



**DIRECTORATE OF TECHNICAL EDUCATION, BENGALURU**

#### JANATA SHIKSHANA SAMITI’S

**K H KABBUR INSTITUTE OF ENGINEERING**

##### VIDYAGIRI, DHARWAD 580004

PROJECT REPORT ON

#### INVOICE MANAGEMENT SYSTEM

**SUBMITTED IN PARTIAL FULFILLMENT OF REQUIREMENTS OF THE**

**DIPLOMA IN**

**COMPUTER SCIENCE & ENGINEERING**

FOR THE ACADEMIC YEAR 2021-22

**UNDER THE GUIDANCE OF Smt. V. R. TIWARI**

**SUBMITTED BY**

##### ABHAY G JOSHI 202CS19001

**JINENDRA B KOTI 202CS19014**

##### JUNAID A SOUDAGAR 202CS19015 KIRAN P CHAVAN 202CS19017 MAHALAXMI P DESHAD 202CS19019

##### NAYANA KUNDAGOL 202CS19023





**A PROJECT REPORT ON**

#### INVOICE MANAGEMENT SYSTEM

**SUBMITTED FOR PARTIAL FULLFILLMENT FOR THE AWARD OF THE OF**

**DIPLOMA IN COMPUTER SCIENCE AND ENGINEERING**

**BY BATCH**

##### ABHAY G JOSHI 202CS19001

**JINENDRA B KOTI 202CS19014**

##### JUNAID A SOUDAGAR 202CS19015 KIRAN P CHAVAN 202CS19017 MAHALAXMI P DESHAD 202CS19019

##### NAYANA KUNDAGOL 202CS19023

**UNDER THE GUIDANCE OF**

##### Smt. V. R. TIWARI HEAD OF DEPARTMENT

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING





##### DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING K H KABBUR INSTITUTE OF ENGINEERING

**VIDYAGIRI, DHARWAD 580004.**

**CANDIDATE DECLARATION**

##### I, ABHAY G JOSHI a

student of **Diploma in Computer Science and Engineering Department** bearing Reg No **202CS18047** of KHKIE polytechnic, hereby declare that I own full responsible for the information, results and conclusions provided in project work titled **“INVOICE MANAGEMENT SYSTEM”** submitted to **State Board Of Technical Examinations, Government of Karnataka** for the award of ***Diploma in Computer Science and Engineering.*** To the best of my knowledge, this project work has not been submitted in part or full elsewhere in any other institution/organization for the award of any certificate/diploma/degree. I have completely taken care in acknowledging the contribution of others in this academic work. I further declare that in case of any violation of intellectual property rights and particulars declared, found at any stage. I, as the candidate will be solely responsible for the same.

##### DATE:

**PLACE:**

**SIGNATURE OF THE CANDIDATE**





## DEPARTMENT OF TECHINICAL EDUCATION BENGALURU-560001



##### K H KABBUR INSTITUTE OF ENGINEERING VIDYAGIRI, DHARWAD 580004.

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING BONAFIDE CERTIFICATE**

Certified that this project report **“INVOICE MANAGEMENT SYSTEM”** is the bonafide work of **ABHAY G JOSHI** bearing Register No **202CS19001** of this institution who carried out the project work under my supervision.

##### SIGNATURE SIGNATURE

**Smt. V. R. TIWARI Smt. V. R. TIWARI**

***GUIDE, Head Of Department,***

Computer Science & Engineering, Computer Science & Engineering,

K H Kabbur Institute of Engineering, K H Kabbur Institute of engineering

Vidyagiri,Dharwad-580004 Vidyagiri,Dharwad-580004.





# DEPARTMENT OF TECHNICAL EDUCATION, BENGALURU 580001

#### K H KABBUR INSTITUTE OF ENGINEERING VIDYAGIRI, DHARWAD-580004



**CERTIFICATE**

Certified that this report entitled **“INVOICE MANAGEMENT SYSTEM”,** which is being submitted by **ABHAY G JOSHI**, Reg. No **202CS18047,** award of **Diploma in Computer Science and Engineering** during the year **2020-21** is record of students own work carried out under my/our guidance. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report and one copy of it being deposited in the polytechnic library.

The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said diploma. It is further understood that by this certificate the undersigned do not endorse or approve any statement made, opinion expressed or conclusion drawn there in but approve the project only for the purpose for which it is submitted.

##### Guide Head of Department Principal Name and Signature Examiner

1.

2.



#### ACKNOWLEDGEMENT

The satisfaction that accompanies that the successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success.

We are grateful to our project guide **Smt. V. R. TIWARI** for the guidance, inspiration and constructive suggestions that helped us in the preparation of the project.

We express our deep gratitude to **Principal KHKIE, Dharwad,** for providing the facilities and support.

Our sincere thanks to all teaching and no-teaching staff of **CSE Department KHKIE** for all the facilities provided, without which, we could no progressed with our work. Thanks to our parents and friends who have been a great source of strengths all through this work.

##### ABHAY G JOSHI

**JINENDRA B KOTI**

##### JUNAID A SOUDAGAR

##### KIRAN P CHAVAN

##### MAHALAXMI P DESHAD

##### NAYANA KUNDAGOL



#### ABSTRACT

This project work is devoted to present a biometric identification system based on fingerprint recognition. Biometrics can be taken literally as 'life measurement' but the term is usually associated with the measurement and use of unique physiological characteristics to identify an individual person.

Biometric Identification Systems are widely used for unique identification and verification of humans. At present, there are many types of biometric technology have been used; such as, fingerprint recognition, face recognition, voice recognition, iris recognition, etc. Fingerprint recognition is considered to be the best and fastest method for biometric identification.

