

ABSTRACT

IoT (Internet of Things) is the network of physical objects-devices, vehicles, buildings and other items embedded with electronics, software, sensors, and network connectivity-that enables these objects to collect and exchange data. Radio frequency identification (RFID) is a new generation of Auto Identification and Data collection technology, which helps to automate business process and identifies a large number of tagged student details using radio waves. In recent years, there have been rise in the number of applications based on Radio Frequency Identification (RFID) systems and is successfully functional in different areas as diverse as transportation, health-care, agriculture, and hospitality industry to name a few. Smart campus is a new idea in the development of information technology. The architecture of smart campus based on the cloud computing and the Internet of things consists of unified portal system, service support platform, data information convergence platform, network convergence platform, as well as information standard system and security maintenance. In this project, an effort is made to solve regular attendance monitoring using RFID technology. The application of RFID to student attendance monitoring as developed and deployed in this project is capable of eliminating wastage of time during manual collection of attendance and an opportunity for the educational administrators to capture face-to-face classroom data for allocation of proper attendance scores and for further managerial decisions. We have proposed IoT based attendance management system which updates the student's attendance in server spreadsheets and can be accessed from anywhere and SMS or push notification is sent to the respective mobiles. RFID system is used to monitor the student attendance but has some drawbacks regarding proxy. The solution proposed to that would be as follows: Once all the student's RFID tags are detected, our system initiates the camera. Camera will thus give us the head count. Management or students can access records from remote place. Using this, one can view or update the data and details about students and staff easily. This system helps in managing the activities like student admission, student registration, timetable management and result declaration. Admin can retrieve information of students. He can create a report about any student at any time using this system. You can submit student's fees and can check fees details anytime. Student can check their result online by logging into the system. You can also add new employee into the system and can check details of the employee easily. Student can also check course details online. This system facilitates colleges to maintain the functionality related to college employees and their students. It also helps teachers to update the internal marks, attendance etc. Keywords: Internet of Things, RFID, Smart campus

ACKNOWLEDGEMENT

At the very onset I would like to place on record my utter gratefulness to all those people who helped me in making this seminar a reality.

I would like to profoundly thank the **Management of R N S Institute of Technology** for providing such a healthy environment for the successful completion of this Seminar.

I would like to express my thank **Dr. H N Shivashankar, Director** and **Dr. M K Venkatesha, Principal**, for their encouragement that motivated me for the successful completion of this seminar.

I am extremely grateful to our beloved **Dr. G T Raju, Dean of Engg, Professor and HOD, Department of Computer Science and Engineering**, for having accepted to patronize me in the right direction with all his wisdom.

I place my heartfelt thanks to **Mr. Devaraju B M, Project Coordinator and Team members**. I would like to thank my Internal guide **Mr. Devaraju B M**, Asst. Prof., for his continuous guidance and constructive suggestions for this seminar work.

I Also thank all the staff members of Computer Science Department for encouragement and support.

On presenting the project report on **“IoT Based Smart College Campus using RFID”** I feel great to express my humble feelings to all who have helped me directly or indirectly in the successful completion of this work.

NISHITA PAL (1RN14CS062)

R. SAIPRASANTH (1RN14CS062)

SOORYA SAHAR P N (1RN14CS103)

SPOORTHY S (1RN14CS105)

TABLE OF CONTENTS

Sl.No. Chapter	Page No.
1. INTRODUCTION	01
2. LITERATURE SURVEY	04
3. PROBLEM STATEMENT	08
3.1 Existing System.....	08
3.2 Limitations of Existing System.....	09
3.3 Proposed System.....	09
3.4 Advantages of the Proposed System.....	10
4. REQUIREMENTS ANALYSIS AND FEASIBILITY STUDY	12
4.1 Software requirements.....	13
4.2 Hardware requirements	13
4.3 Feasibility Study.....	14
5. SYSTEM DESIGN	16
5.1 System Architecture.....	16
5.2 Overall System Functionality.....	17
5.3 Technologies used.....	18
6. DETAILED DESIGN	27
6.1 Attendance Functionality.....	27
6.2 Marks Functionality.....	28
6.3 Placement Details Functionality.....	29
6.4 Notes Upload Functionality.....	30
6.5 Push Notification Functionality.....	31
6.6 Entity Relationship diagrams.....	34
6.7 Level Diagrams.....	35
7. IMPLEMENTATION	38
7.1 Actors on the screen.....	38
7.2 Source code.....	41
8. RESULTS ANALYSIS	
8.1 Unit Testing.....	59
8.2 Integration Testing.....	59
8.3 System Testing	59
8.4 Results and Snapshots	60
BIBLIOGRAPHY	77

LIST OF FIGURES

Sl.No.	Figure Name	Page No.
5.1:	Physical Representation of System Implementation	16
5.2:	Block Diagram of System Functionality	17
5.3.2.1:	Position of JSP container and JSP files in a Web application	21
5.3.2.2:	Steps in the creation of web application	22
5.3.3.1:	Advantages of Android	23
5.3.3.2:	Android operating system components	24
6:	Main modules and their actors	27
6.1:	Attendance Functionality Module	29
6.2:	Marks Functionality Module	30
6.3:	Placement Details Functionality Module	31
6.4:	Notes Uploading Functionality Module	32
6.5:	Push Notification Functionality Module	33
6.6.1:	E-R Diagram for the relationships between the students and faculty	34
6.6.2:	E-R diagram for the relationship between students and the placement department	34
6.7.1:	Level-0	35
6.7.2:	Level-1	35
6.7.3:	Level-2	36
6.7.4:	Level-3	36
6.7.5:	Level-4	37

Chapter 1

INTRODUCTION

The success of any educational organization depends on its ability to acquire accurate and timely data about its operations, to manage this data effectively, and to use it to analyse and guide its activities. Integrated student database system offers users (Student, Registrar, HOD) with a unified view of data from multiple sources. To provide a single consistent result for every object represented in these data sources, data fusion is concerned with resolving data inconsistency present in the heterogeneous sources of data.

Information plays a vital role in the development and growth of every organization. Currently, the various departments manage student information independently in their own ways. There are no common, standardized process and program for capturing, processing and storing student's information. This has kept student information disintegrated in different departments and information provided to the various departments by the students is characterized with discrepancies. The various departments have systems in place to store and process student data but the systems are not able to talk to each other (Interoperability). This makes it difficult for the registrar to collate information of students across departments. For instance, if the registrar wants information about students with respect to their academic performance urgently, he must go to all the departments and collect the required data. On occasions where the department is not able to produce the needed information immediately, the business or activity at that particular time would come to a standstill. On the other hand, time is being wasted going around the various departments to solicit data. This situation is very frustrating and impedes smooth operations and decision-making process. Student Database System deals with all kind of student details, academic related reports, college details, course details, curriculum, batch details and other resource related details too. It tracks all the details of a student from the day one to the end of his course which can be used for all reporting purpose, tracking of attendance, progress in the course, completed semesters years, coming semester year curriculum details, exam details, project or any other assignment details, final exam result; and all these will be available for future references too.

Another major issue in the educational systems is the student's chronic absenteeism. Students are considered chronically absent when they miss 10 percent of days in a school/university year for any reason, that equals 18 days of school/university.

Many parents aren't aware of how many days their child is missing or the impact that can have on school/university success. Students with learning and attention issues are more likely to be chronically absent from school. It's hard to know exactly how many students are chronically absent. That's because most public schools and universities only report average daily attendance. They count the total number of students at university each day, but not the number of students who are frequently absent.

That's beginning to change, however. More states are beginning to look at, and report on, absenteeism. And the data is painting a troubling picture. Missing school/university in the early grades can have a snowball effect. It sets children up to fall behind in the fundamental reading skills they need in order to move on to more complicated work. For some students, frequent absences can become a long-term habit. Research shows that students who are allowed to miss school when they're young are more likely to skip school when they're older. And that can lead to other consequences. Students with learning and attention issues are even more vulnerable to the impact of chronic absences. It can be hard enough for them to master the lessons in school with the support of the teacher or aide. Trying to do it at home can make the work even harder. Plus, each day of learning builds on the previous day. When students miss a few days in a row, it can be hard to follow subsequent lessons. And when students aren't in school, they're missing the opportunities to be identified for intervention and extra supports.

Manual systems put pressure on people to be correct in all details of their work at all times, the problem being that people aren't perfect, however much each of us wishes we were. With manual systems the level of service is dependent on individuals and this puts a requirement on management to run training continuously for staff to keep them motivated and to ensure they are following the correct procedures. It can be all too easy to accidentally switch details and end up with inconsistency in data entry or in hand written orders. This has the effect of not only causing problems with service but also making information unable to be used for reporting or finding trends with data discovery. Reporting and checking that data is robust can be timely and expensive. This is often an area where significant money can be saved by automation. It takes more effort and physical space to keep track of paper documents, to find information and to keep details secure. When mistakes are made or changes or corrections are needed, often a manual transaction must be completely redone rather than just updated. With manual or partially automated systems information often has to be written down and copied or entered more than once. Systemisation can reduce the amount of duplication of data entry. Maintaining the attendance by the faculties manually is a difficult task.

Major drawbacks of the manual attendance system are:

Paper – Paper is inefficient, can be hard to read, easily damaged and lost but most of all it's a wasted resource that can be avoided.

Manual – Paper based time sheets depend on manual labour to keep them updated and correct without human interaction they are useless, having someone manage means a lot of unnecessary effort. This costs money and time.

Handwriting – Some people have terrible handwriting, it is an ultimate frustration to run through hundreds of lines trying to read what each person has written.

Time Accuracy – However honest people are 50 people do not arrive all at 9:00am every day and leave at 5:00pm. In fact, with paper-based time sheets, the cue to sign on and off will take some time.

Double Entry – Each student gets the attendance in their respective hours and teachers might end up doing it all over again into their attendance system.

The main objective of the proposed work is to make an IoT based platform to connect students, faculties and parents. It will maintain student's academic details and his/her attendance remotely through an android application and a website. The proposed system is intended to manage specific information of students such as personal details, course details and exam details etc remotely by the administrator, teachers and the students. This will be achieved by letting the students and teachers use the android application that will be developed for various uses, e.g., students can view his marks, placement status, attendance and also access the notes put up by the faculty members and faculty members can update the student's marks, their attendance and also upload the notes for the students. Administrator will be handling the website and take care of registering students, faculties, maintain their details etc.

Chapter 2

LITERATURE SURVEY

Campus Bridge mainly aims at eliminating the current traditional approach used in college working through a much more efficient system which involves IOT and other applications which improves the speed, accuracy as well faster and remote access to the required records, there by acting as a bridge which connects the students and teachers. There are other ways in which this same proposed system has been implemented so as to gap the bridge between the campus and students, there by introducing the concept of Smart Campus.

The Smart Campus project is a mobile as well as web application. It uses smart phones of android platform and web services on computer systems. The main objective is to develop an application that provides a smart and easy way for the execution of several academic operations to provide students with information regarding complaints, any placement activities, general notices, and important notices regarding all departments. In this system, the student has access to both the android application as well as web portal. The other faculties and principal (Admin) can access only the web portal. All the users need to login into the android application or web portal. The application and web portal have access to the shared database through the internet. Database is on Cloud. The android application is used only by the students. No one else has the access to the android application. There are various features in the application such as feedback, student corner, etc. All the users have access to the web portal including the students. The features include in the web portal are dependent upon the type of the user. Student can access various features on the application like to view the notice, to read the blog posted by the principal as well as other staff. The other feature is question answer. In this the student can post the question on the question answer section and the staff or other student can answer to that question so the redundancy of question will be reduced. The application is to make the interaction between the student and staff. Teachers can only view the notices sent by the principal or HOD. Teachers can have access to read the posts of student corner and they can give suggestions on it. Teachers can give the answers to the questions asked by the student on the question answer section. Teachers can write their own blog or post the notes on the blog and can view blogs of other staff. This project more oriented towards forming a campus portal rather than about various other important functions of a college such as handling of student, faculty records, attendance and marks management, etc.

One of the major day-to-day work to be carried out by a faculty or by a college is attendance management of each student for each class. The traditional way for taking attendance has drawback, which is the data of the attendance list hard to reuse. If the lecturer wants to calculate the percentage of the students that attend to the class, he/she has to calculate manually or input by typing. This also easy lead to human error such as the lecturer may wrongly. The technology-based attendance system will reduce the human involvement and decrease the human error. There are various types of attendance systems that are applied in different fields. Mostly, the working places are still using the punch card system. There have some researches that develop technology-based attendance system. Basically technology-based attendance system can be divided into two groups: i) Biometric-based Attendance System and ii) Sensor-based Attendance System.

Biometric-based attendance system recognize a person identity based on the biological characteristic such as fingerprint, hand geometry, voice, retina, iris and face recognition which reliably distinguishes one person from another or used to recognized the identity. They have five subsystems: data collection, signal process, matcher, storage and transmission. However, the biometric system is suitable for highly secured system and mostly the biometric system is expensive. As a whole, biometrics systems are known for its more expensive means of setup and operational costs. In term of its accuracy, biometrics attendance system prevents cheating and has lesser false alarm rate.

Kadry and Smaili implement an attendance system based on iris recognition. The system takes attendance as follows a) a digital image of one person's eyes to be verified is captured b) feature extracting algorithm is carried out c) minutiae are extracted and stored as a template for verifying later d) people to be verified place his eye on the iris recognition sensor and e) matching algorithm is applied to match minutiae. Talaviya implement a system that takes attendance of student by using fingerprint sensor module. When the student enrolls his/her finger on the finger print sensor module, his/her fingerprint will match with database to mark the attendance. The major disadvantage of this technology is that using the fingerprint scanner does not take into consideration when a person physically changes. For some people it is very intrusive, because is still related to criminal identification. Chintalapti and Raghunadh implement an automated attendance management system based on face detection and recognition algorithm.

Every time the student enters the class, his / her images will be capture by the camera placed in the entrance. The images will retrieve the identity of the student and take attendance for that student. They use Viola-Jones algorithm for the face detection part. There are five performance evaluation conditions used by them for the face recognition part, which are PCA + Distance Classifier, LDA + Distance Classifier, PCA + SVM, PCA + Bayes, LBPH + Distance Classifier.

In Sensor based technology, Barcode is a method of identification, which is used to retrieve in a shape of symbol generally in bar, vertical, space, square and dots which have different width with each one A reader of scanners are required to identify the data that represent by each barcode by using light beam and scan directly to barcode. During scanning process, a scanner measured intensity of reflected light at black and white region. A black region will absorb the light; meanwhile white region will reflect it. Smart card is built with variety of chip with a simple memory consisting of byte of information may have range from 1K up to 64K of microcontroller or multi-application memory. Smart card can use as individual identification, building access and network access are part of a multi-tiered program that is in the final stages of rolling out. The data in smart card can be read when a physical contact has a reader.

Ayush and Ahmad implement NFC supported attendance system in a University Environment named as TouchIn. Before the class start, the lecturer will run a mobile application on his/her own NFC-enabled smartphone, students that want to take the attendance will run another mobile application which will fetch the student ID from file, read the device ID and beam (send) it to the lecturer's device by simply touching the device. The attendance of the student will be taken. This system has disadvantages if compared to this project such as the accuracy is low on the identification part. The student can help his/her friend taking attendance, although his/her friend is absent. They just need to borrow their smartphone to his/her friend and his/her friend can scan the lecture's device with the smartphone and attendance will be taken. On a whole, NFC based system has a bit of initial cost investment involved. On the other hand, having a smart phone on through the class could lead to a sort of distraction to the students, thereby taking away the while purpose of attending a class.

Table 1: Comparison of Various Attendance Systems

Attendance System	Take student Attendance	Reuse Student Attendance Information	Prevent Cheating Issue	Fault Tolerance	Price
Traditional	✗				Cheap
Biometric	✗	✓	✓		Expensive
Barcode	✗	✓			Cheap
Smart Cards	✗	✓			Cheap
RFID	✓	✓	✓	✓	Medium
NFC	✓	✓			Medium

Chapter 3

PROBLEM STATEMENT

3.1 Existing System

The current existing systems of manual maintenance of records of all the information of corresponding to the teachers and students is still prevalent in a few colleges across the country. This is a very time consuming and error-prone process in comparison to the data being stored on the database. The database of students, teachers working in the organisation are retrievable based on queries run by the admin or authorized person. But in all or most of the existing systems, the GUI aided retrieval of information has yet to be provided directly by the students or teachers. The manual method of taking attendance in colleges over the years has become a thing of concern. In the manual method of taking attendance students are required to write down their names and sign the attendance list. The problems associated with this method vary from unnecessary time wastage to improper documentation, students forgetting to put down their names on the attendance list or students writing on behalf of other students that are absent from the class. This also another way of implementation through a roll-call done by the teacher of all the roll numbers in the class and students will have to mark their attendance based on when the number is called out. This method also can mitigate some of the proxies found in previous method as the teacher notices and confirms the person before jotting down the attendance for that student, but again it is very time consuming if the class has a big strength.

Coming to the teacher attendance systems, the newest form of system is that of the biometric system. In this automatic attendance system using fingerprint verification technique was proposed. The fingerprint technique verification was achieved using extraction of abnormal point on the ridge of user's fingerprint or minutiae technique. The verification confirms the authenticity of an authorized user by performing one to one comparison of a captured fingerprint templates against the stored templates in the database. The proposed automatic attendance system signals either true or false based on logical result of previous one to one verification of person's authenticity. There is also a biometric system using fingerprint identification for attendance automation of employees in an organization. But these systems will involve a lot of initial investment and biometrics systems will involve skilled person to handle the inner working and management of all the data accumulated by it. This system again does not provide any forum or platform where the teachers and students are updated about their information on a day-to-day basis.

3.2 Limitations of Existing System

One of the major drawbacks of the current existing system is of it manual collection of student and teacher records which would involve a lot of time and energy to be spent for accurate retrieval of information. The usage of database for storage of data is also not put to the fullest use in today's educational institutions. There are no available for real time checking of various information such as marks, attendance etc by the students, teachers. This existing system is also much more error-prone hence leads to confusion among students and related faculties. There are no readily available efficient ways by which information, notes or reports reach the students within reasonable time. In the case of important notification to be sent to students from the college still works in the way of notices, or a tedious way of sending messages to each and every student phone number which is expensive.

The other big drawback of the existing system is the time consumed in each class while taking attendance during class. This current manual attendance system is not just slow but also less accurate as in case of proxy attendance i.e. answering done for a student who is not present for that class by another student in the same class. This also increases the workload of teachers or related staff to manually update the attendance on a system or a database without any errors which requires high level of concentration and time from them. There are also issues related to campus placements where a student will have to wait till almost the last moment to know whether he is qualified to sit for the company based on the company's required aggregate, spending time or waiting in queues to get hold of the subject notes provided by the faculties rather than a softcopy. Finally, the lack of usage of the latest technology available according to the requirements at the hands for a much smother, efficient process in these educational institutions is the major cause of concern in the current existing system.

