestQueue(this);
    requestQueue.add(stringRequest);

  }



}
```

HOMEFRAGMENT

```
package com.example.studentscell.ui.home;

import android.content.Intent;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;

import androidx.annotation.NonNull;
import androidx.cardview.widget.CardView;
import androidx.fragment.app.Fragment;
import androidx.lifecycle.ViewModelProvider;

import com.example.studentscell.AcademicCalenderActivity;
import com.example.studentscell.ChatbotActivity;
```

```
import com.example.studentscell.Fund;
import com.example.studentscell.Idea;
import com.example.studentscell.R;
import com.example.studentscell.Req_list;
import com.example.studentscell.acadmc;
import com.example.studentscell.databinding.FragmentHomeBinding;
import com.example.studentscell.fund_list;
import com.example.studentscell.idea_list;
import com.example.studentscell.reqmt;



public class HomeFragment extends Fragment {




    private FragmentHomeBinding binding;

    Button b1;
    CardView fundraiser;
    CardView academiccalender;
    CardView requirements;
    CardView ideacorner;
    String fund,acadmc,rqrmnt,idea;

    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle savedInstanceState) {
        HomeViewModel homeViewModel =
                new ViewModelProvider(this).get(HomeViewModel.class);
```

```
        binding = FragmentHomeBinding.inflate(inflater, container, false);
        View root = binding.getRoot();
        b1=root.findViewById(R.id.chat);
        fundraiser=root.findViewById(R.id.fund);
        academiccalender=root.findViewById(R.id.clndr);
        requirements=root.findViewById(R.id.reqrmnt);
        ideacorner=root.findViewById(R.id.idea);

        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent=new Intent(getActivity(), ChatbotActivity.class);
                startActivity(intent);
            }
        });
        fundraiser.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent = new Intent(getActivity(), fund_list.class);
                startActivity(intent);

            }

        });



        academiccalender.setOnClickListener(new  View.OnClickListener()
{
```

```
        @Override
        public void onClick(View view) {
            Intent intent=new Intent(getActivity(),
AcademicCalenderActivity.class);
            startActivity(intent);


        }
    });

    requirements.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Intent intent=new Intent(getActivity(), Req_list.class);
            startActivity(intent);
        }
    });

    ideacorner.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Intent intent=new Intent(getActivity(), idea_list.class);
            startActivity(intent);
        }
    });

    {


    }
```

```
        return root;
    }

    @Override
    public void onDestroyView() {
        super.onDestroyView();
        binding = null;
    }
}
```

FUND ACTIVITY

```
package com.example.studentscell;

import androidx.appcompat.app.AppCompatActivity;

import android.annotation.SuppressLint;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.database.Cursor;
import android.net.Uri;
```

```
import android.os.Bundle;
import android.provider.OpenableColumns;
import android.text.TextUtils;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.Toast;

import com.android.volley.AuthFailureError;
import com.android.volley.DefaultRetryPolicy;
import com.android.volley.NetworkResponse;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.Volley;

import org.json.JSONException;
import org.json.JSONObject;

import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStream;
import java.util.HashMap;
import java.util.Map;
```

```
public class Fund extends AppCompatActivity
{
   EditText edt_date, edt_dte, edt_name, edt_age, edt_accno, edt_ifc, edt_brnch, edt_ddlne;
   Button gallery;
   String date, dte, name, age, accno, ifc, brnch, ddlne,status,message;
   ImageView gly;
   private RequestQueue rQueue;
   private static ProgressDialog mProgressDialog;
   private String url = config.baseUrl +"fund.php";


   @Override
   protected void onCreate(Bundle savedInstanceState)
   {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_fund);
      edt_date = findViewById(R.id.edt_date);
      edt_dte = findViewById(R.id.edt_dte);
      edt_name = findViewById(R.id.edt_name);
      edt_age = findViewById(R.id.edt_age);
      edt_accno = findViewById(R.id.edt_accno);
      edt_ifc = findViewById(R.id.edt_ifc);
      edt_brnch = findViewById(R.id.edt_brnch);
      edt_ddlne = findViewById(R.id.edt_ddlne);
      gallery = findViewById(R.id.gallery);
      gly=findViewById(R.id.gly);
      gallery.setOnClickListener(new View.OnClickListener() {
```

```
        @Override
        public void onClick(View view) {
            fgallery();

        }
    });

}

private void fgallery() {
    date = edt_date.getText().toString();
    dte = edt_dte.getText().toString();
    name = edt_name.getText().toString();
    age = edt_age.getText().toString();
    accno = edt_accno.getText().toString();
    ifc = edt_ifc.getText().toString();
    brnch = edt_brnch.getText().toString();
    ddlne = edt_ddlne.getText().toString();
    if (TextUtils.isEmpty(date))
    {
        edt_date.setError("Required");
        edt_date.requestFocus();
        return;
    }
    else if (TextUtils.isEmpty(dte))
    {
        edt_dte.setError("Required");
        edt_dte.requestFocus();
        return;
```

```
}
else if (TextUtils.isEmpty(name)) {
    edt_name.setError("Required");
    edt_name.requestFocus();
    return;
}
else if (TextUtils.isEmpty(age)) {
    edt_age.setError("Required");
    edt_age.requestFocus();
    return;
}
else if (TextUtils.isEmpty(accno)) {
    edt_accno.setError("Required");
    edt_accno.requestFocus();
    return;
}
else if (TextUtils.isEmpty(ifc)) {
    edt_ifc.setError("Required");
    edt_ifc.requestFocus();
    return;
}
else if (TextUtils.isEmpty(brnch)) {
    edt_brnch.setError("Required");
    edt_brnch.requestFocus();
    return;
}
else if (TextUtils.isEmpty(ddlne)) {
    edt_ddlne.setError("Required");
    edt_ddlne.requestFocus();
```