The biometric fingerprints features are secure to use, unique for every person and do not change in one's lifetime. The aim of this project is to develop a fingerprint recognition system that can accurately identify the students of the department.

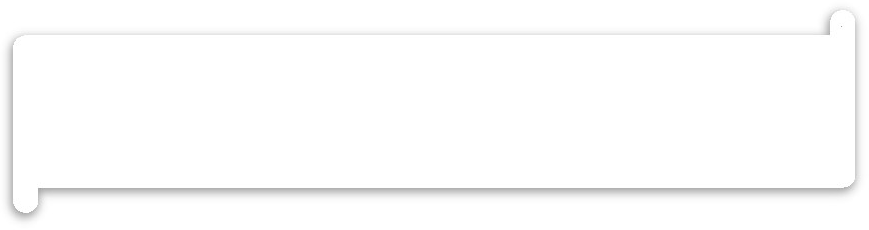


#### INDEX

|  |  |  |
| --- | --- | --- |
| **SL.NO** | **CONTENTS** | **PAGE NO** |
| 1 | **CHAPTER-1** INTRODUCTION | 1 |
| 2 | **CHAPTER-2** OBJECTIVE | 2-4 |
| 3 | **CHAPTER-3** TOOLS/ENVIRONMENT USED   * 1. Hardware requirement   2. Software requirement   3. Introduction to languages | 5-10 |
| 4 | **CHAPTER-4** SYSTEM ANALYSIS   * 1. Introduction   2. System specification   3. Analysis | 11-13 |
| 5 | **CHAPTER-5** SYSTEM DESIGN  5.1Data flow diagram 5.2Entity relation diagram | 14-25 |
| 6 | **CHAPTER-6** PROGRAM CODE | 26-31 |
| 7 | **CHAPTER-7** TESTING   * 1. Unit testing   2. Integration testing   3. System testing | 32-36 |
| 8 | **CHAPTER-8** INPUT AND OUTPUT SCREENS | 37-40 |
| 9 | **CHAPTER-9** LIMITATIONS OF THE PROJECTS | 41 |
| 10 | **CHAPTER-10**  FUTURE SCOPE OF THE PROJECTS | 42 |
| 11 | **CHAPTER-11** CONCLUSION | 43 |
| 12 | **CHAPTER-12**  REFERENCES AND BIBLIOGRAPHY | 44 |



**CHAPTER-1**



**INTRODUCTION**

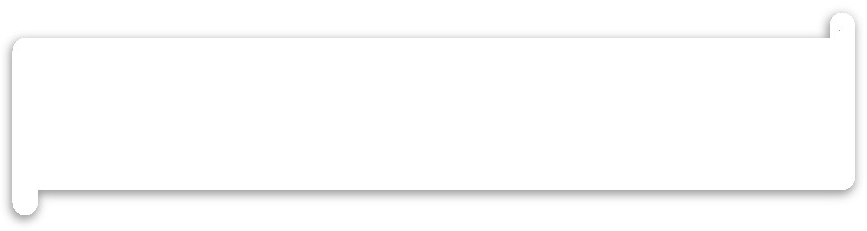
Today the Internet is the most powerful technology in the world. From morning to night, we need the help of technology. This is the revolutionary time of computer technology.

It’s time to move beyond manual data entry and double-handling. All of the aspects of invoice management that slow down large or small businesses can be fixed.82% of finance departments are overwhelmed by the high numbers of invoices they are expected to process on a daily basis and the variety of formats they’re received in.Just like lots of other finance processes, invoice management has officially become a burden.

The trust of this research is to use technology for classifying, managing and generating invoices online for large or small case businesses so that the use of papers for invoice and billing purpose can be reduced, if not eliminated. The major features of the proposed Invoice Management System is to manage products and generate invoices of the same; also create a digital copy of the invoice to send it to the user. Effective use of technology can help reduce usage of papers and money.



**CHAPTER-2**



**OBJECTIVE**

#### The overall objective of this project is to build a web application that helps businesses to easily manage products and generate invoices for the same in a PDF format if required. This aim will involve completing the objective of designing and developing a web application.

#### Scope

* The proposed system is a web-based application that helps businesses to manage and generate invoices of products.
* The owner/admin can login and add products that were sold along with the information of that product.
* The owner/admin can also add customers to whom he wants to generate an invoice.
* The saved invoices are stored in the database and the owner can search, update, delete or generate any invoice at any time.
* The owner/admin can generate a PDF of any particular invoice with all the details of the invoice included in the PDF automatically.
* Once the bill is paid by the user, the owner/admin can tick the invoice as “paid”.

#### Aim of the Project

To develop a web application for managing the invoices of products in a small- or large-scale business.

#### Present System

The present system is complex and mostly uses paper for invoices. Most invoices are generated using paper and after the work is over, that paper is of no use. Also, it’s hard to keep track of the invoices manually and requires a lot of human labour for searching some particular invoice. It’s also very time consuming and requires money to print invoices. The invoices are also lost in some cases which is a great loss. There are a lot of invoice fraud cases as well.

#### Proposed System

* Invoice Management System is the best solution for small- and large-scale businesses.
* This system helps sellers reduce the money spent on producing invoices manually using paper.
* Instead of issuing a physical invoice copy, we can generate a PDF of an invoice with the product information and cost, etc., and then share it with the customer.
* This system contains a database of Invoice information which includes the product details and other attributes.
* Strain of manual labor can be reduced; Risk of invoice fraud is also reduced.
* Keeps track of products sold.

#### Input of the Project

* Admin credentials for login.
* Products information.
* Customer Information.
* Unit cost, quantity of products.
* Invoice type like Receipt, Invoice.
* Status like, if bill was paid or not.