3.3 Proposed System

The main objective of the proposed work is to make an IoT based platform to connect students, faculties and parents. To develop a platform where the attendance and other information of the students can be accessed easily from remote places through the help of our smart phones. The platforms used are apps which would allow both teachers and students to access their portals based on their credentials and view or edit their related information. In this project, an effort is made to solve regular attendance monitoring using RFID technology. The application of RFID to student attendance monitoring as developed and deployed in this project is capable of

eliminating wastage of time during manual collection of attendance and an opportunity for the educational administrators to capture face-to-face classroom data for allocation of proper attendance scores. We have proposed IoT based attendance management system which updates the student's attendance in server spreadsheets and can be accessed from anywhere push notification is sent to the respective mobile apps.

RFID system is used to monitor the student attendance but has some drawbacks regarding proxy. The solution proposed to that would be as follows:

Once all the student's RFID tags are detected, our system initiates the camera. Camera will thus give us the head count. Teachers or Students can access records from remote place. Smart campus system is to automate all functionalities of a college or university. Using this system, you can manage all college activities like time table management and result declaration. Using this, one can view or update the data and details about students and staff easily. These functionalities and correct working of the proposed system is taken care by the Admin. There will be a separate admin portal which is RESTful web application which allows him to add, view, delete or even update the current information present in the record.

Admin can also retrieve information of students. He can create a report about any student at any time using this system. Using this system, you can register new student and their course details. Student can check their result online by logging into the system. You can also add new faculty into the system and can check details of the faculty easily. Student can also check course details online. This system facilitates colleges to maintain the functionality related to college employees and their students. It also helps teachers to update the internal marks, attendance etc.

3.4 Advantages of Proposed System

The major advantage presented through the implementation of our proposed project is the automated process of collection of data which was earlier much more human dependent. It also brings a lot more error-free and faster efficient way of doing the entire process. It embraces technology such as RFID to reduce the amount of workload on teachers during attendance. This automated process leads to a very stable and transparent flow of information from the management to the students or teachers which also improves the accessibility through the development of apps. It also provides a much safer and secure way to access, collect information as it requires credentials to login as students, teachers respectively.

It also reduces the workload on students to access the hardcopies of notes available in nearby xerox shops by making them all digitally available in their smart phones which propels them to learn on the go. Some of the other advantages in our system is the ease of use through a very a user-friendly interface which allows even a non-technical person to be an admin with just basic knowledge.

The double checking through head count in our attendance monitoring system leads to complete removal proxy attendance that prevails in today's classrooms. The camera used also is just a mobile camera which is present in the faculty's phone hence not requiring any extra cost to deploy the system in use. The usage of cloud notifications leads to a faster and more ideal way of sending alerts to the students which augurs well for the institution.

Chapter 4

REQUIREMENT ANALYSIS AND FEASIBILITY STUDY

4.1 Software Requirements

4.1.1 Navicat for MySql

Navicat is a series of graphical database management and development software produced by Premium Soft CyberTech Ltd. for MySQL, MariaDB, Oracle, SQLite, PostgreSQL and Microsoft SQL Server. It has an Explorer-like graphical user interface and supports multiple database connections for local and remote databases.

Navicat's rich feature set eases and streamlines database administration and provides comprehensive set of tools for the database.

4.1.2 NetBeans for Java

NetBeans is an open-source integrated development environment (IDE) for developing with Java, PHP, C++, and other programming languages. NetBeans is also referred to as a platform of modular components used for developing Java desktop applications for simplifying the development of Java Swing desktop applications.

4.1.3 JDK 8

The Java Development Kit (JDK) is a software development environment used for developing Java applications and applets. It includes the Java Runtime Environment (JRE), an interpreter/loader (java), a compiler (javac), an archiver (jar), a documentation generator (javadoc) and other tools needed in Java development.

4.1.4 Android Studio

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development.

Android Studio offers even more features that enhance your productivity when building Android apps, such as: C++ and NDK support.

4.1.5 OpenCV for Java

Open Source Computer Vision is a library of programming functions mainly aimed at real-time computer vision. The library is cross-platform and free for use under the open-source BSD license.

4.2 Hardware Requirements

4.2.1 RFID USB reader

A RFID Reader is a device that uses radio-frequency waves to wirelessly transfer data between itself and a RFID tag/label in order to identify, categorize and track assets.

4.2.2 Camera

Android phone's camera is used for the purpose of image processing.

4.2.3 Android phone

Android phone is required for deployment and testing of application.

4.3 Feasibility study

- **Product:** The final outcome of our project will be a smart campus which will have automated functionalities for the manual works required to maintain the details of students and faculties. It will also make attendance management lot easier. Users can access their accounts remotely.
- **Technical Feasibility:** Technically, the software and hardware such as android studio, NetBeans etc used are used in the current market extensively and thus will be available for a longer period of time. Since, Java will be used for the development of the project, so the project can be in use for a longer period of time.
- **Social Feasibility:** The proposed system will be feasible enough for users as in the current trend, android applications and websites are used in many number of ways. Almost 8/10 people are aware of using android applications and websites which enables them to use this proposed system very easily. It will be user-friendly and cost-effective. Thus, it will be acceptable by the users of the application provided they are guided through its working.
- **Economic Feasibility:** When the economy is considered, initial investment for the installation of RFID devices in the classrooms, locating the server and the storage of

information costs if any database other than MySQL is considered will be high. But once, it is in use, it can be used for many years, with lesser maintenance costs. The cost of the RFID devices will vary depending upon its frequency, distance, active or passive etc. Since the android application is available to the students of the university as an apk, installation of application is easier and cheaper.

- **Market Research:** The development of the proposed system is a major requirement by most of the educational organisations. The labour work shall be reduced. Time shall be utilised in a proper way. Paper work will be reduced as the details are directly updated in the server. Faster communication and remote accessing is the essence of the proposed system.
- **Alternative Solution:** Advanced methods can be used e.g., facial recognition, biometric identification etc. but costs when considered are considerably high as compared to the method the project uses.

Chapter 5

SYSTEM DESIGN

A system architecture is the structure of the components of a program or system, their interrelationships and the principles and guidelines governing their design and evolution over time. Relationships are both run-time and non-run-time and hence architecture is also expressed in terms of components and connectors. A system architecture is not only a product of requirements but equally a result of organizational goals, the architect's experience and her technical environment. The system design provides overall guidance on system functions, performance requirements, security requirements, platform characteristics, etc. This is one of the most important phases of the software development as it is easier and cost-friendly to correct the various flaws during system design rather than during implementation or testing stages. The entire system design requires special attention and detailing as to make the life easier for developers and also for testers to some extent as some of the quality goals are formed by using this system design. Thus, system design plays a crucial and important part in this project as well.

System design is the process of defining elements of a system like modules, architecture, components and their interfaces and data for a system based on the specified requirements. It is the process of defining, developing and designing systems which satisfies the specific needs and requirements of a business or organization. A systemic approach is required for a coherent and well-running system. Bottom-Up or Top-Down approach is required to take into account all related variables of the system. A designer uses the modelling languages to express the information and knowledge in a structure of system that is defined by a consistent set of rules and definitions. The designs can be defined in graphical or textual modelling languages. Unified Modelling Language (UML) is used to describe software both structurally and behaviourally with graphical notation. A flowchart is another type which provides a schematic or stepwise representation of an algorithm. Business Process Modelling Notation (BPMN) is used for Process Modelling language. Systems Modelling Language (SysML) is used for systems engineering and related works. The system design has been split on the basis of physical and logical architectures. There is also a functionality-based Activity Diagram which has been provided in the detailed design of the report which illustrates the various modules implemented in our project.

5.1 System Architecture

Smart Campus project involves IOT at its core of working. Hence, interconnection and communication between various devices is key to this project to become successful. Hence, the physical representation of this entire system architecture will present a better idea of how this project would be implemented and the various communications required to achieve goals.

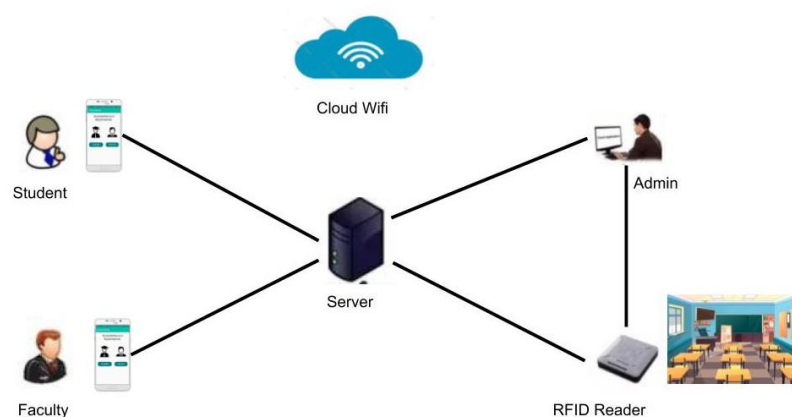


Figure 5.1: Physical Representation of System Implementation

The above pictorial representation shows the various devices and their connections with other devices which provide the project the hardware as well as the storage requirements of this project. The server plays a key role in this system as it hosts the web application as well as handles the various requests from the devices. The servers will also have access to the current operational information to be sent usually as responses to the devices. It remains operational 24x7 so as to give the students and faculty service throughout the day. Hence, the server has to remain robust and have a very little downtime so as to provide seamless service and uninterrupted working of the system. The admin system also has to be connected to the server through a particular port so as to form a communication channel between them. This is provided by the Wi-Fi router, which provides Wi-Fi signals connecting the various devices and providing a communication channel with the server through a communication port.

The smart phones provide the user-interface for the faculty and students for the various functionalities provided by the system. They all communicate through the server which hosts the various functionalities provided by the system. The platform supported by our project currently is Android but can be expanded to various other platforms such as IOS, Windows, Firefox OS, etc. The RFID system plays a crucial role in the attendance module of our system.

Hence, the RFID reader is placed outside or wherever convenient for the students to scan their provided RFID cards and enter the class. These RFID reader is connected to the server which automatically and instantly transmits the data collected on scanning by a RFID tag. There is no delay in this transmission, hence it provides a real-time sense of attendance monitoring. There is also usage of the faculty's smart phone camera to provide the image of the class to give a head count of the class through image processing for double confirmation.

5.2 Overall System Functionality

As mentioned earlier, System design has to provide detailed description of the modules involved to achieve the various goals. This system provides various functionalities divided into modules which form the basis of the system design architecture. The various modules involve various actors on the scene according to their requirements, hence it is critical to design it meticulously and without a flaw as it will stand as the backbone of the entire system implementation and its final working. The below given block diagram provides the overall functionality design of the system.

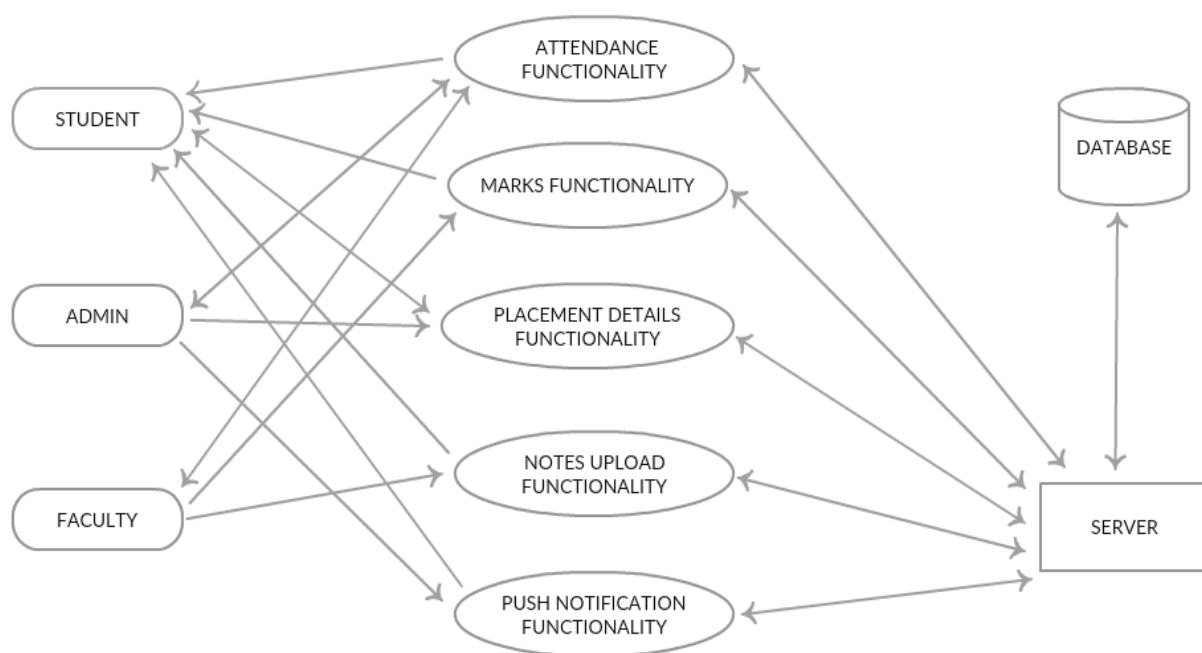


Figure 5.2: Block Diagram of System Functionality

The block diagram shows the major modules in the system and their various actors which use or require these modules. All these modules require the server to carry out their working, hence all modules are associated with the server.

The attendance functionality can be accessed by the student and the teachers from their smart phones for viewing and carrying out respectively. This attendance functionality also majorly involves the usage of RFID system and the camera.

The marks functionality module provides the students to view their internal marks through their smart phones and the uploading part of this functionality is carried out by the faculty. Hence, they are associated with both student as well as faculty. The placement details functionality gives the students opportunity to know about the companies their eligible to apply. Hence, this functionality involves the admin who is authorized to enter the respected company details according to which students can view their companies. The notes upload functionality involves the faculty who can upload any important notes or handouts which can be retrieved by the student remotely through their app. This functionality also requires a major part to be played by the database in its working. The push notifications functionality is to provide required news and information to all students and faculty who have the app about the campus or related information in real time. This notification is handled by the admin who is authorized to send these notifications whenever required.

5.3. Technologies used

5.3.1 Java

Java is an object-oriented multithreaded programming language. It is designed to be small, simple and portable across different platforms as well as operating systems.

5.3.1.1 Features of Java

- Platform Independence: the Write-Once-Run-Anywhere ideal has not been achieved (tuning for different platforms usually required), but closer than with other languages.
- Object oriented throughout - no coding outside of class definitions, including main ().
- An extensive class library available in the core language packages.
- Compiler/Interpreter Combination.
- Code is compiled to byte codes that are interpreted by Java virtual machines (JVM).
- This provides portability to any machine for which a virtual machine has been written.
- The two steps of compilation and interpretation allow for extensive code checking and improved security.
- Robust Exception handling built-in, strong type checking (that is, all data must be declared an explicit type), local variables must be initialized.

5.3.1.2 Several features of C & C++ eliminated:

- No memory pointers
- No pre-processor
- Array index limit checking

5.3.1.3 Security

- No memory pointers
- Programs run inside the virtual machine sandbox.
- Array index limit checking
- Code pathologies reduced by
 - o Bytecode verifier - checks classes after loading
 - o Class loader - confines objects to unique namespaces.

5.3.1.4 Dynamic Binding

- The linking of data and methods to where they are located is done at run-time.
- New classes can be loaded while a program is running. Linking is done on the fly.
- Even if libraries are re-compiled, there is no need to recompile code that uses classes in those libraries. This differs from C++, which uses static binding. This can result in fragile classes for cases where linked code is changed and memory pointers then point to the wrong addresses.
- Interpretation of byte codes slowed performance in early versions, but advanced virtual machines with adaptive and just-in-time compilation and other techniques now typically provide performance up to 50% to 100% the speed of C++ programs.
- Threading Lightweight processes, called threads, can easily be spun off to perform multiprocessing.
- Can take advantage of multiprocessors where available
- Great for multimedia displays.
- Built-in Networking Java was designed with networking in mind and comes with many classes to develop sophisticated Internet communications.

5.3.2. JSP(JAVA server pages)

Java Server Pages (JSP) is a technology for developing Web pages that supports dynamic content.

This helps developers insert java code in HTML pages by making use of special JSP tags, most of which start with `<%` and end with `%>`.

A Java Server Pages component is a type of Java servlet that is designed to fulfil the role of a user interface for a Java web application. Web developers write JSPs as text files that combine HTML or XHTML code, XML elements, and embedded JSP actions and commands.

Using JSP, you can collect input from users through Webpage forms, present records from a database or another source, and create Web pages dynamically.

JSP tags can be used for a variety of purposes, such as retrieving information from a database or registering user preferences, accessing JavaBeans components, passing control between pages, and sharing information between requests, pages etc.

JavaServer Pages often serve the same purpose as programs implemented using the Common Gateway Interface (CGI).

But JSP offers several advantages in comparison with the CGI.

- Performance is significantly better because JSP allows embedding Dynamic Elements in HTML Pages itself instead of having separate CGI files.
- JSP are always compiled before they are processed by the server unlike CGI/Perl which requires the server to load an interpreter and the target script each time the page is requested.
- JavaServer Pages are built on top of the Java Servlets API, so like Servlets, JSP also has access to all the powerful Enterprise Java APIs, including JDBC, JNDI, EJB, JAXP, etc.
- JSP pages can be used in combination with servlets that handle the business logic, the model supported by Java servlet template engines.

5.3.2.1 JSP architecture

The web server needs a JSP engine, i.e., a container to process JSP pages. The JSP container is responsible for intercepting requests for JSP pages. This tutorial makes use of Apache which has built-in JSP container to support JSP pages development.

A JSP container works with the Web server to provide the runtime environment and other services a JSP needs. It knows how to understand the special elements that are part of JSPs.