```
            return;
        }
        Intent intent = new Intent();
        intent.setAction( Intent.ACTION_GET_CONTENT );
        intent.setType( "image/*" );
        //intent.setType("application/pdf");
        startActivityForResult( intent, 1 );
    }
    @SuppressLint("Range")
    @Override
    public void onActivityResult(int requestCode, int resultCode, Intent
data) {
        if (resultCode == RESULT_OK) {
            // Get the Uri of the selected file
            Uri uri = data.getData();
            String uriString = uri.toString();
            File myFile = new File(uriString);
            String path = myFile.getAbsolutePath();
            String displayName = null;

            if (uriString.startsWith("content://")) {
                Cursor cursor = null;
                try {
                    cursor =getContentResolver().query(uri, null, null, null,
null);
                    if (cursor != null && cursor.moveToFirst()) {
                        displayName = cursor.getString(cursor.getColumnIndex(
OpenableColumns.DISPLAY_NAME));
                        Log.d("name ",displayName);
```

```
                uploadPDF(displayName,uri);
            }
        } finally {
            cursor.close();
        }
    } else if (uriString.startsWith("file://")) {
        displayName = myFile.getName();
        Log.d("nameeeee>>>>  ",displayName);
    }
}

super.onActivityResult(requestCode, resultCode, data);

}

private void uploadPDF(final String pdfname, Uri pdffile) {
    InputStream iStream = null;
    try {

        iStream = getContentResolver().openInputStream(pdffile);
        final byte[] inputData = getBytes(iStream);

        showSimpleProgressDialog(getApplicationContext(), null,
"Uploading image", false);

        VolleyMultipartRequest volleyMultipartRequest = new
VolleyMultipartRequest( Request.Method.POST,url,
                new Response.Listener<NetworkResponse>() {
```

```
                @Override
                public void onResponse(NetworkResponse response) {
                    removeSimpleProgressDialog();
                    Log.d("res",new String(response.data));
                    rQueue.getCache().clear();
                    try {
                        JSONObject jsonObject = new JSONObject(new
String(response.data));

                        jsonObject.toString().replace("\\\\","");

                        status = jsonObject.getString("status");
                        message = jsonObject.getString("message");

                        if (status.equals("1")) {
                            Toast.makeText(Fund.this,          message,
Toast.LENGTH_SHORT).show();
                            finish();
                        }
                        else {
                            Toast.makeText(Fund.this,          message,
Toast.LENGTH_SHORT).show();
                        }


                    } catch (JSONException e) {
                        e.printStackTrace();
                    }
                }
```

```
            },
            new Response.ErrorListener() {
                @Override
                public void onErrorResponse(VolleyError error) {
                    removeSimpleProgressDialog();
                    Toast.makeText(getApplicationContext(),
error.toString(), Toast.LENGTH_LONG).show();
                }
            }) {

        @Override
        protected Map<String, String> getParams() throws
AuthFailureError {
            Map<String, String> params = new HashMap<>();
            //add string parameters
            params.put("name", name);
            params.put("date", date);
            params.put("dte", dte);
            params.put("age",age);
            params.put("accno",accno);
            params.put("ifc",ifc);
            params.put("brnch",brnch);
            params.put("ddlne",ddlne);



            return params;
        }
```

```
            @Override
            protected Map<String, DataPart> getByteData() {
                Map<String, DataPart> params = new HashMap<>();
                params.put("filename", new DataPart(pdfname ,inputData));
                return params;
            }
        };


        volleyMultipartRequest.setRetryPolicy(new DefaultRetryPolicy(
                0,
                DefaultRetryPolicy.DEFAULT_MAX_RETRIES,
                DefaultRetryPolicy.DEFAULT_BACKOFF_MULT));

        rQueue = Volley.newRequestQueue(Fund.this);
        rQueue.add(volleyMultipartRequest);



    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }


}
```

```
    public byte[] getBytes(InputStream inputStream) throws IOException
{
        ByteArrayOutputStream byteBuffer = new
ByteArrayOutputStream();
        int bufferSize = 1024;
        byte[] buffer = new byte[bufferSize];

        int len = 0;
        while ((len = inputStream.read(buffer)) != -1) {
            byteBuffer.write(buffer, 0, len);
        }
        return byteBuffer.toByteArray();
    }


    public  void removeSimpleProgressDialog() {
        try {
            if (mProgressDialog != null) {
                if (mProgressDialog.isShowing()) {
                    mProgressDialog.dismiss();
                    mProgressDialog = null;
                }
            }
        } catch (IllegalArgumentException ie) {
            Log.e("Log", "inside catch IllegalArgumentException");
            ie.printStackTrace();
        } catch (RuntimeException re) {
            Log.e("Log", "inside catch RuntimeException");
            re.printStackTrace();
```

```
        } catch (Exception e) {
            Log.e("Log", "Inside catch Exception");
            e.printStackTrace();
        }

    }

    public void showSimpleProgressDialog(Context context, String title,
                                         String msg, boolean isCancelable) {
        try {
            if (mProgressDialog == null) {
                mProgressDialog = ProgressDialog.show( context, title, msg );
                mProgressDialog.setCancelable( isCancelable );
            }
            if (!mProgressDialog.isShowing()) {
                mProgressDialog.show();
            }
        } catch (IllegalArgumentException ie) {
            ie.printStackTrace();
        } catch (RuntimeException re) {
            re.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }

    }
}
```