#### Output of the project

* Admin can see the added invoices of various products, products or customers.
* Admin can search for any particular invoice, product or customer.
* Admin can generate a PDF of an added invoice automatically.

#### Modules included

* Admin Login module.
* Invoice module.
* Customer module.
* Inventory module.

**Admin Login Module:**

* To perform any actions in the web-application, the admin has to provide their credentials and must be logged in.
* If the credentials are incorrect the web-app won’t allow the admin to enter into the app and displays a notification message as “Incorrect username/password.”
* If the credentials are correct the web-app will immediately log-in to the website and allows the admin to perform any actions.

**Invoice Module:**

* This module shows all the options like, creating new invoices, etc...
* A web page with options about adding invoice number, invoice date, name of the products, quantity of the products, the unit price of the products, name of the customer, phone number of the customer is displayed.
* The total of each item as well as the grand total is computed automatically by the system.
* There is a checkbox called “Paid”, which can be ticked if the bill is paid.
* Finally, there is an option to save the invoice in the database.

##### Customer Module:

* This module helps the owner to make changes to the invoices like, adding, deleting, updating items.
* This module also allows the owner to search for a particular invoice with the help of the invoice number or grand total or customer name as input.
* There is also a checkbox called as “Generate”, which if checked and then pressed the button “Generate”, downloads a PDF document which has the invoice along with a section for the seller’s signature.

##### Inventory module:

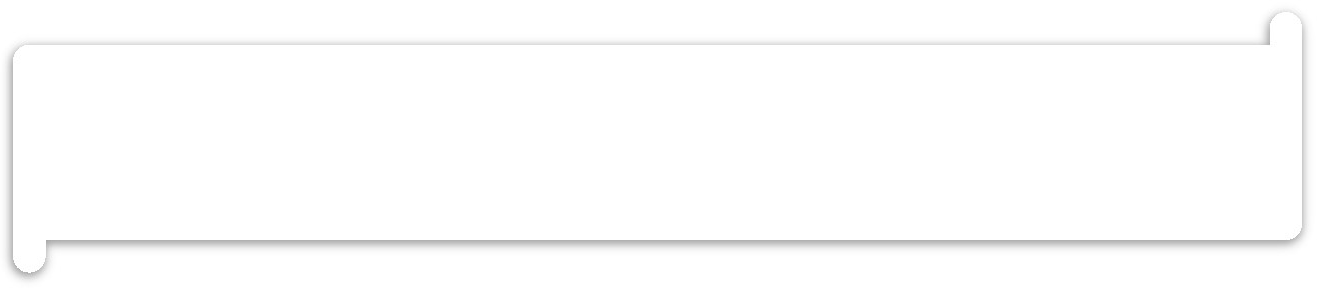
* In this module, the Admin can view all the registered students.
* The student details which are registered are fetched from the Firebase Realtime Database and are displayed onto the screen in the list format.

**Advantages**

* Better time management.
* No or few paper works.
* Supports in making decision.
* Reduces human labour.
* Easiness of modification of data.
* Reduced risk of invoice fraud.



**CHAPTER-3**



**TOOLS/ENVIRONMENT USED**

# Hardware Requirements

* Processor: Dual Core
* RAM: 2GB
* Hard Disk: 100GB

# Software Requirements

* **IDE –**Atom
* **Front end –** Bootstrap, HTML, CSS, jQuery
* **Logics and operations –** Python
* **Database –**MySQL
* **Authentication –**User Authentication in Django
* **Frameworks –** Django
* **Libraries –**Crispy-forms
  1. **Introduction to languages**

#### Python –

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small- and large-scale projects.

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It’s free and open source.

Some of the important core features of Python are:

* Python is easy to learn.
* Python is a Dynamically-Typed Language.
* It has a large standard library and Frameworks.
* It is platform independent.

#### Django –

Django is a high-level Python web framework that enables rapid development of secure and maintainable websites.

1. Excellent Documentation: If we compare Django with other open-source technologies, it offers the best documentation in the market. Better documentation of any technology is like a very well-established library for any developer. There, he can search for any function desired with ease with the time involving in the searching purpose only.
2. Versatile in Nature: The **logical project structure** and MVT architecture of Django sometimes seem very limiting. But, that’s just opposite because by giving us the files it is providing us with a solid foundation which can then be used to make whichever application we want to create.
3. Offers high security: Django is super secure. To prove the feature, you can always take examples of lots of websites which are worldwide and possess huge traffic. Django is secure because it covers the loopholes by default which were once left open for the backend developer to complete. Although while using Django you may not feel it but those expert backend developers can tell the quality and security of the work done by Django.

**Models in Django –**

A model is the single, definitive source of information about your data. It contains the essential fields and behaviours of the data you’re storing. Generally, each model maps to a single database table.

The basics:

* Each model is a Python class that subclasses **django.db.models.Model.**
* Each attribute of the model represents a database field.
* With all of this, Django gives you an automatically-generated database-access API.

This example model defines a **Person**, which has a **first\_name** and **last\_name**:

**fromdjango.dbimport** models

**classPerson**(models.Model):

first\_name = models.CharField(max\_length=30)

last\_name = models.CharField(max\_length=30)

**first\_name** and **last\_name** are fields of the model. Each field is specified as a class attribute, and each attribute maps to a database column.

The above **Person** model would create a database table like this:

**CREATETABLE**myapp\_person (

"id" serial **NOTNULLPRIMARYKEY**,

"first\_name" varchar(30) **NOTNULL**,

"last\_name" varchar(30) **NOTNULL**

);

A **ViewGroup** is a subclass of View, and is essentially an 'invisible container' that holds multiple Views or ViewGroups together, and defines their layout properties.