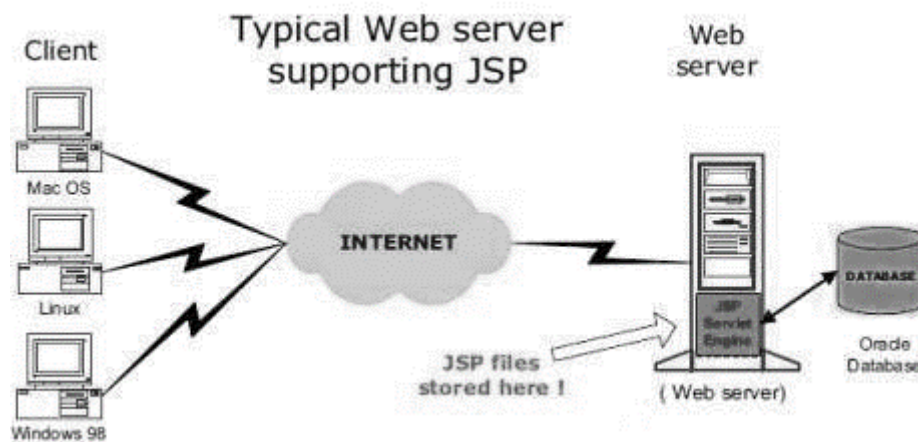


Figure 5.3.2.1: Position of JSP container and JSP files in a Web application

5.3.2.2 JSP Processing

The following steps explain how the web server creates the Webpage using JSP –

- As with a normal page, your browser sends an HTTP request to the web server.
- The web server recognizes that the HTTP request is for a JSP page and forwards it to a JSP engine. This is done by using the URL or JSP page which ends with .jsp instead of .html.
- The JSP engine loads the JSP page from disk and converts it into a servlet content. This conversion is very simple in which all template text is converted to `println()` statements and all JSP elements are converted to Java code. This code implements the corresponding dynamic behaviour of the page.
- The JSP engine compiles the servlet into an executable class and forwards the original request to a servlet engine.

- A part of the web server called the servlet engine loads the Servlet class and executes it. During execution, the servlet produces an output in HTML format. The output is further passed on to the web server by the servlet engine inside an HTTP response.
- The web server forwards the HTTP response to your browser in terms of static HTML content.
- Finally, the web browser handles the dynamically-generated HTML page inside the HTTP response exactly as if it were a static page.

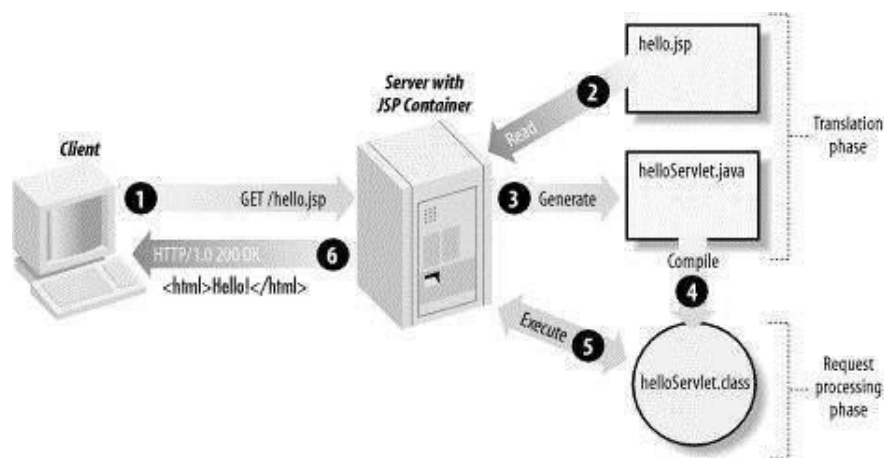


Figure 5.3.2.2: Steps in the creation of web application

Typically, the JSP engine checks to see whether a servlet for a JSP file already exists and whether the modification date on the JSP is older than the servlet. If the JSP is older than its generated servlet, the JSP container assumes that the JSP hasn't changed and that the generated servlet still matches the JSP's contents. This makes the process more efficient than with the other scripting languages (such as PHP) and therefore faster.

So in a way, a JSP page is really just another way to write a servlet without having to be a Java programming wiz. Except for the translation phase, a JSP page is handled exactly like a regular servlet.

5.3.3 Android

Android is an open source and Linux-based Operating System for mobile devices such as smartphones and tablet computers. Android was developed by the Open Handset Alliance, led by Google, and other companies.

Android offers a unified approach to application development for mobile devices which means developers need only develop for Android, and their applications should be able to run on different devices powered by Android.

The source code for Android is available under free and open source software licenses. Google publishes most of the code under the Apache License version 2.0 and the rest, Linux kernel changes, under the GNU General Public License version 2.

5.3.3.1 Why Android?

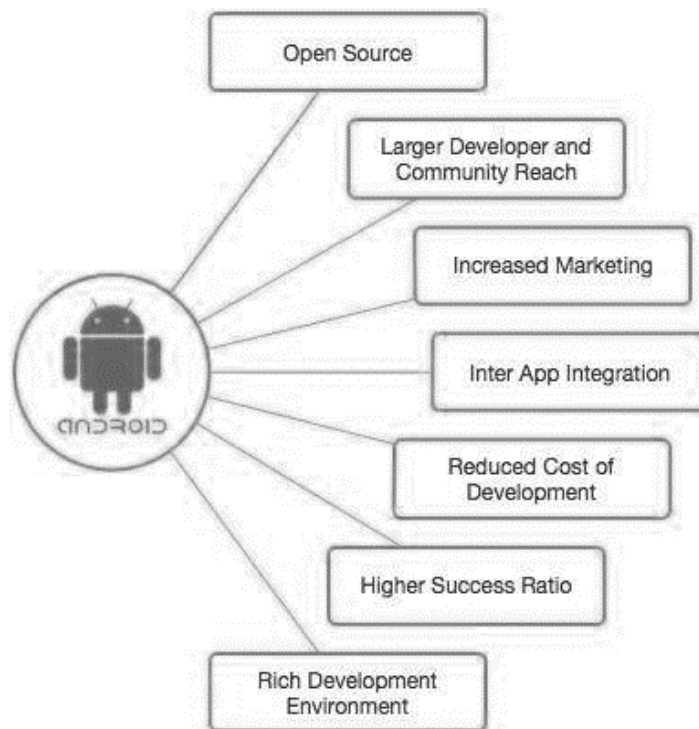


Figure 5.3.3.1: Advantages of Android

Android applications are usually developed in the Java language using the Android Software Development Kit. Once developed, Android applications can be packaged easily and sold out either through a store such as Google Play, SlideME, Opera Mobile Store, Mobango, F-droid and the Amazon Appstore.

Android powers hundreds of millions of mobile devices in more than 190 countries around the world. It's the largest installed base of any mobile platform and growing fast. Every day more than 1 million new Android devices are activated worldwide.

Android operating system is a stack of software components which is roughly divided into five sections and four main layers as shown below in the architecture diagram.

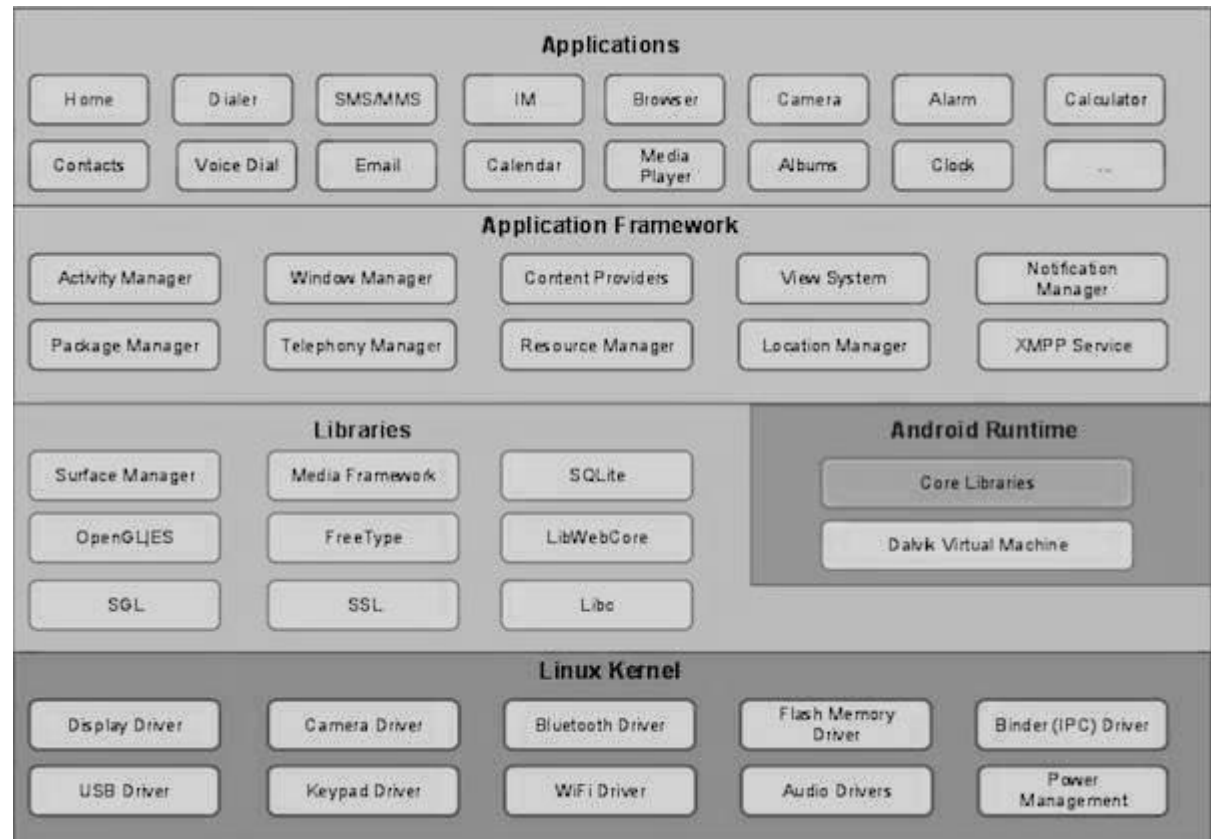


Figure 5.3.3.2: Android operating system components

Application components are the essential building blocks of an Android application. These components are loosely coupled by the application manifest file *AndroidManifest.xml* that describes each component of the application and how they interact.

There are following four main components that can be used within an Android application –

- **Activities:** They dictate the UI and handle the user interaction to the smartphone screen.
- **Services:** They handle background processing associated with an application.

- **Broadcast Receivers:** They handle communication between Android OS and applications
- **Content Providers:** They handle data and database management issues.

Android Studio is the official IDE for android application development. It works based on IntelliJ IDEA. The following features are provided in the current stable version:

- Gradle-based build support
- Android-specific refactoring and quick fixes
- Lint tools to catch performance, usability, version compatibility and other problems
- ProGuard integration and app-signing capabilities
- Template-based wizards to create common Android designs and components
- A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations
- Support for building Android Wear apps
- Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine
- Android Virtual Device (Emulator) to run and debug apps in the Android studio.

There are additional components which will be used in the construction of above mentioned entities, their logic, and wiring between them. These components are –

- **Fragments:** Represents a portion of user interface in an Activity.
- **Views:** UI elements that are drawn on-screen including buttons, lists forms etc.
- **Layouts:** View hierarchies that control screen format and appearance of the views.
- **Intents:** Messages wiring components together.
- **Resources:** External elements, such as strings, constants and drawable pictures
- **Manifest:** Configuration file for the application

5.2.4 OPENCV with Java

OpenCV is released under a BSD license and hence it is free for both academic and commercial use. It has C++, C, Python, and Java interfaces, and it supports Windows, Linux, Mac OS, iOS, and Android.

OpenCV was designed for computational efficiency and with a strong focus on real-time applications. Written in optimized C/C++, the library can take advantage of multi-core processing.

Some of the basic features of OpenCV are described below:

- **Smoothing Images:** This involves applying Blur, Gaussian Blur, medianBlur, and bilateral Filter.
- **Eroding and Dilating:** It can apply two very common morphology operators: Dilation and Erosion.
- **Morphology Transformations:** OpenCV function morphologyEx to apply Morphological Transformation such as opening, closing, TopHat, and BlackHat etc.
- **Image Pyramids:** OpenCV functions pyrUp and pyrDown to down sample or up sample a given image
- **Basic Thresholding Operations:** It can perform basic thresholding operations using OpenCV function threshold.
- **Adding borders to your images:** OpenCV function copyMakeBorder is used to set the borders (extra padding to your image).
- **Remapping:** In OpenCV, the function remap offers a simple remapping implementation.
- **Histogram Calculation:** For simple purposes, OpenCV implements the function calcHist, which calculates the histogram of a set of arrays (usually images or image planes). It can operate with up to 32 dimensions.

Digital Image Processing (DIP) deals with manipulation of digital images using a digital computer. It is a subfield of signals and systems but focuses particularly on images. DIP focuses on developing a computer system that is able to perform processing on an image. The input of such system is a digital image. The system processes the image using efficient algorithms and gives an image as an output. Java is a high level programming language that is widely used in the modern world. It can support and handle digital image processing efficiently using various functions.

Chapter 6

DETAILED DESIGN

Campus Bridge project deals with various functionalities which it handles which are separated out as modules. Some of the main modules handled are:

- Attendance
- Marks
- Placement Details
- Notes and Circular Uploading
- Push Notifications

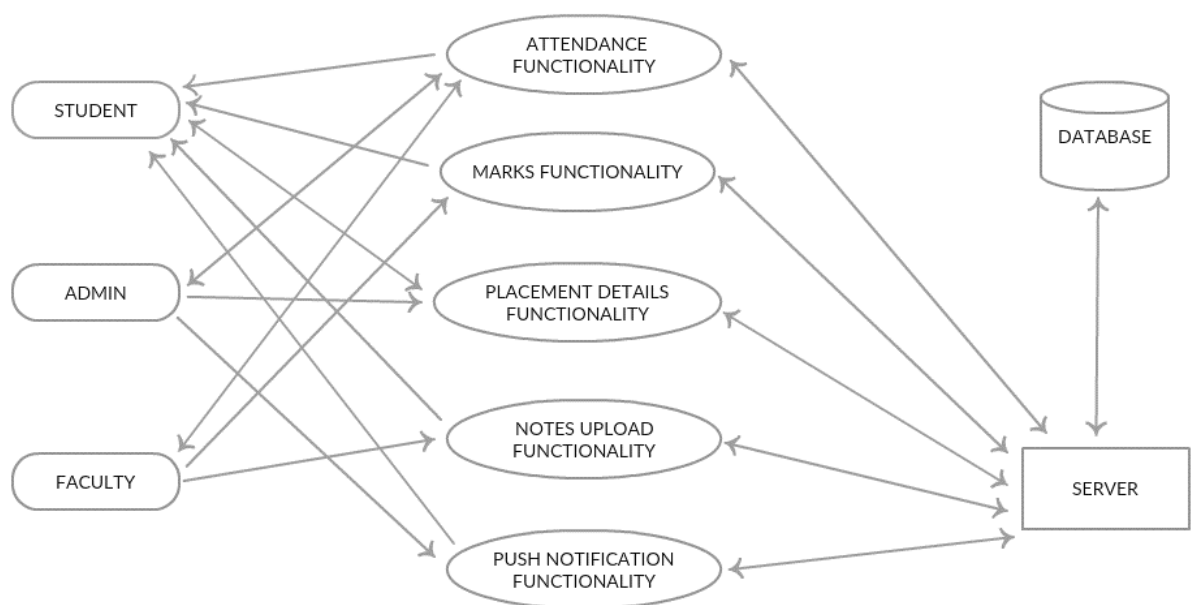


Figure 6: Main modules and their actors

These modules are carried out in various manners as illustrated with activity diagrams below.

6.1 Attendance Functionality

This functionality is a day-to-day basis or more a period-to-period carried out working. The main aim of this functionality is to reduce time wasted during attendance and also avoid proxies in class. The major role of this functionality is carried out by the RFID system in place. Each student has been assigned a unique RFID tag with a unique tag id to each of them. These tags are scanned by the RFID reader as they enter into the class. These entered RFID tag id's are instantly passed to the system by the reader.

Usually the scanning distance has to be within 10 cm. This tag id's and their associated USN are collected from the DB by the system and stored in a temporary table for that period. The associated and required details of the student is also retrieved by using the student USN from the student table. The next part of the working jumps to the faculty when the attendance has to be confirmed. The faculty opens the Campus Bridge app and logins with their respective credentials. The faculty selects the attendance section from their dashboard. This section requires to select the related subject code and section from the dropdown list which already holds the list of them handled by the faculty. Then, the selected subject code and section are sent to the system to be queried for the current list of students whose USN is available in the temporary table for that period. This USN list is dropped down along with checkboxes in the faculty app. The faculty is then prompted to capture a proper picture of the class which is sent automatically to the image processing system. The image processing system functions independently giving a head count of the class from the image. This number is tallied with the number of USN in the app list. If matched, the faculty directly updates the attendance for that period. If there is a difference, then faculty takes a proxy check and updates the attendance accordingly. Each of the USN with checked box, attendance for that particular subject is incremented by one.

Student is provided the view attendance functionality in the app. The student enters into the app and into the attendance section. Here, the system automatically retrieves the section, sem and other required details from the DB. The system queries the retrieved details to gather the various temporary table and subject-codes applicable to the student and automatically updates the attendance values in the main student table. This updated attendance values are displayed subject-code wise in the student app.

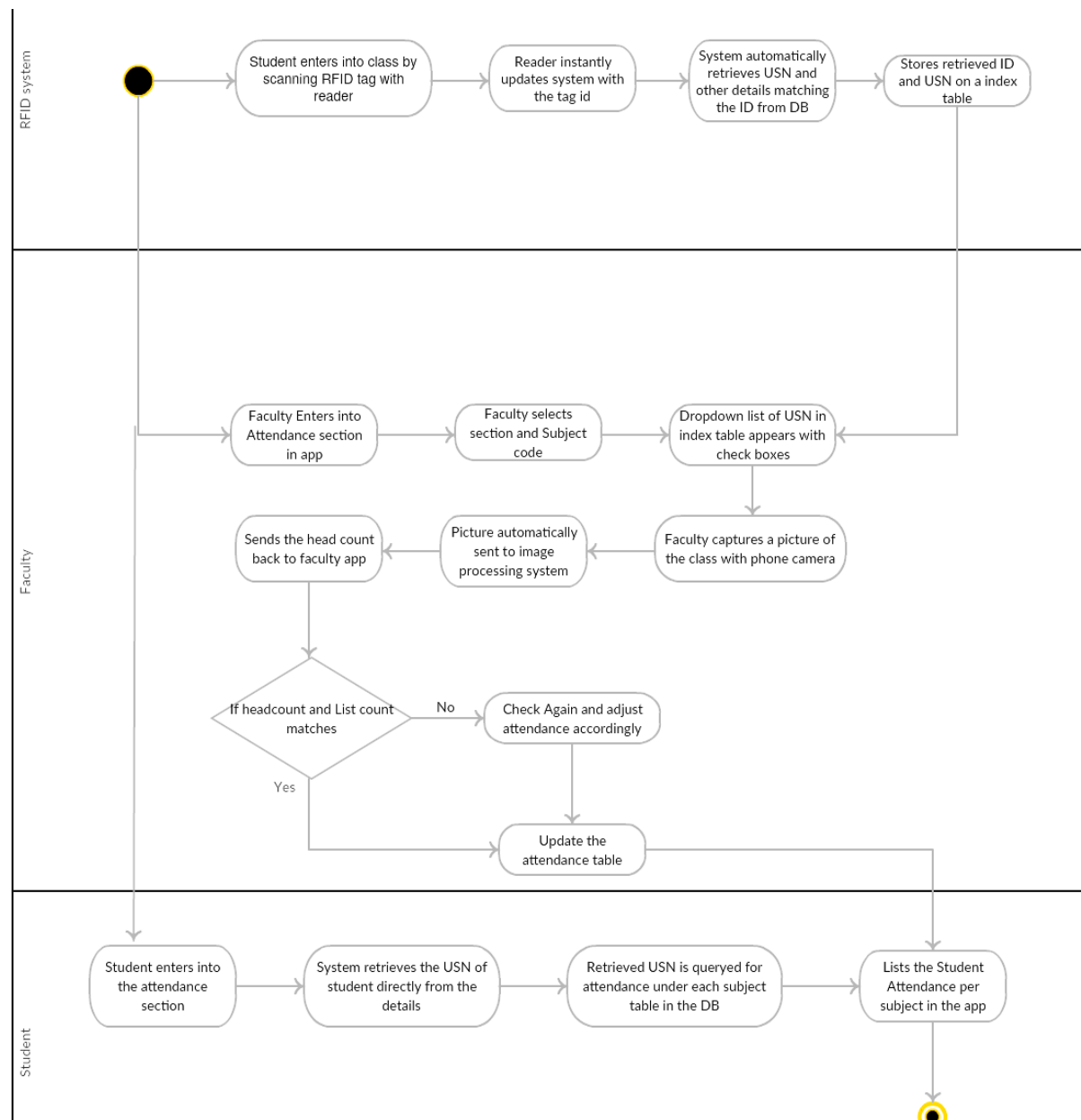


Figure 6.1: Attendance Functionality Module

6.2 Marks Functionality

This functionality is aimed at easy viewing and hassle-free updating of marks. This functionality starts with updating phase by each faculty. The faculty enters into the app along with the credentials. They are allowed to enter into the marks section in the app.

This section requires them to select the subject-code and section from the dropdown list which already holds the list handled by them. These details are queried by the system to retrieve the list of students who have taken that subject in that section. The USN of these selected students are displayed on the app where the faculty selects individually and updates the marks for the particular internals.

Student enters into the app and into the marks functionality. The system automatically retrieves the various details from the USN of the system from the student table. The marks are also retrieved from the USN from the student table with the help of subjects retrieved from the earlier working. These marks are displayed in the app subject-wise for each of the internals. Untaken or yet to be completed internals are displayed as 0 in the app.

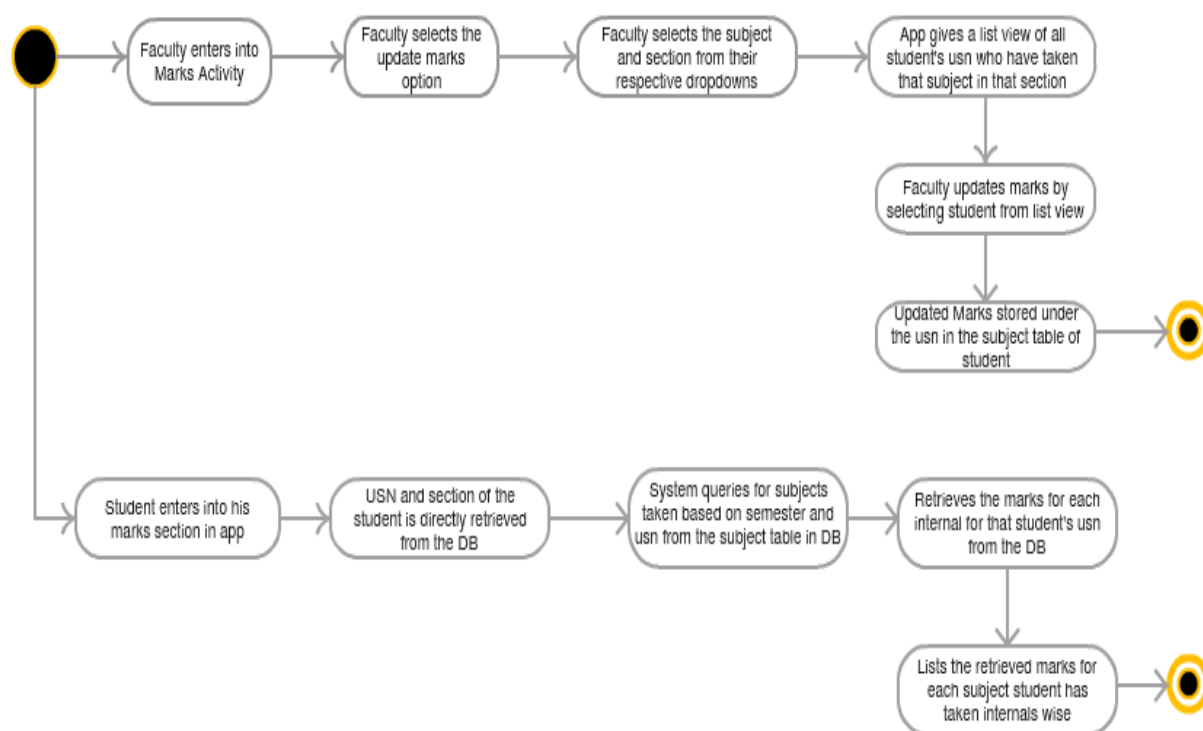


Figure 6.2: Marks Functionality Module

6.3 Placement Details Functionality

This functionality provides the various list of companies applicable to the student on his app with just his aggregate. The main aim is to provide the students with the students the various companies which they can look to seal a job and also to the junior years on the improvement to be done to achieve the various company interviews. The main role here is played by the admin.

The admin enters the various company details such as the name of the company, Tier of the company, eligible branches which are stored in the placement database.

Student enters into the app and the placement section. Here, the student is prompted to enter the current aggregate. This aggregate is used to filter the companies the student is eligible to apply. The tier of the company denotes the range of aggregate the student has to score to apply for the company. The other required details such as branch is taken automatically by the system. These details are queried accordingly and displays the list of companies on the app.

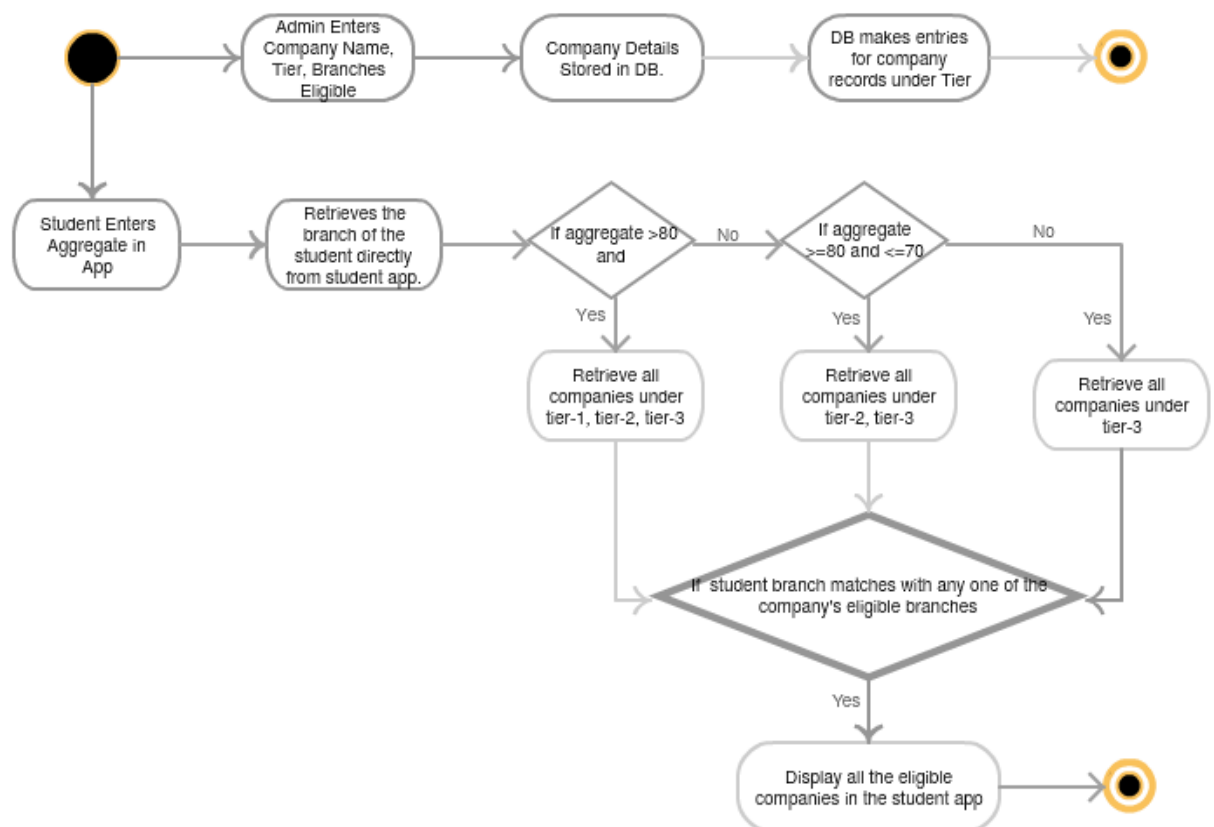


Figure 6.3: Placement Details Functionality Module

6.4 Notes Upload Functionality

The notes uploading functionality allows faculty to easily share their important notes and handouts with the students. Rather than like old times, where students queue up to take hard copy of notes outside shops, with the advent of smartphones into hands of students, having them in their phones is much more easy and feasible option. The faculty enters into the notes section in the app.

The faculty is prompted to enter the semester and subject to which the notes belongs to. Then, the faculty selects the notes to be uploaded from the local storage of their phone. The notes are uploaded to the database under the entered semester and subject-code folder. The student enters into his notes functionality which lists the various notes under the semester that the student belongs to. The student selects the notes accordingly and it automatically downloaded into the local storage of their phone which can be accessed. Most of the formats are supported and allowed to be uploaded by the faculty.

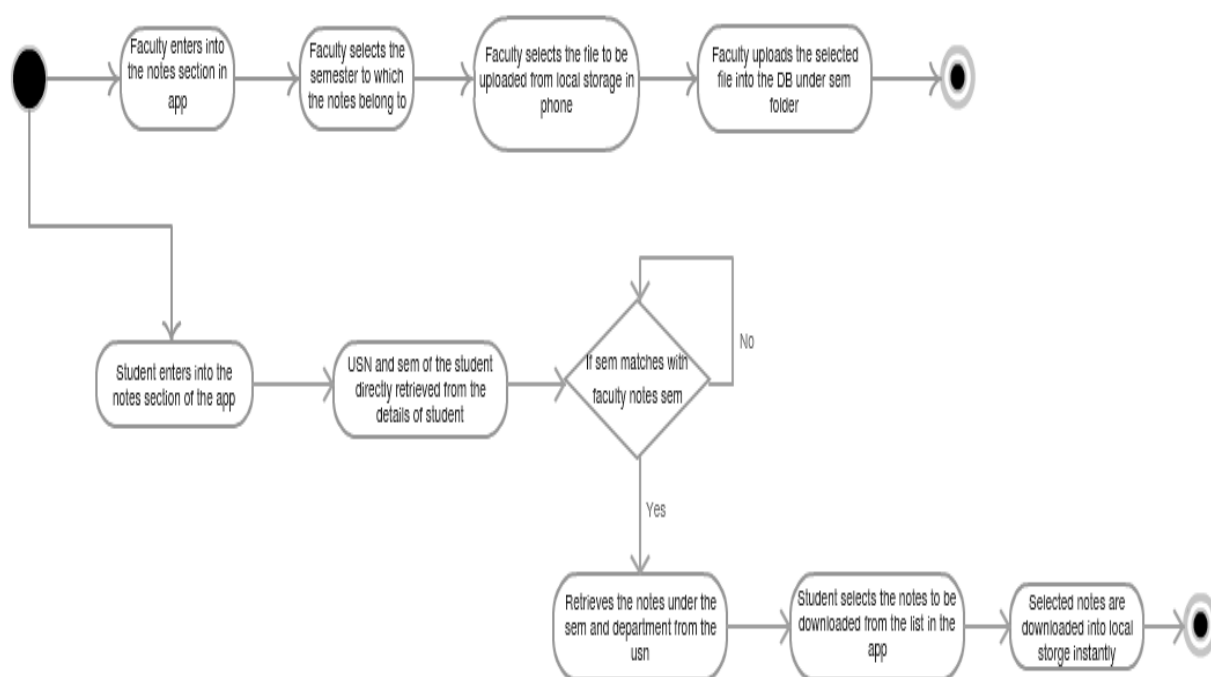


Figure 6.4: Notes Uploading Functionality Module

6.5 Push Notification Functionality

A push notification is a message that pops up on a mobile device. App publishers can send them at any time; users do not have to be in the app or using their devices to receive them. This push notifications functionality always provides a sense of convenience and value to the app users. In our Campus Bridge Project, the admin is entitled with the responsibility of handling all the push notifications to be sent to the faculty and the students. Our push notification setup is done by using the Google Firebase Cloud Messaging Service platform, which allows easier and convenient cross-platform messaging solution that lets you reliably deliver messages at no cost.

An FCM implementation includes two main components for sending and receiving. First, a trusted environment such as Cloud Functions for Firebase or an app server on which to build, target, and send messages. Finally, an IOS, Android, or web (JavaScript) client app that receives messages.

However, we have concentrated only on the Android Platform for our project. Thus, the admin has to login to his firebase console and select the cloud messaging option. This will lead him to a compose message page where the admin can enter the message along with many various options such as delayed delivery, topic subscription and send the message. The push notification is sent to every phone that has app either running in the foreground or background. Thus, students and faculty can be updated about events or any important information through this functionality in real time literally. Push Notifications also provide a cost-effective method over SMS or emails, as they can be sent free of cost and without remembering any phone numbers or email-ids as required for the other means of communication respectively.

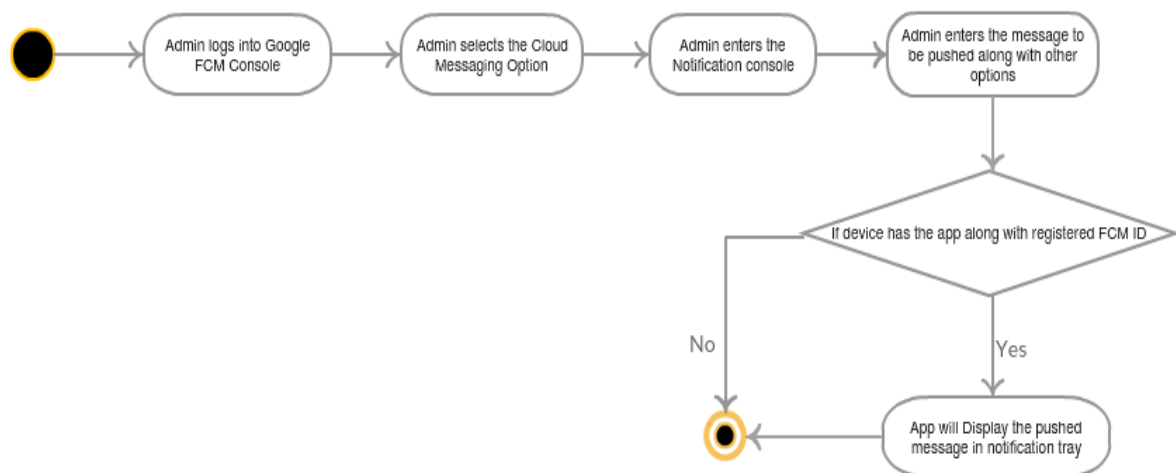


Figure 6.5: Push Notification Functionality Module

6.6 Entity relationships for the Proposed System

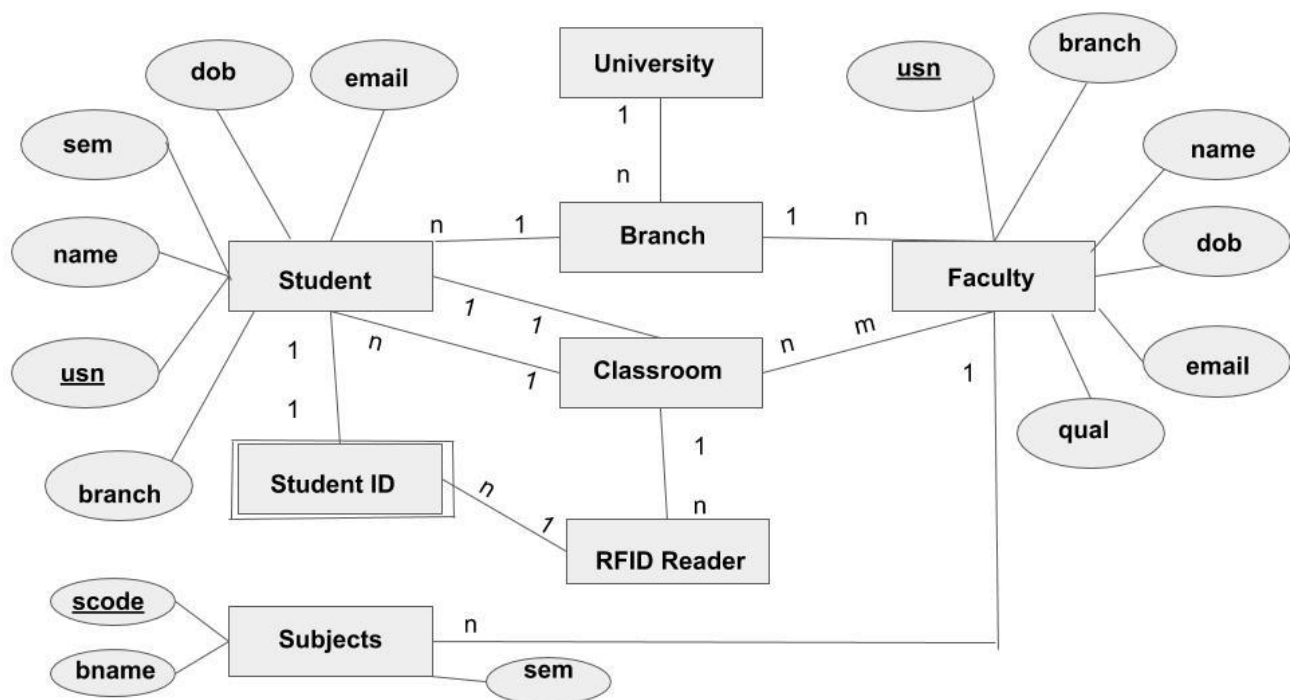


Figure 6.6.1: E-R Diagram for the relationships between the students and faculty

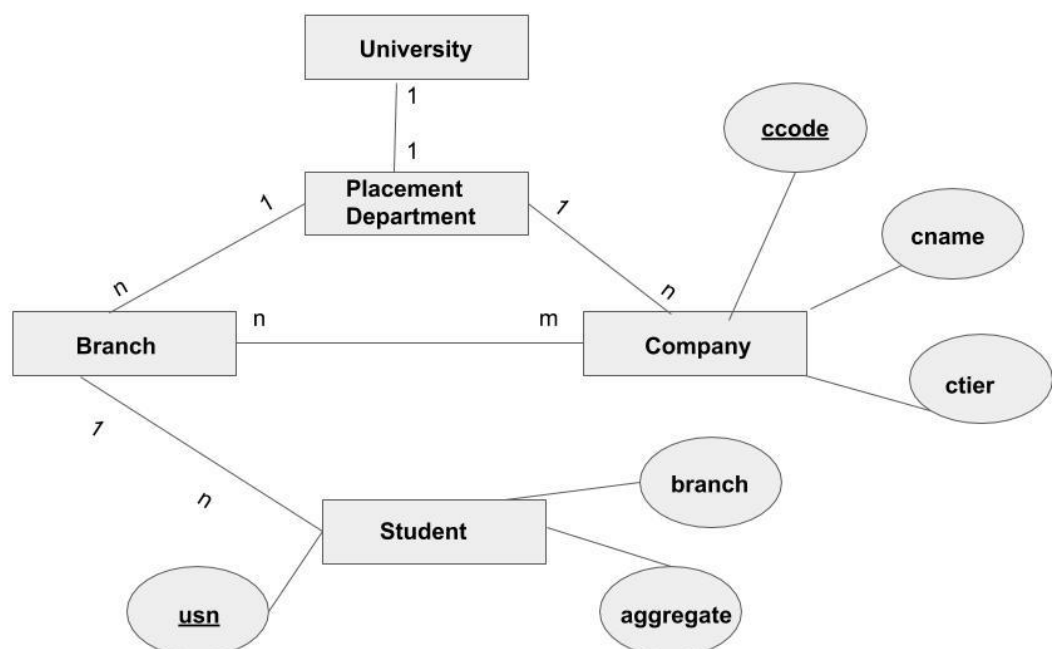


Figure 6.6.2: E-R diagram for the relationship between students and the placement department