**Views in Django –**

A view function, or view for short, is a Python function that takes a web request and returns a web response. This response can be the HTML contents of a web page, or a redirect, or a 404 error, or an XML document, or an image . . . or anything, really. The view itself contains whatever arbitrary logic is necessary to return that response. This code can live anywhere you want, as long as it’s on your Python path. There’s no other requirement–no “magic,” so to speak. For the sake of putting the code somewhere, the convention is to put views in a file called **views.py**, placed in your project or application directory.

**A simple view**[**¶**](https://docs.djangoproject.com/en/4.0/topics/http/views/#a-simple-view)

Here’s a view that returns the current date and time, as an HTML document:

**fromdjango.httpimport**HttpResponse

**importdatetime**

**def**current\_datetime(request):

now = datetime.datetime.now()

html = "<html><body>It is now **%s**.</body></html>" % now

**return**HttpResponse(html)

**URL dispatcher in Django –**

A clean, elegant URL scheme is an important detail in a high-quality web application. Django lets you design URLs however you want, with no framework limitations.

To design URLs for an app, you create a Python module informally called a **URLconf** (URL configuration). This module is pure Python code and is a mapping between URL path expressions to Python functions (your views).

This mapping can be as short or as long as needed. It can reference other mappings. And, because it’s pure Python code, it can be constructed dynamically.

Example

Here’s a sample URLconf:

**fromdjango.urlsimport** path

**from.import** views

urlpatterns = [

path('articles/2003/', views.special\_case\_2003),

]

#### Constraint Layouts

Newer versions of Android Studio may create default layouts called ConstraintLayout instead of LinearLayouts or RealtiveLayouts. ConstraintLayouts are a newer layout format that is more powerful, but also more complex for beginners to work with.

#### Advantages of using ConstraintLayout in Android

* + - ConstraintLayout provides you the ability to completely design your UI with the drag and drop feature provided by the Android Studio design editor.
    - It helps to improve the UI performance over other layouts.
    - With the help of ConstraintLayout, we can control the group of widgets through a single line of code.
    - With the help of ConstraintLayout, we can easily add animations to the UI components which we used in our app.

#### Disadvantages of using ConstraintLayout

* + - When we use the Constraint Layout in our app, the XML code generated becomes a bit difficult to understand.
    - In most of the cases, the result obtain will not be the same as we got to see in the design editor.
    - Sometimes we have to create a separate layout file for handling the UI for the landscape mode.

#### Firebase Realtime Database -

Store and sync data with our NoSQL cloud database. Data is synced across all clients in realtime, and remains available when your app goes offline.

The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in realtime to every connected client. When you build cross-platform apps with our iOS, Android, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data.

Most databases require you to make HTTP calls to get and sync your data. Most databases give you data only when you ask for it.

When you connect your app to Firebase, you’re not connecting through normal HTTP. You’re connecting through a WebSocket. WebSockets are much, much faster than HTTP. You don’t have to make individual WebSocket calls, because one socket connection is plenty. All of your data syncs automagically through that single WebSocket as fast as your client’s network can carry it.

Firebase sends you new data as soon as it’s updated. When your client saves a change to the data, all connected clients receive the updated data almost instantly.

#### Firebase Authentication -

Most apps need to know the identity of a user. Knowing a user's identity allows an app to securely save user data in the cloud and provide the same personalized experience across all of the user's devices.

Firebase Authentication provides backend services, easy-to-use SDKs, and ready- made UI libraries to authenticate users to your app. It supports authentication using passwords, phone numbers, popular federated identity providers like Google, Facebook and Twitter, and more.

Firebase Authentication integrates tightly with other Firebase services, and it leverages industry standards like OAuth 2.0 and OpenID Connect, so it can be easily integrated with your custom backend.

#### Firebase Cloud Storage –

Cloud Storage for Firebase is built for app developers who need to store and serve user- generated content, such as photos or videos.

Cloud Storage for Firebase is a powerful, simple, and cost-effective object storage service built for Google scale. The Firebase SDKs for Cloud Storage add Google security to file uploads and downloads for your Firebase apps, regardless of network quality.

You can use Firebase SDKs to store images, audio, video, or other user-generated content. On the server, you can use Google Cloud Storage APIs to access the same files.

#### Glide Library –

Glide is a fast and efficient open source media management and image loading framework for Android that wraps media decoding, memory and disk caching, and resource pooling into a simple and easy to use interface.

Glide supports fetching, decoding, and displaying video stills, images, and animated GIFs. Glide includes a flexible API that allows developers to plug in to almost any network stack. By default Glide uses a custom HttpUrlConnection based stack, but also includes utility libraries plug in to Google's Volley project or Square's OkHttp library instead.

Glide's primary focus is on making scrolling any kind of a list of images as smooth and fast as possible, but Glide is also effective for almost any case where you need to fetch, resize, and display a remote image.

#### Android Studio

Android Studio is the official integrated development environment (IDE) for Android application development. It is based on the IntelliJ IDEA, a Java integrated development environment for software, and incorporates its code editing and developer tools. To support application development within the Android operating system, Android Studio uses a Gradle- based build system, emulator, code templates, and Github integration. Every project in Android Studio has one or more modalities with source code and resource files. These modalities include Android app modules, Library modules, and Google App Engine modules. Android Studio uses an Instant Push feature to push code and resource changes to a running application. A code editor assists the developer with writing code and offering code completion, refraction, and analysis. Applications built in Android Studio are then compiled into the APK format for submission to the Google Play Store.

In android operating system, there are four layers. Android has its own libraries; it is helpful for developing and designing any application of android platform. These libraries are written in C/C++. Linux kernel is the 1st layer which is written in C. Linux also helps to wrap the application. The unveiling of the Android platform on 5 November 2007 was announced with the founding of the Open Handset Alliance, a consortium of 34 hardware, software and telecom companies devoted to advancing open standards for mobile devices.

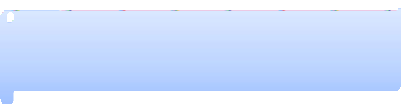
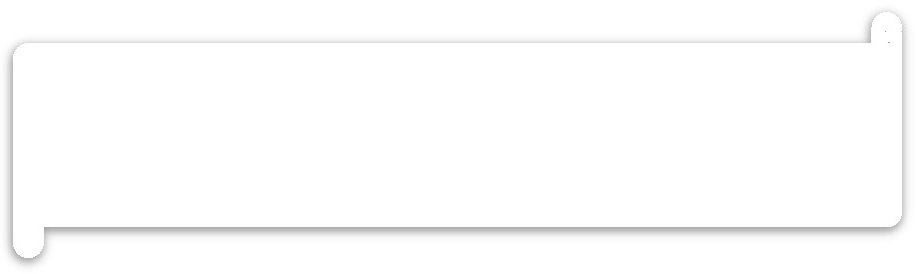
When released in 2008, most of the Android platform will be made available under the Apache free-software and open source license. Currently Android represents 31.2 percent of the U.S Smartphone market. Android has a large community of developers writing application programs. There are currently over 150,000 apps available for Android. Android Market is the online app store run by Google, though apps can also be downloaded from third party sites.

#### Features

* + - Support for a fast emulator
    - Support for Gradle
    - Support for plenty of code templates and GitHub integration
    - Support for Google Cloud Platform
    - Support for template-based wizards for creating Android designs and components
    - Support for rich layout editor
    - Support for deep code analysis
    - Support for extensive set of tools and frameworks

##### Mantra mfs100 –

Mantra MFS100 is a optical fingerprint sensor. MFS100 is high quality USB fingerprint sensor for fingerprint authentication in desktop, android or network security. MFS100 is based on optical sensing technology which efficiently recognizes poor quality fingerprints also. MFS100 can be used for authentication, identification and verification functions that let your fingerprint act like digital passwords that cannot be lost, forgotten or stolen. Hard optical sensor is resistant to scratches, impact, vibration and electrostatic shock.



**CHAPTER-4**

**SYSTEM ANALYSIS**

#### INTRODUCTION

A software requirements specification (SRS) is a description of a software system to be developed. It is modelled after business requirements specification (CONOPS), also known as a stakeholder requirements specification (SRS). The software requirements specification lays out functional and non-functional requirements, and it may include a set of use cases that describe user interactions that the software must provide to the user for perfect interaction.

Software requirements specification establishes the basis for an agreement between customers and contractors or suppliers on how the software product should function (in a market-driven project, these roles may be played by the marketing and development divisions). Software requirements specification is a rigorous assessment of requirements before the more specific system design stages, and its goal is to reduce later redesign. It should also provide a realistic basis for estimating product costs, risks, and schedules. Used appropriately, software requirements specifications can help prevent software project failure.

##### PURPOSE:

A software requirements specification (SRS) is a comprehensive description of the intended purpose and environment for software under development An SRS minimizes the

time and effort required by developers to achieve desired goals and also minimizes the development cost.

##### SCOPE:

The scope of this document lies between developer of the application and endUser.

The document can also for future enhancements.

##### SYSTEM SPECIFICATION:

System specification is the phase which partitions the requirements into hardware and, software and performance requirements. It also specifies the details of implementation tools. A system specification describe the operational and performance requirements of a system, such as a computer. It is considered a high-level document that dictates global functions.

System specifications help to define the operational and performance guidelines for a system. It may outline how the system is expected to perform, and what that may include. Key specifications may include interface definitions, document design rules and functional areas. When purchasing software or a computer, system specifications may be outlined during the evaluation process and agreed upon during the payment process. The specifications may determine security access. Many organizations will offer templates and resources to help facilitate the adherence of system specifications. In some cases, system specifications can be quite specific and difficult to follow without these guidelines.

#### ANALYSIS

##### Requirement specification

At present, students are given the physical hall tickets for attending the examinations. Which is paper print. The Invigilator marks the attendance on that paper. Cheating may happen sometimes. In this project we are going to make a virtual hall ticket using an android application.

##### Reviewing the analysis document

We have analysed the problem of cheating during exam and the effort in the making the hall ticket manually. The student will be given a physical hall ticket which they will lose or forget to bring it during the examination. In this application the registration and verifying will be done. And the cheating and the effort will be reduced.

##### Designing the interface

We have designed the modules interfaces using android studio with widgets and XML tags. That will be user friendly in future user interface enhancements.

##### Implementing the idea:

According to the requirements we have done app in android by using requirement code such as the admin will be given the students information.

The admin will do the registration on FingerPrintHallticket Application, by using Android Studio we have coded for application and designed it using XMl and Widgets and Firebase Realtime Database as its Database.

##### Validation:

The project has been tested all the modules with different values of testing stages and rectified some faults in the code of java.

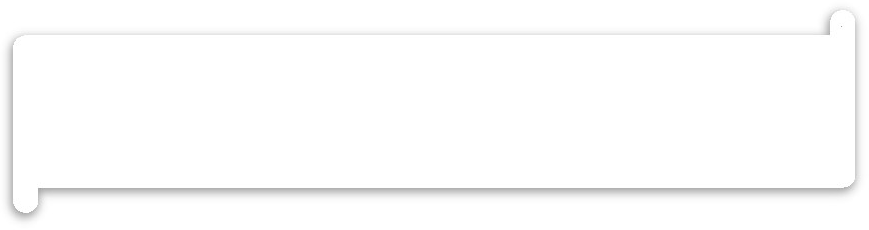
##### Registration:

This app is private based privacy concerned app. The personal data won’t be accessed to others. Only the admin has permissions to access it.