6.7 Level diagrams for the system

Level-0

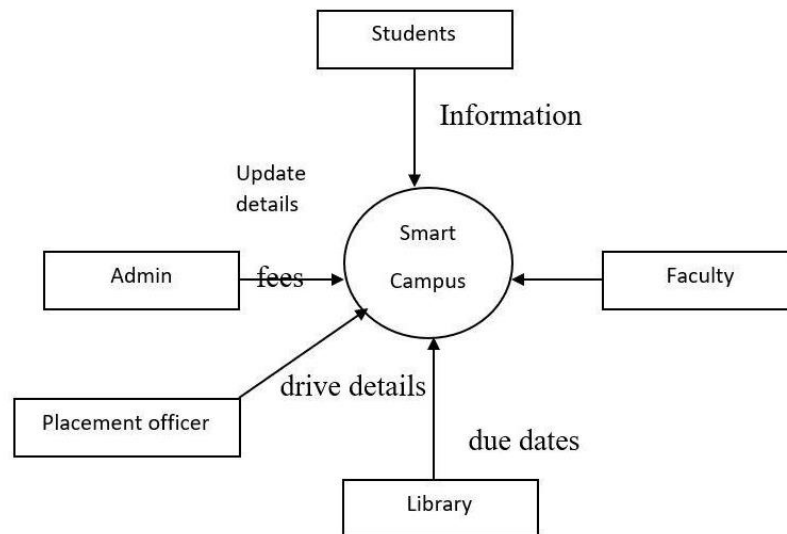


Figure 6.7.1: Level-0

Level 1

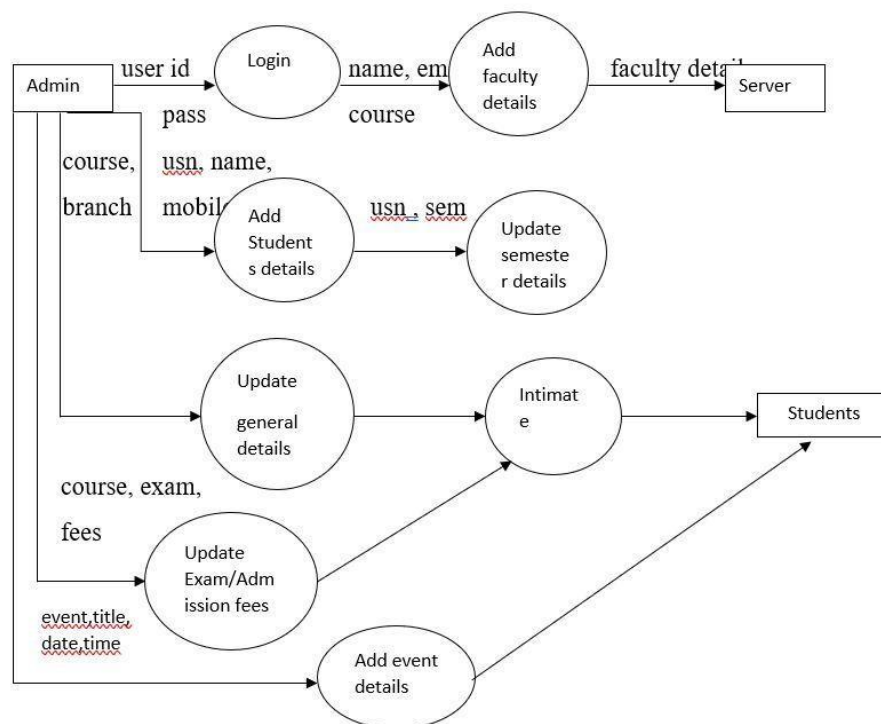


Figure 6.7.2: Level-1

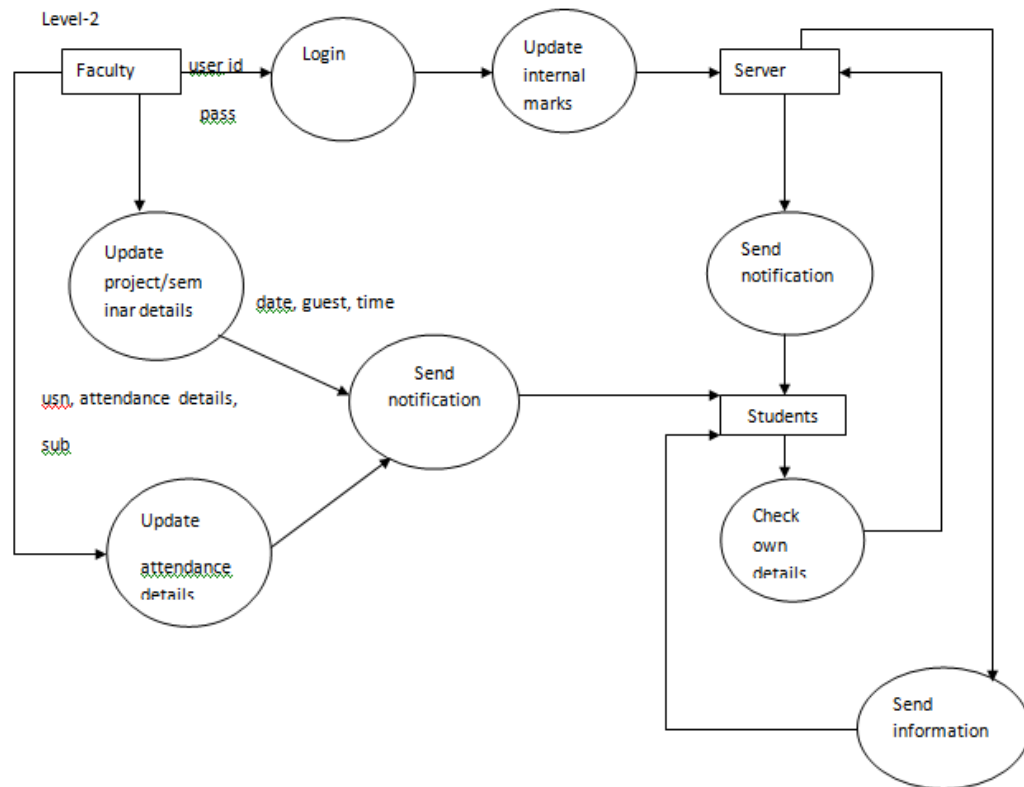


Figure 6.7.3: Level-2

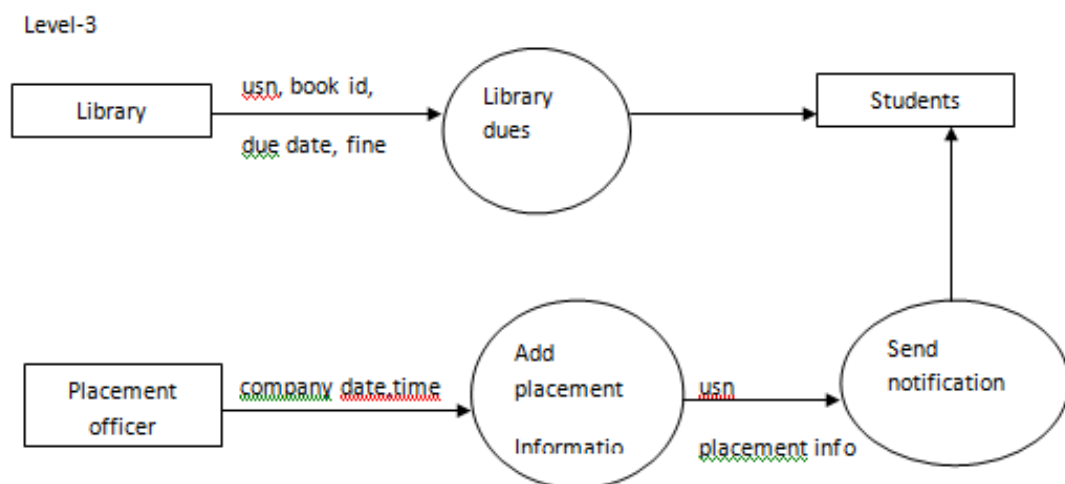


Figure 6.7.4:Level -3

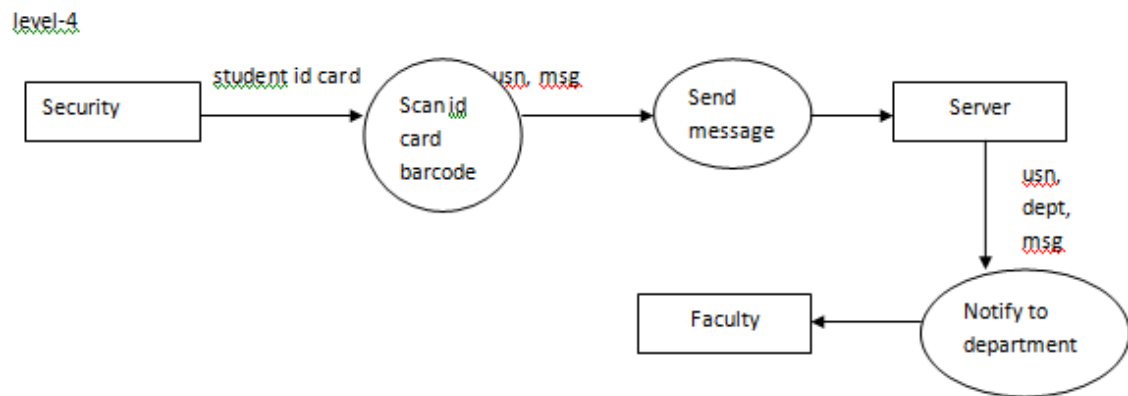


Figure 6.7.5: Level-4

Chapter 7

IMPLEMENTATION

The project implements Android Studio, MySQL 5.1, Navicat for MySQL, HTML, CSS, JAVA. JDBC API is used for connectivity between the Java programming language and MySQL and Bootstrap has been used for front end. The project is capable of running on standard internet web browsers. The interface is used to bridge the gap between students, teachers and the administration.

7.1 Actors on the screen

- Students
- Faculty
- Admin

1) Admin

A login page welcomes the admin where in the credentials of admin must be input. Once, the credentials match the data stored in the dB the admin 's login is successful and will be lead to the next page. The query written interacts with the admin_ information table to verify whether right details have been inserted.

As said before on successful login the admin is lead to his home page from the login page. The pages are named as home.jsp and login .jsp in NetBeans. The home page has a navbar towards its left and a home screen welcoming the administrator. The functionality of the navbar us to traverse the admin through the website and thus he will witness the various tasks he is able to perform. The options included in the navbar are Functionality, Placements, Attendance, and Logout.

1. **Functionality:** Choosing this option will scroll down to show the above picture. The admin is now able to add, delete, and view any student or faculty's details. The UI is comforting rather than the old school methods. On hovering on a student's image or a faculty's image the add, delete, view operations appear, on clicking any one of these will give away a form for the user to fill in the necessary details.

USN is required to view a student or a faculty details or to delete of their details. Once, the USN is inserted the individual's details pops up on the screen. The details contains every field in the form must be filled out for successful updation and addition.

Query communicates with tables named “student_information” & “ faculty_information”.

- **View** – Incase the admin wants to retrieve information about a student or a faculty , view button does the job . When View button is clicked and the USN is entered all the necessary data is fetched from the tables.
 - **Add** – On clicking the add button a form fills in the screen the admin has to fill in all the fields for successful addition of the individual into the system database.
 - **Delete**- If a person has to be eliminated or removed from the dB delete button has to be clicked , entering the USN and clicking submit will delete the data successfully from the respective tables.
 - **Update**- USN acts as a primary key in updating a person's details. The form is popped up again and the required fields can be updates for the person.
2. **Placements:** The placements option presents different fields for the admin to enter various companies visiting the campus and their requirements. The fields included are: Company name, Branches the company prefer, company code, company tier . Companies have been classified into various tires based of their perecentage requirement. The company_details table and the ctype tables are queried to add information.
 3. **Attendance:** This section is used assign RFID to every student. The RFID reader detects the tag and the student is assigned with a unique ID.
 4. **Logout:** On clicking logout the admin is directed back to the login page and thus he signs off.

2) Students and Faculty

While the admin is provided with a website, working on desktop the students as well as faculties are provided with an android app. Android studio has been used to implement this application.

Initially, a login page has been set up to read the credentials of the person, as per the data received the individual will be classified into a faculty or a student. If it is a faculty then UI, flow differs from that of the students.

Faculty – He / she will be able to update attendance and marks of students. The faculty can also upload notes for the students in the application. When the faculty clicks on marks, a new page is shown where a drop down of all the subjects he/she handles is thrown and the sections taught must be chosen. Once, the choices are made a list view is seen where in all the students' USN is show , the faculty just have to tap on the usn to enter the marks and on clicking submit all the students' marks are been uploaded . The “assign_fac” table contains details of each faculty's subjects and the sections they handle. Thus, we make use of this table to fetch the required content. RFID tags have been used to handle attendance. Every student scans their RFID before entering the class. Once, the faculty opens the app the attendance for all the students is already marked (i.e., a tick mark against every student's USN in the class). The app also gives the total number of students present in the class as well. The project also involves a camera, which takes the picture of all the class and returns the head count if there is any mismatch between the camera count, and the total attendance count returned by the app the teacher could always re check. The attendance update is initially stored in a “temp_attendance” table. Later, the attendance for that particular subject is updated in the respective subject's table.

Student – Student is given four options of viewing his Marks, attendance, placements details and finally downloading the notes. A tap on Marks will view the score gained by the student in each internal assessments. The table gives us the marks taken in every subject in that semester. Placements option gives an idea of students's eligibility, he/she can enter his percentage to check out the companies they can apply or are eligible for. Notes section gives away all the notes uploaded by the teacher for that semester.

The app also provides a navbar that on clicking helps one update his personal information or upload his/ her image. The students can also check their overall aggregate by the option provided. It also provides the Logout option for the user to end his session.