**Admin**: Admin is the main in charge of this whole project. He does the Registration and Verification.



**CHAPTER-5**



**SYSTEM DESIGN**

# Introduction

0poSystem design is modelling process. It can be as transaction from user’s view to programmer view. The term design used in two ways. Used as noun; it represents the resulted in design process. The term designer’s goal is to produce a module or representation of an entity, which can be used to build a system.

“Software Design sits at technical kernel software engineering process and is applied regardless of the development paradigm that is used”-Roger .S. Spressman says .Design is essentially a bridge between the requirement specification and Final implementation.

For satisfying requirements. The design process of software has two levels. At the first level, the focusing is on deciding which module should be interconnected. This is called System design”, or Top level design”. In the second level, the internal design of the model or how specification of the Modules can be satisfied is decided. The design level is often called Detailed Design or Logical design”. The design of the system is print plan of the system.

# System design

Here the focus is a deciding which modules area needed for the system, the specifications of these modules and how it should be interconnected.

Data design: It describe structure that resides within the system.

Architecture Design: It uses information flow characteristics and map into the program me structure. Transformation mapping method is used to exhibit distinct boundaries between incoming

The purpose of the design phase is to plan a solution of the problems specified by the requirement document .this first phase step in moving problem domain to solution domain. It involves a process in which conceiving planning and carrying out by generating the necessary reports and printouts. The design of the system is perhaps the most critical

Factor of affecting the quality software, and has major impact on the later phases, particularly testing maintenance. The output of this phase is the design document.

#### Logical Design

Understanding software document requires the logical structure analysis of diagrammatic notations such as data flow diagrams .the logical design describes the detailed specifications for the system it should include the following:

* External system structure
* Relationship between all the elements
* Physical construction activity
* Global data
* Control flow
* Derived program structure.

#### Data Flow Diagrams

A DFD usually comprises of four components. These four components can be represented by four simple symbols. These symbols can be explained in detail as follows: External entities(source/destination of data) are represented by squares; Processes(input- processing-output) are represented by rectangles with rounded corners; Data Flows(Physical or electronic –output ) are represented by arrows; and finally, Data Stores (Physical or electronic like XML files) are represented by open-ended rectangles.

Data store

A data store stores data passively for later access. A Data store responds to requests to store and access data. It does not generate any operations.

A data store allows values to be accessed in order different from the order in which they were generated.

Input flows indicate information or operations that modify the stored data such as adding or deleting elements or changing values. Output flows indicate information retrieved form the store; this information can be entire value or a component of a value.

Data Flow:

A data flow moves data between processes or between processes and data stores. As such it represent a data value at some point within a computation and an intermediate value within a computation if the flow is internal to the diagram. This value is not changed. The names of input and output flows can indicate their roles in the computation or the type of the value they move. Data names are preferably nouns. The name of a typical piece of data, the data aspect , is written alongside the arrow.

Data Process

A process show a transformation or manipulation of data flows within the system . The symbol used is rounded rectangle or oval. A descriptive title is placed in the centre of the circle. This should be a simple imperative sentence with a specific with a specific verb, for example ‘maintain customer records’ or ‘find driver’. E.g: May be a clerk computing discounts or a combination of manual and electronic activities.

Source of Sink

An external entity is a source or destination of a data flow which is outside the area of study . The symbol used is a rectangle or square containing a meaningful and unique identifier.

Only those entities which originate or receive data are represented on a business process diagram. E.g: May be one customer or a number of customer with transactions (Orders).

**Rules for Creating DFD**

* Processes should be named and numbered for easy reference. Each name should be representative of the process.
* The direction of flow top to bottom and from left to right, Data traditionally flow from the source(upper left corner) to the destination although they may flow back to a source. One way to indicate this is to repeat the source symbol as a destination. Since it is used more than once in the DFD, it is marked with a short diagonal in the right corner.
* When a process is exploded into the lower-level details, they are numbered. The names od data stores ,sources and destinations are written in capital letters. Process and data flow names have the first letter of each word capitalized

**Data Flow Diagram –**

Store it in

database

Capture the

fingerprint

Registration

Fingerprint

based hall ticket

Verification

Register the

student

Print the

result

Print the registered

students

Request to

view

Registration

**Registration Module DFD -**

Admin

Registration

process

Student and his

information

Enter username

and password

Registers

Information

of the student

## Registration Module DFD -

Information of the student

Admin

Takes the student reg no.

and fingerprint Verifies

Verification process

Print the results.

Fetches the student’s info

Mark the subject as attended

Mark the Attendance.

## View Registered Students Module DFD -

Information of the student

Admin

Requests to view all students

View Registered STudents process

Prints

Print the results.

Fetches the student’s info

## ENTITY-RELATIONSHIP DIAGRAM

An entity–relationship model, a graphical representation of entities and their relationship to each other, typically used in computing regard to the organization of data within databases or information systems.

An entity is a piece of data-an object or concept about which the data is stored. A relationship between entities.

* + 1. ONE TO ONE: one instance of an entity(A) is associated with the other entity

(b) for example employee name is associated with only one security number.

* + 1. ONE TO MANY: one instance of an entity(A) is associated with zero, one or many instances of another entity(b), but for one instance of an entity B there is only one instance of an entity .A for example, for a company with all employees working in one building , the building name(A) is associated with many different employees(b) but those employee all share with same entity with A.
    2. MANY To MANY: one instance of an entity (A) is associated with one, zero or many instance of an entity (B) and one instance of an entity B is associated with one zero work on multiple employee (A) associated with it.

The entity is a person , object , place or event for which data is collected for example if you consider the information system for a business , entities , would include not only customers, but the customers but the customers address , and order as well. The entity is represented by a rectangle and single labelled with a singular noun.

The relationship is the interaction between the entities would include only to customers

, but the customers address , and well orders .The entity is represented by a diamond shape , by the line connecting the entities in either case verb used about basic relationships.

The cardinality defines the relationship between the entities the relationships. The cardinality defines the relationship between the entities in terms of numbers . An entity may be optional: for example, there must be at least one production.

Name Reg No.

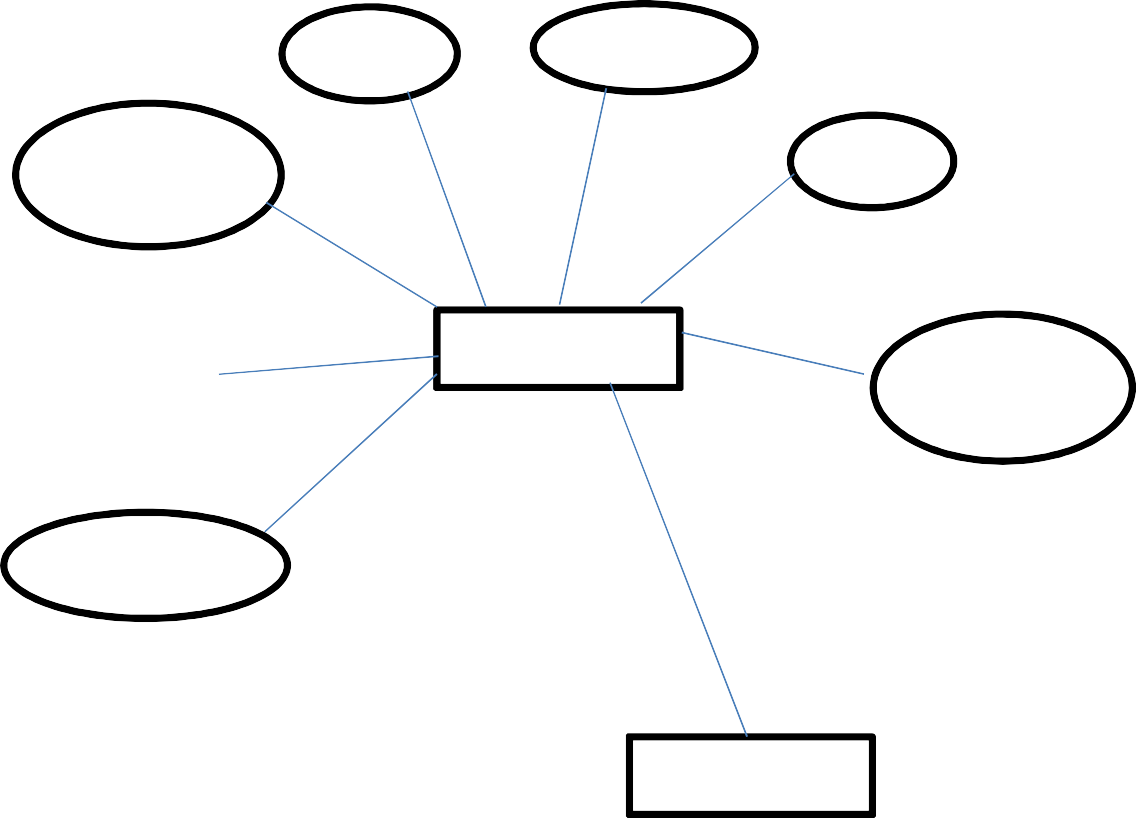
Fingerprint Template

Sem

Image URL

### Students

Father’s Name



Department

### Subjects

Name

QPCode

Reg No.

Status

Serial no

Date/Time

#### Introduction to database

A database is a collection of related information defining a database involves the data type, attributes and constraints for the data to be stored. Constructing a database is a process of storing itself on some storage medium like disk or tape that can be provided by DBMS. Manipulating a database includes such as functions like querying a database to specific data, updating the database to reflect changes in the mini world and operating report from the data. A DBMS represents complex relationship with data. It **has** controlled data redundancy and enforces integrity between various entities.

## Student details:

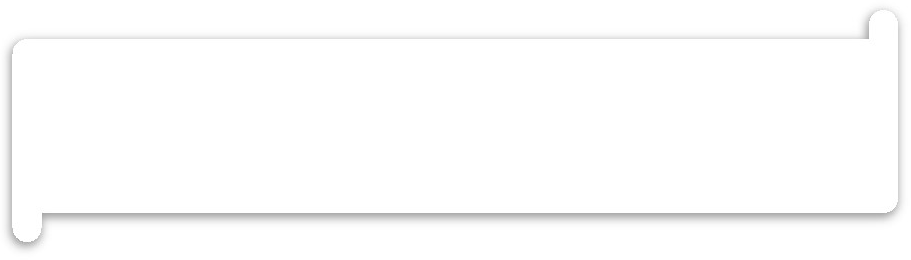
|  |  |
| --- | --- |
| **Column** | **Type** |
| Dept | String |
| Fingerprinttemplate | String |
| Fname | String |
| Imageurl | String |
| Name | String |
| Reg No. | String |
| Sem | String |