7.2. Source Code

Android

```
//student requests for attendance
@Override
protected String doInBackground(URL... urls) {
    try {
        URL url = new URL(RegURL.url + "getStudAtt");
        JSONObject jsn = new JSONObject();
        jsn.put("usn", susn);
        jsn.put("sem", ssem);
        res = HttpClientConnection.executeClient(url, jsn);
    } catch (MalformedURLException e) {
        e.printStackTrace();
    } catch (JSONException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }
}

return res;
}

//student view attendance from servlet
@Override
public View getView(int position, View convertView, ViewGroup
parent) {
    listPosititon = position;
    ViewHolder viewHolder = null;
    if (convertView == null) {
        LayoutInflater inflater = context.getLayoutInflater();
        convertView = inflater.inflate(R.layout.rowattendance,
null);

        viewHolder = new ViewHolder();
        viewHolder.tusn = convertView.findViewById(R.id.tvusn);
        viewHolder.tname = convertView.findViewById(R.id.tvname);
        viewHolder.chkbox= convertView.findViewById(R.id.chkbox);
        convertView.setTag(viewHolder);
        convertView.setTag(R.id.tvusn, viewHolder.tusn);
        convertView.setTag(R.id.tvname, viewHolder.tname);
        convertView.setTag(R.id.chkbox, viewHolder.chkbox);
    }
}
```

```

        else {
            viewHolder = (ViewHolder) convertView.getTag();

        }

        viewHolder.tusn.setTag(position); // This line is important.
        viewHolder.tname.setTag(position);
        viewHolder.chbox.setTag(position);
        viewHolder.tusn.setText(list.get(position).getUsn());
        viewHolder.tname.setText(list.get(position).getName());
        viewHolder.chbox.setChecked(list.get(position).getStatus());

        return convertView;
    }

//Faculty checks attendance
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_fac_attendance);
    usn=getIntent().getExtras().getString("usn");
    lv=findViewById(R.id.classlist);
    new fetchsec().execute();
    try {
        Thread.sleep(2000);
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
    response=response.trim().substring(1,response.length()-1);
    flist=response.split(",");
    for(int i =0;i<flist.length;i=i+2)
    {
        sec.add(flist[i+1]);
        section.append(flist[i+1]);
        subcode.append(flist[i]);
    }
    //Toast.makeText(this, section.toString()+subcode.toString(),
    Toast.LENGTH_SHORT).show();
    adapter=new
    ArrayAdapter<String>(FacAttendanceActivity.this,R.layout.activity_list_v
    iew,R.id.ltxtview,sec);
    lv.setAdapter(adapter);
    lv.setClickable(true);
    lv.setOnItemClickListener(new AdapterView.OnItemClickListener()
    {

        @Override
        public void onItemClick(AdapterView<?> arg0, View arg1, int
        position, long arg3) {
            Object o = lv.getItemAtPosition(position);
            str=(String)o;//As you are using Default String Adapter
            Intent i =new
            Intent(FacAttendanceActivity.this,FacAttendanceUpdateActivity.class);

```

```

        i.putExtra("usn", usn);
        i.putExtra("section", str);
        i.putExtra("strsec", section.toString());
        i.putExtra("strscore", subcode.toString());
        startActivity(i);
    }
    });
}

//faculty updates the attendance
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_fac_attendance_update);
    section=getIntent().getExtras().getString("section");
    fullsec=getIntent().getExtras().getString("strsec");
    facusn=getIntent().getExtras().getString("usn");

    fullscore=getIntent().getExtras().getString("strscore");
    subjspin=findViewById(R.id.subjcode);
    lv=findViewById(R.id.attlist);
    cap=findViewById(R.id.capture);
    update=findViewById(R.id.attup);
    String [] sec=fullsec.trim().split(" ");
    String [] sub=fullscore.trim().split(" ");
    subj = new
    ArrayAdapter(FacAttendanceUpdateActivity.this, android.R.layout.simple
    _spinner_item, sub);

    subj.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_it
    em);

    subjspin.setAdapter(subj);
    subjspin.setOnItemClickListener(new
    AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView<?> parent, View view,
        int position, long id) {
            strsubj = parent.getItemAtPosition(position).toString();

        }

        @Override
        public void onNothingSelected(AdapterView<?> adapterView) {

        }
    });
    new fetchtempatt().execute();
    try {
        Thread.sleep(2000);
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
}

```

```

        Toast.makeText(this, resp, Toast.LENGTH_SHORT).show();
        resp=resp.substring(1,resp.length()-1);
        String [] attlist=resp.split(",");
        lv.setChoiceMode(ListView.CHOICE_MODE_MULTIPLE);
        lv.setTextFilterEnabled(true);
        adapter=new
        ArrayAdapter<String>(this,R.layout.simple_checked_list_item,attlist);
        lv.setAdapter(adapter);

        lv.setClickable(true);
        for(int i=0;i<attlist.length;i++)
        lv.setItemChecked(i,true);
        lv.setOnItemClickListener(new AdapterView.OnItemClickListener()
        {

            @Override
            public void onItemClick(AdapterView<?> arg0, View arg1,
final int position, long arg3) {
                Object o = lv.getItemAtPosition(position);
                String str = (String) o;//As you are using Default
String Adapter
                CheckedTextView item = (CheckedTextView) arg1;
                Toast.makeText(FacAttendanceUpdateActivity.this,
lv.getCount()+" "+item.isChecked()+" "+position + " "+str,
Toast.LENGTH_SHORT).show();

            }
        });
        update.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                for(int i=0;i<lv.getCount();i++)
                {
                    if(lv.isItemChecked(i))
                    {
                        String b="true";

hmap.put(lv.getAdapter().getItem(i).toString(),b);
                    }
                    else
                    {
                        String b="false";

hmap.put(lv.getAdapter().getItem(i).toString(),b);

                    }
                }
                ArrayList<Map.Entry<String, String>> arrayList = new
                ArrayList<>();
                Set set=hmap.entrySet();

```



```

        Iterator i = set.iterator();
        while(i.hasNext()) {
            Map.Entry mentry = (Map.Entry)i.next();
            if(mentry.getValue().equals("true")) {
                s.append(mentry.getKey());
                s.append(",");
            }
        }
        Toast.makeText(FacAttendanceUpdateActivity.this, "Updated Successfully",
        Toast.LENGTH_SHORT).show();
        new addatt().execute();
        update.setClickable(false);
    }
});
cap.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent cameraIntent = new
        Intent(android.provider.MediaStore.ACTION_IMAGE_CAPTURE);
        startActivityForResult(cameraIntent, CAMERA_REQUEST);
    }
});

    }
    protected void onActivityResult(int requestCode, int resultCode,
    Intent data) {
        if (requestCode == CAMERA_REQUEST && resultCode ==
        Activity.RESULT_OK) {
            Bitmap bm = (Bitmap) data.getExtras().get("data");
            Toast.makeText(this, "<><><>"
            "+bm.getHeight()+" "+bm.getHeight(), Toast.LENGTH_SHORT).show();
            //imageView.setImageBitmap(photo);
            (int)(bm.getHeight()*0.5), true);
            ByteArrayOutputStream baos = new ByteArrayOutputStream();
            bm.compress(Bitmap.CompressFormat.JPEG, 100, baos);
            Toast.makeText(this, " "+bm.getHeight()+" "+bm.getHeight(),
            Toast.LENGTH_SHORT).show();

            byte[] b = baos.toByteArray();
            imgString = Base64.encodeToString(b, Base64.DEFAULT);
            URL url = null;
            try {
                url = new URL(RegURL.url + "Headcount");
            } catch (MalformedURLException e) {
                e.printStackTrace();
            }
            new SendPic().execute(url);
        }
    }
    private class SendPic extends AsyncTask<URL, Void, String> {
    @Override

```

```
protected String doInBackground(URL... urls) {

    try {
        URL url = urls[0];
        JSONObject jsn = new JSONObject();
        jsn.put("img", imgString);
        String response =
HttpClientConnection.executeClient(url, jsn);
        ires = response;
    } catch (MalformedURLException e) {
        e.printStackTrace();
    } catch (JSONException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }

    return ires;
} @Override
protected void onProgressUpdate(Void... values) {
    super.onProgressUpdate(values);
}

@Override
protected void onPostExecute(String s) {
    super.onPostExecute(s);
    Toast.makeText(FacAttendanceUpdateActivity.this, ires,
Toast.LENGTH_SHORT).show();
}
}
```

//student requests for marks

```
public class fetch_section extends AsyncTask<URL, Void, String> {

    @Override
    protected String doInBackground(URL... urls) {

        try {
            URL url = new URL(RegURL.url + "Student");
            JSONObject jsn = new JSONObject();
            jsn.put("usn", usn);
            jsn.put("sem", sem);
            jsn.put("ia", ia);
            res = HttpClientConnection.executeClient(url, jsn);
        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (JSONException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

```

        return res;
    }
//faculty view marks

@Override
protected String doInBackground(URL... urls) {

    try {
        URL url = new URL(RegURL.url + "StudDetails");
        JSONObject jsn = new JSONObject();
        jsn.put("scode",scode );
        jsn.put("section",semval );
        res = HttpClientConnection.executeClient(url, jsn);
    } catch (MalformedURLException e) {
        e.printStackTrace();
    } catch (JSONException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }

    return res;
}

//faculty update marks

getbtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        new FacUpdateMarks.GetStudInfo().execute();
        try {
            Thread.sleep(2000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        sresponse= sresponse.substring(1,sresponse.length()-1);
        Toast.makeText(FacUpdateMarks.this, sresponse,
Toast.LENGTH_SHORT).show();
        String usn[]= sresponse.split(",");
        ArrayAdapter<String>adapter=new
ArrayAdapter<String>(FacUpdateMarks.this,R.layout.activity_list_view,R.i
d.ltxtextView,usn);

        StrictMode.ThreadPolicy policy = new
StrictMode.ThreadPolicy.Builder().build();
        StrictMode.setThreadPolicy(policy);
        final HashMap<String,String> hmap= new HashMap<>();
        lv.setAdapter(adapter);
        lv.setClickable(true);
        lv.setOnItemClickListener(new
AdapterView.OnItemClickListener() {

```

```

@Override
    public void onItemClick(AdapterView<?> arg0, View arg1, final int
position, long arg3) {
        Object o = lv.getItemAtPosition(position);
        str=(String)o;//As you are using Default String Adapter

        Toast.makeText(FacUpdateMarks.this,position+"",Toast.LENGTH_SHORT).show(
        );
        final EditText txt = new EditText(FacUpdateMarks.this);
        AlertDialog.Builder builder = new
        AlertDialog.Builder(FacUpdateMarks.this);
        builder.setTitle("Enter Marks");
        builder.setView(txt);

        builder.setPositiveButton("OK", new DialogInterface.OnClickListener()
        {
            @Override
                public void onClick(DialogInterface dialog, int which) {
                    String name = txt.getText().toString();

                    Toast.makeText(FacUpdateMarks.this,name,Toast.LENGTH_SHORT).show();
                    hmap.put(str,name);
                }
            });

        builder.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {
            @Override
                public void onClick(DialogInterface dialog, int which) {
                    dialog.cancel();
                }
            })

        builder.show();
    }

    updatebtn.setOnClickListener(new View.OnClickListener() {
        @Override
            public void onClick(View view) {
                ArrayList<Map.Entry<String, String>> arrayList = new ArrayList<>();
                arrayList.addAll(hmap.entrySet());
                Log.i("qwerty ",arrayList.toString());
                Toast.makeText(FacUpdateMarks.this,
                "\n"+arrayList.toString(), Toast.LENGTH_SHORT).show();
                URL url = null;
                try {
                    url = new URL(RegURL.url+"FacUpdateMarks");
                    JSONObject jsn= new JSONObject();
                    jsn.put("upmarks",arrayList);
                    jsn.put("sem",strsem);
                    jsn.put("scode",strscore);
                    jsn.put("ia",ia);
                    response=
                    HttpClientConnection.executeClient(url, jsn);

```

```

        Toast.makeText(FacUpdateMarks.this, response,
        Toast.LENGTH_SHORT).show();
            } catch (MalformedURLException e) {
                e.printStackTrace();
            } catch (IOException e) {
                e.printStackTrace();
            } catch (JSONException e) {
                e.printStackTrace();
            }
        }
    });
}

});
}

}

@Override
protected void onPostExecute(String res) {
    res=res.substring(1,res.length()-1);

    Toast.makeText(Fac_View_Marks.this,res,Toast.LENGTH_LONG).show();
    arr= res.split(",");

    List<Model> list= new ArrayList<Model>();
    for(int i=0; i < arr.length; i+=4) {
        list.add(new
Model(""+arr[i].trim(),arr[i+1]+"\\t"+arr[i+2]+"\\t"+arr[i+3]));
    }
    adapter = new MyAdapter(Fac_View_Marks.this,list);
    listView.setAdapter(adapter);
}
}

```

//student downloads the notes

```

@Override
protected String doInBackground(String... f_url) {
    int count;
    try {
        URL url = new URL(f_url[0]);
        URLConnection conection = url.openConnection();
        conection.connect();
        int lenghtOfFile = conection.getContentLength();

        // download the file
        InputStream input = new
BufferedInputStream(url.openStream(), 8192);

        // Output stream
        OutputStream output = new
FileOutputStream(folder+"/"+str);

```

```
        byte data[] = new byte[1024];

        long total = 0;

        while ((count = input.read(data)) != -1) {
            total += count;
            publishProgress("" + (int) ((total * 100) / lenghtOfFile));

            output.write(data, 0, count);
        }

        output.flush();

        output.close();
        input.close();

    } catch (Exception e) {
        Log.e("Error: ", e.getMessage());
    }

    return null;
}

//faculty uploads the notes
public void uploadFile(String sourceFileUri) {
    String serverResponseMessage = "";
    String upLoadServerUri = RegURL.url + "FacUploadData";
    String fileName = sourceFileUri;
    HttpURLConnection conn = null;
    DataOutputStream dos = null;
    String lineEnd = "\r\n";
    String twoHyphens = "--";
    String boundary = "*****";
    int bytesRead, bytesAvailable, bufferSize;
    byte[] buffer;
    int maxBufferSize = 10 * 1024 * 1024;
    File sourceFile = new File(sourceFileUri);
    if (!sourceFile.isFile()) {
        Log.e("uploadFile", "Source File Does not exist");
    }
    try {
        FileInputStream fileInputStream = new
FileInputStream(sourceFile);
        URL url = new URL(upLoadServerUri);
        conn = (HttpURLConnection) url.openConnection(); // Open a
HTTP connection to the URL
        conn.setDoInput(true); // Allow Inputs
        conn.setDoOutput(true); // Allow Outputs
        conn.setUseCaches(false); // Don't use a Cached Copy
        conn.setRequestMethod("POST");
        conn.setRequestProperty("Connection", "Keep-Alive");
```

```

        conn.setRequestProperty("ENCTYPE", "multipart/form-data");
        conn.setRequestProperty("Content-Type", "multipart/form-
data;boundary=" + boundary);
        conn.setRequestProperty("uploaded_file", fileName);

        dos = new DataOutputStream(conn.getOutputStream());

        dos.writeBytes(twoHyphens + boundary + lineEnd);
        //dos.writeUTF("Content-Disposition: form-data;
name=\"lat\";lat=\""+ lat + "\"" + lineEnd);
        // dos.writeUTF(lon);
        dos.writeBytes("Content-Disposition: form-data;
name=\"uploaded_file\";filename=\""+ fileName + "\"" + lineEnd);

        dos.writeBytes(lineEnd);

        bytesAvailable = fileInputStream.available(); // create a
buffer of maximum size

        bufferSize = Math.min(bytesAvailable, maxBufferSize);
        buffer = new byte[bufferSize];

        // read file and write it into form...
        bytesRead = fileInputStream.read(buffer, 0, bufferSize);

        while (bytesRead > 0) {
            dos.write(buffer, 0, bufferSize);
            bytesAvailable = fileInputStream.available();
            bufferSize = Math.min(bytesAvailable, maxBufferSize);
            bytesRead = fileInputStream.read(buffer, 0, bufferSize);
        }

        // send multipart form data necesssary after file data...
        dos.writeBytes(lineEnd);
        dos.writeBytes(twoHyphens + boundary + twoHyphens +
lineEnd);

        // Responses from the server (code and message)
        serverResponseCode = conn.getResponseCode();
        serverResponseMessage = conn.getResponseMessage();

        //close the streams //
        fileInputStream.close();
        dos.flush();
        dos.close();

    } catch (MalformedURLException ex) {

        ex.printStackTrace();
        Toast.makeText(getApplicationContext(),
"MalformedURLException", Toast.LENGTH_SHORT).show();

```

```

        Log.e("Upload file to server", "error: " + ex.getMessage(), ex);
    } catch (Exception e) {

        e.printStackTrace();
        Toast.makeText(getApplicationContext(), "Exception : " +
e.getMessage(), Toast.LENGTH_SHORT).show();

    }

    Log.i("uploadFile", "HTTP Response is : " +
serverResponseMessage + ": " + serverResponseCode);
    if (serverResponseCode == 200) {
        runOnUiThread(new Runnable() {
            public void run() {
                dialog.cancel();
                Toast.makeText(getApplicationContext(), "File Upload
Complete.", Toast.LENGTH_SHORT).show();
            }
        });
    }
}

```

//placement activity

```

public class PlacementActivity extends AppCompatActivity {
    EditText aggval;
    String aggvalue;
    Button get;
    String usn;
    TextView txtcname;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_placement);
        aggval= findViewById(R.id.aggtxt);
        usn=getIntent().getExtras().getString("usn");
        get=findViewById(R.id.compbtn);
        txtcname= findViewById(R.id.tvcname);
        get.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                try
                {
                    txtcname.setText("");
                    aggvalue=aggval.getText().toString();
                    URL url = new URL(RegURL.url + "PlacementData");
                    JSONObject jsn = new JSONObject();
                    jsn.put("aggregate",aggvalue);
                    jsn.put("usn",usn);
                    StrictMode.ThreadPolicy pthread = new
StrictMode.ThreadPolicy.Builder().build();
                    StrictMode.setThreadPolicy(pthread);

```



```

String response = HttpConnection.getResponse(url, jsn);
String[] clist=response.split(":");

Toast.makeText(PlacementActivity.this,list.toString(),Toast.LENGTH_SHORT
).show();

        txtcname.append(clist[i]+"\\n");
    }

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

```

NetBeans

//update attendance from server

```

protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException, SQLException,
ClassNotFoundException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        DBQuery db=new DBQuery();
        int j,k;
        String li=request.getParameter("list");
        String sec=request.getParameter("sec");
        String scode=request.getParameter("scode");
        System.out.println("scode = " + scode);
        System.out.println("sec = " + sec);
        System.out.println("li = " + li);
        li=li.substring(0,li.length()-1);
        System.out.println("li = " + li);
        String []arr=li.split(",");
        System.out.println("arr = " + arr[0]+arr[1]);
        k=db.updateecc(scode,sec);
        for(int i=0;i<arr.length;i++)
            j=db.updateatt(scode,arr[i],sec);
        int l=db.getcc(scode,sec);
        System.out.println("l = " + l);
        ArrayList<String>ar=new ArrayList<>();
        ArrayList<String>arl=new ArrayList<>();
        ArrayList<Double>vals=new ArrayList<>();
    }
}

```

```

        ar=db.getattofstud(scode,sec);
        arl=db.getusnofstud(scode,sec);
        System.out.println("arl = " + arl);
        System.out.println("ar = " + ar);
        for(int i=0;i<ar.size();i++)
        {
            vals.add((Double.parseDouble(ar.get(i))/1)*100);
        }
        System.out.println("vals = " + vals);
        int m=db.updateattper(vals,arl,scode,sec);
        System.out.println("m = " + m);
    }
}

//get student attendance in the server
protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException, SQLException,
ClassNotFoundException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        String usn=request.getParameter("usn");
        String sem=request.getParameter("sem");
        System.out.println("sem = " + sem);
        String branch="";
        System.out.println("usn = " + usn);
        String ch=usn.substring(5,7);
        switch(ch)
        {
            case "CS":branch="CSE";
                break;
            case "EC":branch="ECE";
                break;
            case "EE":branch="EEE";
                break;
            case "IS":branch="ISE";
                break;
            case "ME":branch="MECH";
                break;
            case "CV":branch="CIV";
                break;
            case "EI":branch="EIE";
                break;
        }
        System.out.println("ch = " + branch);
        DBQuery db=new DBQuery();
        ArrayList <String> a=db.get_scode(sem,branch);
        System.out.println("a = " + a);
        StringBuilder att=new StringBuilder();
        for(int i=0;i<a.size();i++)
        {
            att.append(a.get(i));

```

```

        att.append(" ");
        att.append(db.get_att(a.get(i), usn));
        att.append(" ");
    }
    System.out.println("attval = " + att);
    out.print(att);
}
}

//get student marks in the server

protected void processRequest(HttpServletRequest request,
    HttpServletResponse response)
    throws ServletException, IOException,
    ClassNotFoundException, SQLException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        DBQuery db= new DBQuery();
        String scode1= request.getParameter("scode");
        String sec= request.getParameter("section");
        System.out.println("sec = " + sec);
        System.out.println("scode = " + scode1);
        ArrayList<String> stud =db.get_student_marks(scode1,sec);

        System.out.println("stud = " + stud);
        out.print(stud);
    }
}

//faculty view marks
protected void processRequest(HttpServletRequest request,
    HttpServletResponse response)
    throws ServletException, IOException,
    ClassNotFoundException, SQLException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        DBQuery db= new DBQuery();
        String usn1= request.getParameter("usn");
        System.out.println("usn1 = " + usn1);
        ArrayList<String>val= db.get_fac_assign(usn1);
        System.out.println("val = " + val);

        out.print(val);

    }
}
}

```

//faculty update marks

```

protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException, SQLException,
ClassNotFoundException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        DBQuery db=new DBQuery();
        String val= request.getParameter("upmarks");
        String scode=request.getParameter("scode");
        System.out.println("scode = " + scode);
        String sem=request.getParameter("sem");
        System.out.println("sem = " + sem);
        String ia=request.getParameter("ia");
        System.out.println("ia = " + ia);
        int k=0;
        if(val.length() > 0)
        {
            val= val.substring(1,val.length()-1);
        }
        System.out.println(val);
        String valarr[]= val.split(",");
        for(int i=0; i < valarr.length; i++){
            valarr[i]= valarr[i].trim();
            System.out.println("valarr = " + valarr[i]);
            String arr[]= valarr[i].split("=");
            for(int j=0;j<arr.length;j=j+2)
            {
                System.out.println(arr[j]+"\\t" + arr[j+1]);
                k=db.update_stud_marks(arr[j],arr[j+1],scode,ia);
            }
        }
    }
}

```

//placement details

```

protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException,
ClassNotFoundException, SQLException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter())
    {
        DBQuery db= new DBQuery();

        Double
        aggregate=Double.parseDouble(request.getParameter("aggregate").toString(
));

        System.out.println("aggregate = " + aggregate);
        String usn=request.getParameter("usn");
        String bcode=usn.substring(5,7);        String query="";
    }
}

```

```

System.out.println("bcode = " + bcode);

        bset.add("CS");
        bset.add("IS");
        bset.add("EC");
        bset.add("EE");
        bset.add("EI");
        bset.add("CV");
        bset.add("ME");

        if(!bset.contains(bcode))
        {
            System.out.println("Branch not defined...");
        }
        else
        {
            System.out.println("Branch available...");
            if(aggregate>=75)
            {
                System.out.println("inside 75");
                query="select cname from company_details where ccode=
any(select ccode from ctype where cbranch='"+bcode+"')";
            }
            else if(aggregate>=60)
            {
                query="select cname from company_details where (ctier
='t3' or ctier='t2') and ccode= any(select ccode from ctype where
cbranch='"+bcode+"')";
            }
            else if(aggregate>=50)
            {
                query="select      *      from      company_details      where
ctier='t3'and      ccode=      any(select      ccode      from      ctype      where
cbranch='"+bcode+"')";
            }
        }
        String value="";
        value= db.getCompany_Details(query);

        //      String splitval[]= value.split("\\:");
        out.print(value);
    }
}

```

// RFID Read

```

protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException,
ClassNotFoundException, SQLException {
    response.setContentType("text/html;charset=UTF-8");
    RequestDispatcher rd= null;
    HttpSession session= request.getSession(false);
    DBQuery db= new DBQuery();
    try (PrintWriter out = response.getWriter()) {
        /* TODO output your page here. You may use following
sample code. */
        String rfno=request.getParameter("rfidno");
        System.out.println("rfno = " + rfno);

        String [] stud =db.getstudusn(rfno);
        int i=
db.temp_att(rfno,stud[0],stud[1],stud[2],stud[3]);
        System.out.println("usn = " + stud[0]+stud[1]);
        if(i == 1)
        {
            session.setAttribute("id", stud[0]);
            session.setAttribute("name", stud[1]);
            session.setAttribute("msg", "Attendance updated
successfully for");
            rd= request.getRequestDispatcher("rfid.jsp");
            rd.forward(request, response);
        }
        else
        {
            session.setAttribute("id", stud[0]);
            session.setAttribute("msg", "OOps! Something went
wrong...");
            out.print("Something went wrong.");
        }
    }
}

```

Chapter 8

RESULT ANALYSIS

8.1 Unit Testing

Unit testing focuses on verifying the effort on the smallest unit of software-module. The local data structure is examined to ensure that the data stored temporarily maintains its integrity during all steps in the algorithm's execution. Boundary conditions are tested to ensure that the module operates properly at boundaries established to limit or restrict processing.

8.2 Integration Testing

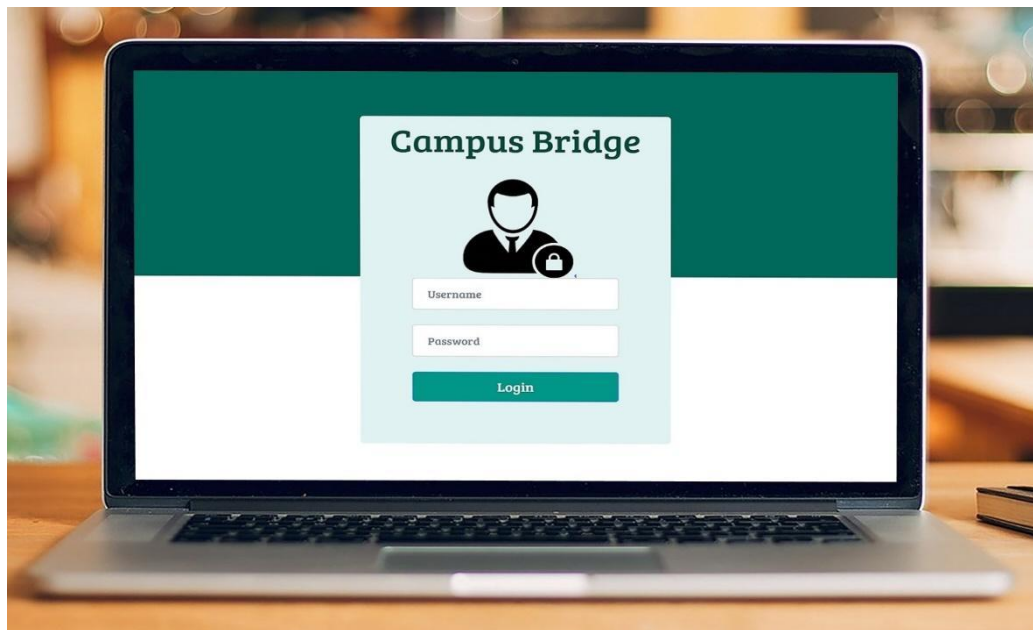
Data can be tested across an interface. One module can have an inadvertent, adverse effect on the other. Integration testing is a systematic technique for constructing a program structure while conducting tests to uncover errors associated with interring.

8.3 System Testing

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic. System testing is performed on the entire system in the context of a Functional Requirements Specification (FRS) and/or a System Requirements Specification (SRS). System testing tests not only the design, but also the behavior and even the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software/hardware requirements specifications. After finishing the development of any computer-based system the next complicated time-consuming process is system testing. During the time of testing only the development company can know that, how far the user requirements have been met out, and so on.

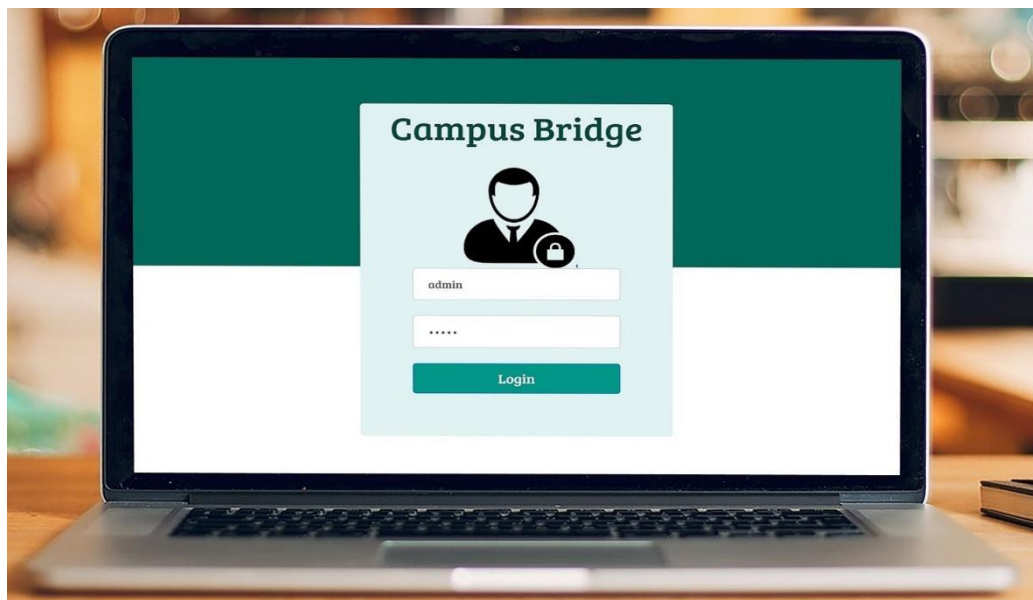
Software testing is an important element of the software quality assurance and represents the ultimate review of specification, design and coding. The increasing feasibility of software as a system and the cost associated with the software failures are motivated forces for well planned through testing.

8.4 Results and Snapshots



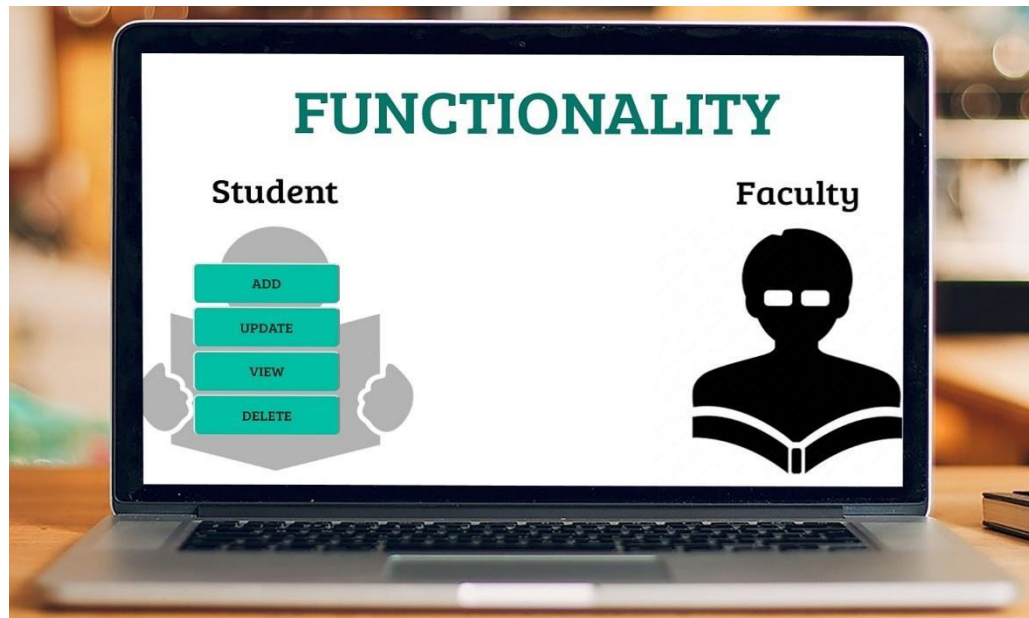
Snapshot 8.1: Admin Login Page

The above snapshot shows the first page of the web portal provided to the admin, which is the Login page for the admin. This page requests for the credentials required to authenticate the admin and successfully proceed to the Home page of the web portal.



Snapshot 8.2: Admin Login Page with Credentials

This above snapshot shows the login page after entering the credentials i.e. username and password by the admin. On pressing the Login button, it checks for the valid username and corresponding password.



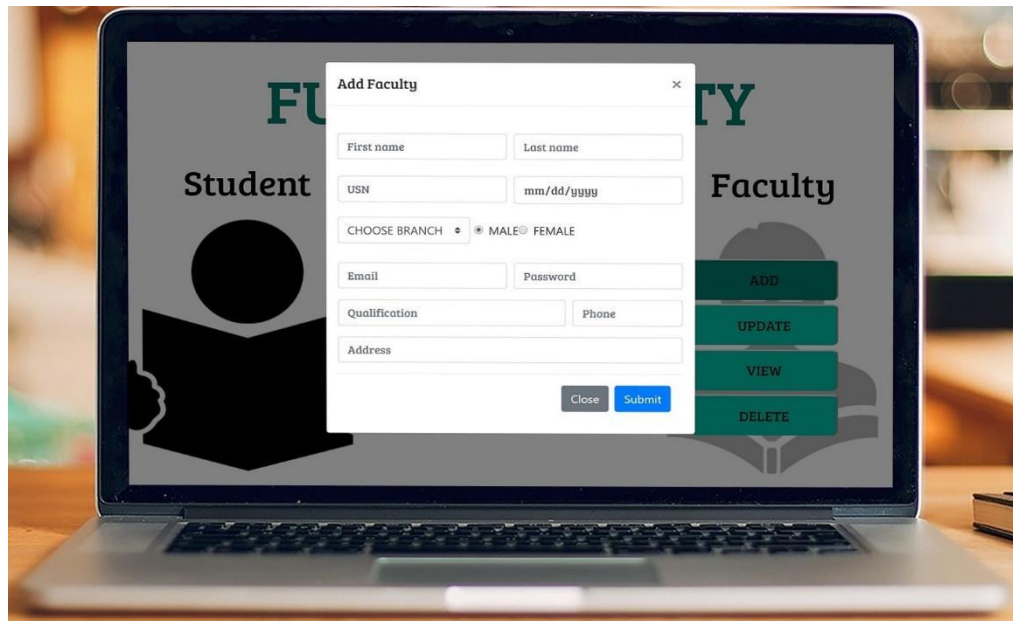
Snapshot 8.3: Functionality Page

This above snapshot shows the admin functionality page where the various options of insertion, deletion, updating and viewing of both the students and faculties is present. This page represents the front end of CRUD functionality provided to the admin.



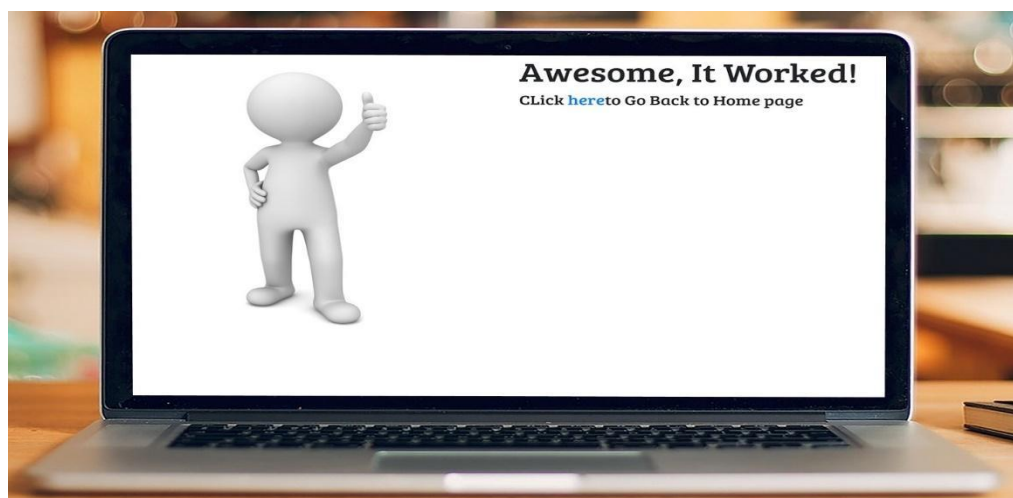
Snapshot 8.4: Add New Student Functionality

This above snapshot shows the modal, which appears on selecting the ADD option in the student, section of the functionality page. This modal allows adding the required student record in the corresponding fields, which on submission would be stored in the database. Validation of these fields in the form have been handled, hence these student record stored are all valid entries.



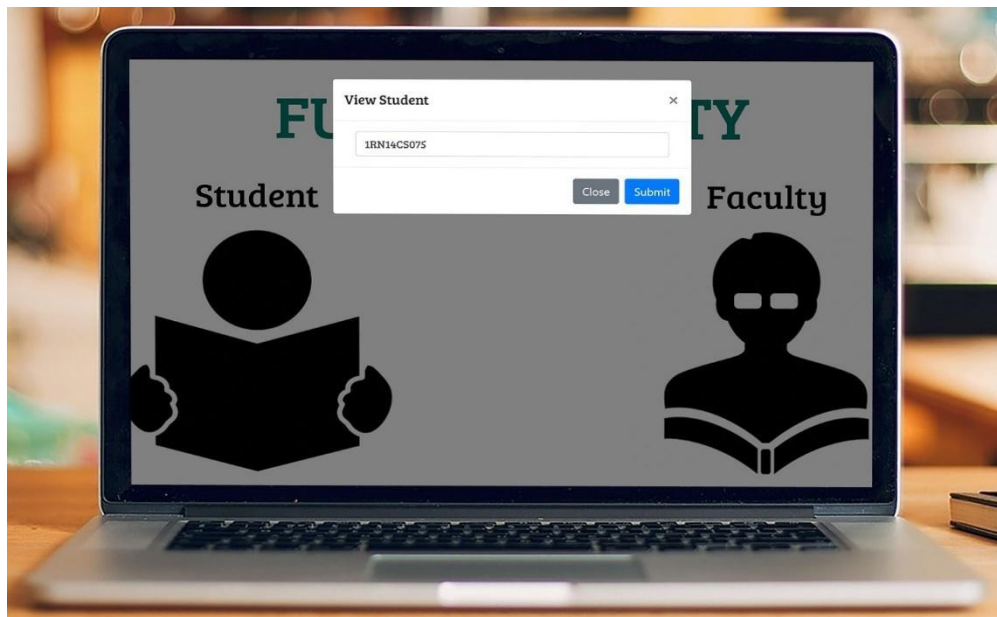
Snapshot 8.5: Add New Faculty Functionality

This above snapshot is similar to the add student functionality with the corresponding fields changed according to the faculty details. The same faculty details stored in the database can be retrieved for the various other requirements in the system.



Snapshot 8.6: Success Page

This above snapshot shows the success page, which confirms the successful addition, deletion or updating done by the admin in either the student or faculty record.



Snapshot 8.7: View Student Functionality

This above snapshot shows the view student modal which appears on selecting the view option in the student section. This requires the USN of the student that retrieves the corresponding record of the student stored in the database.



Snapshot 8.8: Student Details Page

This snapshot shows the retrieved student details from the database by the USN of the student as shown above.



Snapshot 8.9: View Faculty Details Functionality

This above snapshot shows the view faculty modal which appears on selecting the view option in the faculty section. This requires the USN assigned to the faculty that retrieves the corresponding record of the faculty stored in the database.



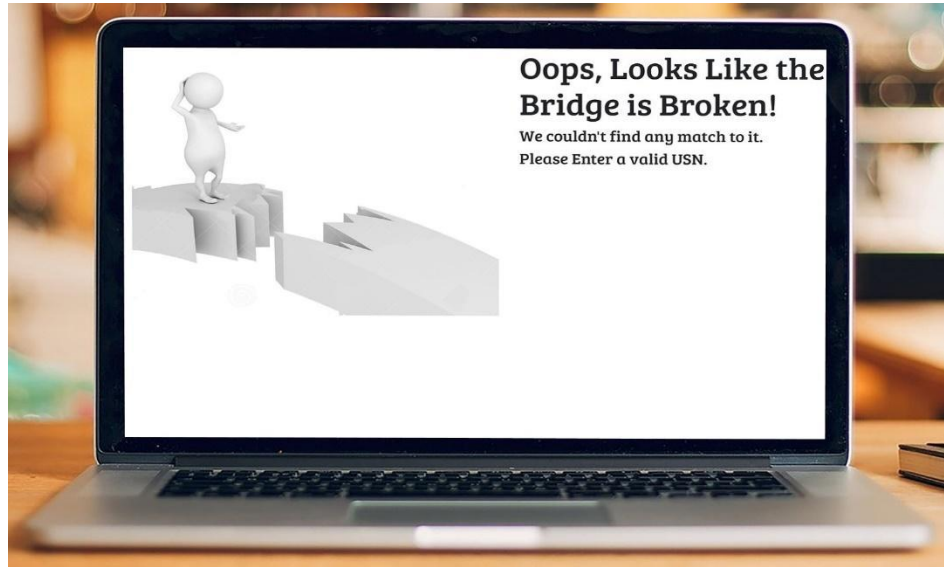
Snapshot 8.10: Faculty Details Page

This above snapshot shows the corresponding details of the USN of the faculty given in the above modal. This shows that the USN is the primary key of the records of the students and faculties stored in the database.



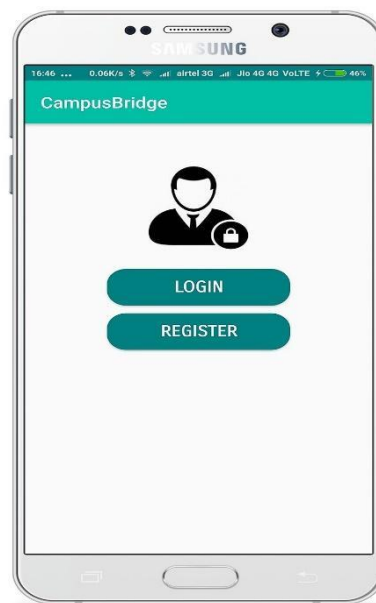
Snapshot 8.11: Placement Details Functionality

This above snapshot shows the placement functionality page which is also handled by the admin. This functionality allows the various companies along with their required eligibility criteria such as branches, overall minimum aggregate and as well as the tier under which the company falls in place. This allows for a much easier access to the information various companies students are eligible to apply for.



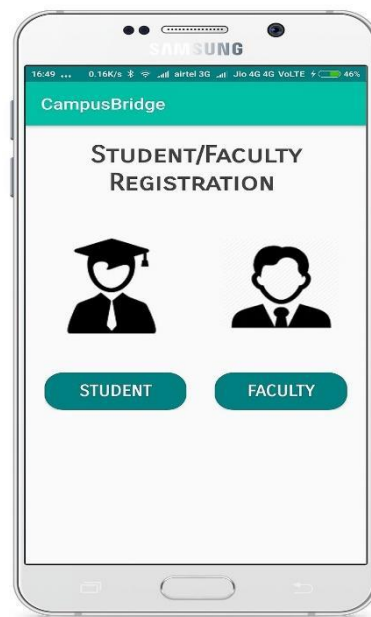
Snapshot 8.12: Error Page

This above snapshot shows the error page that handles any uncertain conditions such as unable to access the required details, USN not found, etc. This page allows the admin to know where the problem is present and rectify it.



Snapshot 8.13: Welcome page

This snapshot shows the first screen which appears on opening the Campus Bridge app in an Android smartphone. This page provides the options of login or register to the student or faculty.



Snapshot 8.14: Registration Functionality

This snapshot shows the main registration page on installing the app and for usage of the app. This functionality allows for the separate registration as a student or as a faculty.

A screenshot of a Samsung smartphone displaying the CampusBridge app. The app's header is green with the text "CampusBridge". Below the header, the text "STUDENT REGISTRATION" is centered. The form contains the following fields: "First Name" and "Last Name" (separate input boxes), "USN", "CSE" (with a dropdown arrow), "DOB" (with the value "17/4/2018"), "MALE" (selected with a radio button) and "FEMALE" (unselected with a radio button), "Phone", "Sem" (with a dropdown arrow), "Email", "Address", "Password", and "Confirm Password". At the bottom of the form is a green button with white text: "SUBMIT". The phone's status bar at the top shows the time as 16:49, signal strength, and battery level at 46%.

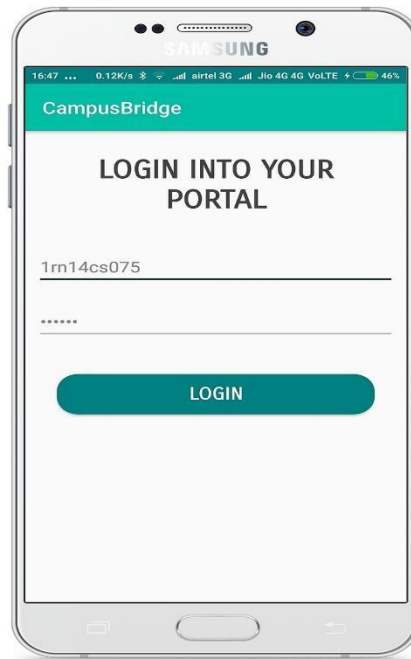
Snapshot 8.15: Student Registration Form

This snapshot shows the student registration form page which needs to be filled out by the student so as to register himself in order to login to his portal in the app. This form collects all the required details from the student to match it with the internal records of the same student, after which the access will be provided.

The image shows a Samsung smartphone screen with the 'CampusBridge' app interface. The title 'FACULTY REGISTRATION' is centered at the top of the form. Below the title, there are input fields for 'First Name' and 'Last Name'. This is followed by a 'USN' field. Then, there's a dropdown menu for 'B.E/B.Tech', a 'DOB' field with the value '17/4/2018', and radio buttons for 'MALE' (selected) and 'FEMALE'. Next is a 'CSE' dropdown menu. Below these are fields for 'Phone', 'Email', 'Address', 'Password', and 'Confirm Password'. At the bottom of the form is a large green button labeled 'SUBMIT'.

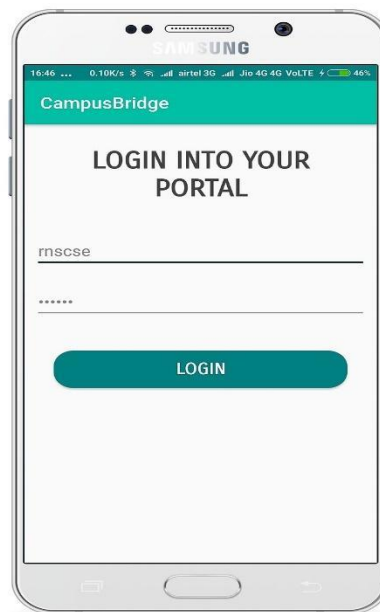
Snapshot 8.16: Faculty Registration Form

This snapshot shows the faculty registration form page that has the same functionality as above but provided for the registration of faculty.



Snapshot 8.17: Student Login Functionality

This snapshot shows the student login page in the app where the credentials given during registration phase needs to be given. This provides security from unauthorized access to student-related information. On successful Login, it redirects to the student portal where the various functionalities of the app are available.



Snapshot 8.18: Faculty Login Functionality

This snapshot shows the faculty login page in the app that is very similar to the student login page except that it is for the faculty.



Snapshot 8.19: Student Home Portal

This snapshot shows the various options available in the student portal of the app. Some of the functionalities include checking attendance for each subject, internal marks of each subject, downloading various notes uploaded by faculties, placement details of companies the student is eligible for, etc.



Snapshot 8.20: Faculty Home Portal

This snapshot shows the various options available in the faculty home portal. Some of these functionalities include attendance confirmation, marks updation for each student and notes uploading for the subject handled, etc.

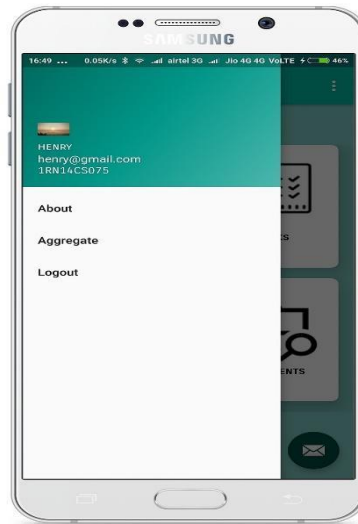
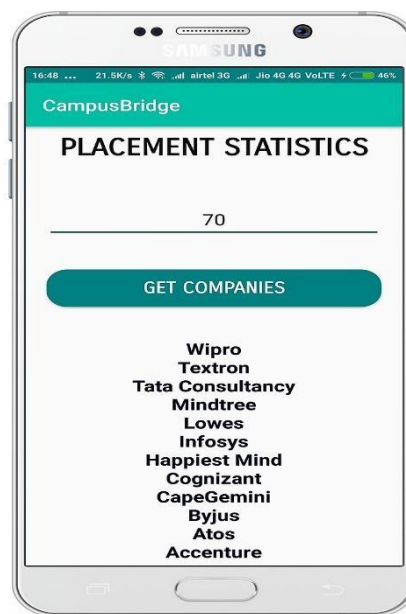


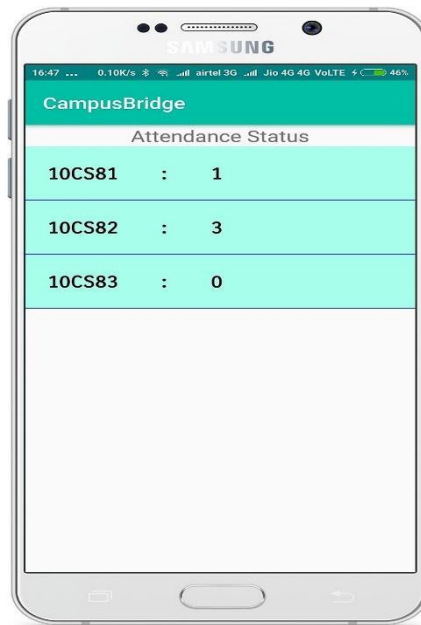
Figure 8.21: Student Portal Side Menu View

This snapshot shows the drawer menu available on clicking in the hamburger icon in the login home portal. This drawer shows the name of the registered student, USN as well as the email-id. It also includes the information such as aggregate calculation, Logout option etc.




Snapshot 8.22: Student Placement Eligibility List

This snapshot shows the various companies to which the student is eligible to apply for using the data from the placement board in the college. This makes the applying to companies and placement department activities much easier and less-time consuming.



Snapshot 8.23: Student Attendance Status

This snapshot shows the class attended per subject by the logged in student in the app. Each time taking class attendance is over, and then the corresponding attendance values are refreshed at the end of the period. This also provides a cross verification which ultimately leads to better efficiency in the system.



MARKS			
10CS81	:	25	0 0
10CS82	:	0	0 0
10CS83	:	0	0 0

Snapshot 8.24: Student Internal Marks details

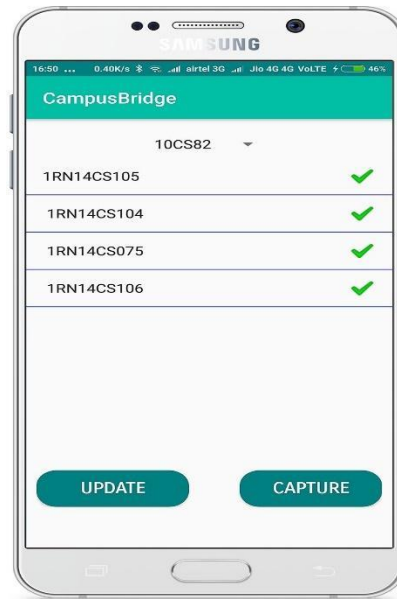
This snapshot shows the various internal marks scored by the student in various subjects in the particular semester. This shows the order of the marks according to the internal test. Most of the marks are updated by the faculty for the particular student USN.



SEM 8
1507.08445.pdf
CS591.pdf
education-08-00026.pdf
Soorya -Front Page and Certificate.pdf

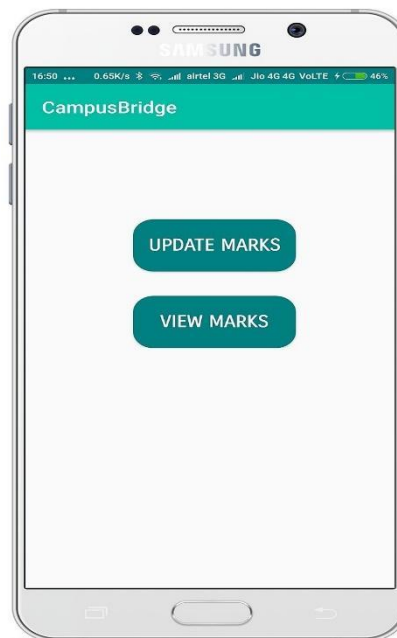
Snapshot 8.25: Student Notes Access Page

This snapshot shows the various notes uploaded by various faculties that can be downloaded by the student based on his requirements. These notes are organized and made available for each student based on his current semester retrieved from the registration details.



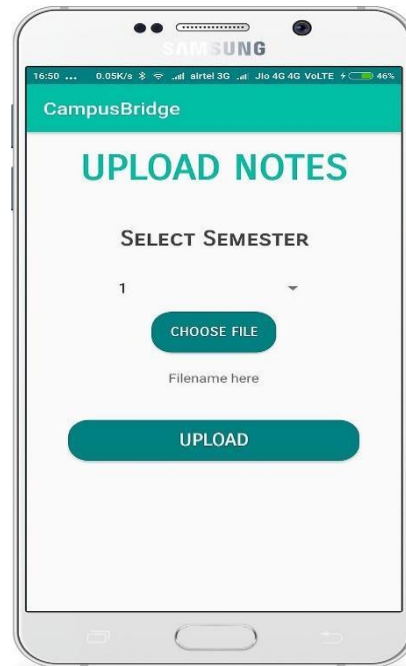
Snapshot 8.26: Attendance Confirmation Page

This snapshot shows the Attendance Confirmation Page that is a part of the attendance functionality in the Faculty app. This page allows for the faculty updates opening the camera app required to capture the photo of the class for image processing and according to the tally between RFID status and this count, the attendance.



Snapshot 8.27: Faculty Internal Marks Updating

This snapshot shows the faculty app's internal marks updating page that provides them with the functionality of updating the marks scored by a student directly under his USN for that subject for any of the internals. These updated marks will be applied in the database and can be viewed from the marks section of the student app.



Snapshot 8.28: Notes Upload Functionality

This snapshot shows the notes upload functionality page available in the faculty app that provides the faculty a window to upload notes that are then segregated according to their mentioned semesters.

Chapter 9

CONCLUSION

As we are running towards a much smarter and technology driven world, where manual work has become a thing of the past, our proposed Campus Bridge project is definitely going to be a step in the right direction. It will not just provide efficient solutions to various day-to-day functionalities carried out in an educational institution, but also gives a scope to more technology driven approach to our educational system. Tablets will replace all the books, RFID tags will replace all ID cards, as digitalization takes over the way in which our colleges and institutions run. This project is not just about the end of the traditional ways of college working but more about the dawn of a new, modern and technology-driven approach to bridge the gap between students and campus.

Our project aims mainly at eliminating the current traditional ways of student and faculty record management and remote access to these records for both faculty and students. This project builds this efficient system by utilizing the power of RFID and IoT. We have RFID reader reads the attendance and stores it in database, this data can be accessed through the student's app. The attendance aggregate is calculated automatically and stored into the database. Faculty can upload notes; insert marks which helps in automation and maintaining the data. Student can access marks and download notes anytime and from anywhere. There is also an admin who is responsible for addition, deletion and updating of faculties, students and placement details in the database that later could be gathered through the mobile app. He is also responsible for sending push notifications to the mobile apps regarding any important news, information or notices to be conveyed to all students and faculties from the college side.

In future, we can enhance our project by replacing the RFID USB reader which always needs to be connected with a system with a Wireless enabled RFID reader which is more efficient and makes the project more flexible and portable. Another scope of improvement can be done in the field of image processing by using a much more efficient method to detect faces even in low light shots. The current project also can be expanded in various other functionalities such as the library department, even hostel and parking allotment and management to monitor and regulate these in much more efficient way. We are also working upon bringing up our apps in various other platforms such as Windows, IOS, Firefox OS, etc. to enable a much wider reach to the project.

BIBLIOGRAPHY

- [1] Manual Required for Java Servlet and database handling
URL: <https://www.javatpoint.com/>
- [2] Firebase Cloud Messaging Service Manual URL: <https://firebase.google.com/>
- [3] Herve Chabanne, Pascal Urien, Jean-Ferdinand Susini., RFID and the Internet of Things, Wiley Publications, pages 57-218, New York, Mar 2013.
- [4] Prof. Sagar Rajebhosale, Mr. Shashank Choudhari, Mr. Sachin Patil, Mr. Akshay Vyavahare, Mr. Sanket Khabiya., SMART CAMPUS – An Academic Web Portal with Android Application., Volume 3, Issue 4, pages 389–394, Apr. 2016.
- [5] Lalit Mohan Joshi., “A Research Paper on College Management System. International Journal of Computer Applications ”, (0975 – 8887) Volume 122 – No.11, July 2015.
- [6] Chitresh, S and Amit K, ”An efficient Automatic Attendance Using Fingerprint Verification Technique”, International Journal on Computer Science and Engineering (IJCSE), Vol. 2 No. 2, pp 264-269. Dec 2010.
- [7] M.R.M.Veeramanickam, Dr. M. Mohanapriya., IOT enabled Futurus Smart Campus with effective E-Learning : I-Campus, GSTF Journal of Engineering Technology (JET) Vol.3 No.4, , pages 81-87, Apr. 2016.
- [8] C. E. Geoffrey, “Automatic Access Control System using Student’s Identification Card based on RFID Technology”, Unpublished Thesis Faculty of Electrical Engineering, University of Teknologi Malaysia, Jan 2012.
- [9] Pranay Kujur and Kiran Gautam, "Smart Interaction of Object on Internet of Things", International Journal of Computer Sciences and Engineering, Volume-03, Issue-02, Page No (15-19), Feb -2015.

- [10] Intelligent Device-to-Device Communication in the Internet of Things Oladayo Bello, Member, IEEE, and Sherali Zeadally, Senior Member, IEEE, Dec 2013.