**Subject details:**

|  |  |
| --- | --- |
| **Column** | **Type** |
| Dresscode | Int |
| Dressname | Varchar(20) |
| Agegroup | Varchar(10) |
| Making charges | Float |
| Image | Varchar(50) |
| Dresstype | Varchar(10) |
| Fittingparameters | Varchar(100) |



**CHAPTER-6**



**PROGRAM CODE**

package com.example.sugan.project; import android.app.ProgressDialog; import android.content.Intent; import android.os.AsyncTask; import android.os.Bundle;

import android.support.v7.app.AppCompatActivity; import android.view.View;

import android.widget.Button; import android.widget.EditText; import android.widget.TextView; import java.io.BufferedReader; import java.io.BufferedWriter; import java.io.IOException; import java.io.InputStream; import java.io.InputStreamReader; import java.io.OutputStream;

import java.io.OutputStreamWriter; import java.net.HttpURLConnection; import java.net.URL;

import java.net.URLEncoder;

public class register extends AppCompatActivity { Button btn,btnlogin;

TextView tv;

EditText et1, et2, et3, et4, et5,et6,et7;

String registerURL = Urlsettings.weburl + "register.php"; @Override

protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.activity\_register); tv=(TextView) findViewById(R.id.tv2); et1=(EditText) findViewById(R.id.et1); et2=(EditText) findViewById(R.id.et2); et3=(EditText) findViewById(R.id.et3); et4=(EditText) findViewById(R.id.et4); et5=(EditText) findViewById(R.id.et5); et6=(EditText) findViewById(R.id.et6); et7=(EditText) findViewById(R.id.et7); btn=(Button)findViewById(R.id.btn); btnlogin=(Button)findViewById(R.id.btnlogin);

btnlogin.setOnClickListener(new View.OnClickListener() { @Override

public void onClick(View v) {

Intent intent=new Intent(register.this, login.class); startActivity(intent)

});

btn.setOnClickListener(new View.OnClickListener() {

String name, address, city, pincode, mobile, email, password; @Override

public void onClick(View v) { name = et1.getText().toString(); address = et2.getText().toString(); city=et3.getText().toString();

pincode=et4.getText().toString(); mobile = et5.getText().toString(); email = et6.getText().toString(); password = et7.getText().toString();

AsyncTaskRunner runner = new AsyncTaskRunner(); runner.execute(name, address, city, pincode, mobile, email, password);

});

}

//doinbackgroud, progressupdate, postexecute-result parameter

class AsyncTaskRunner extends AsyncTask<String, Void, String> { private String resp;

ProgressDialog progressDialog; String msg = ""; HttpURLConnection conn; @Override

protected void onPreExecute() {

progressDialog = ProgressDialog.show(register.this, "ProgressDialog", "please wait while sending data to server");

<RelativeLayout xmlns:android="<http://schemas.android.com/apk/res/android>"

xmlns:tools="<http://schemas.android.com/tools>" android:layout\_width="match\_parent" android:layout\_height="match\_parent"

android:paddingLeft="@dimen/activity\_horizontal\_margin" android:paddingRight="@dimen/activity\_horizontal\_margin" android:paddingTop="@dimen/activity\_vertical\_margin" android:paddingBottom="@dimen/activity\_vertical\_margin" tools:context=".MainActivity" android:background="#4dffee"

<TextView android:layout\_width="350dp" android:layout\_height="50dp" android:text="CUSTOMER LOGIN"

android:id="@+id/tv"

android:textSize="20dp" android:textStyle="bold" android:gravity="center" />

<EditText android:layout\_width="250dp" android:layout\_height="50dp" android:layout\_marginTop="20dp" android:id="@+id/et1" android:gravity="center" android:textStyle="bold" android:layout\_below="@+id/tv" android:layout\_marginLeft="50dp" android:text=""/>

<TextView android:layout\_width="250dp" android:layout\_height="50dp" android:layout\_below="@+id/bt1" android:layout\_marginLeft="50dp" android:layout\_marginTop="60dp" android:gravity="center"

android:text="-----------OR "/>

<Button

android:layout\_width="200dp" android:layout\_height="50dp" android:text="New registration" android:id="@+id/bt2" android:layout\_below="@+id/tv2" android:layout\_marginLeft="80dp" android:layout\_marginTop="20dp" />



**CHAPTER-7**



**TESTING**

Software testing is an investigation conducted to provide stakeholders with information about the quality of the software product or service under test.[1] Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include the process of executing a program or application with the intent of finding software bugs (errors or other defects), and verifying that the software product is fit for use.

Software testing involves the execution of a software component or system component to evaluate one or more properties of interest. In general, these properties indicate the extent to which the component or system under test:

* meets the requirements that guided its design and development,
* responds correctly to all kinds of inputs,
* performs its functions within an acceptable time,
* it is sufficiently usable,
* can be installed and run in its intended environments, and
* achieves the general result its stakeholders desire.

##### Objectives of Testing:

Software Testing has different goals and objectives.The major objectives of Software testing are as follows:

* + Finding defects which may get created by the programmer while developing the software.
  + Gaining confidence in and providing information about the level of quality.
  + To prevent defects.
  + To make sure that the end result meets the business and user requirements.
  + To ensure that it satisfies the BRS that is Business Requirement Specification and SRS that is System Requirement Specifications.
  + To gain the confidence of the customers by providing them a quality product.

##### Principles of Testing:

There are seven principles in software testing:

* + - Testing shows presence of defects.
    - Exhaustive testing is not possible.
    - Early testing.
    - Defect clustering.
    - Pesticide paradox.
    - Testing is context dependent.
    - Absence of errors fallacy.

##### Software testing strategies:

A strategy of software testing is shown in the context of spiral.

* 1. **Unit testing**

Unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use. Tests are typically automated tests written and run by software developers to ensure that a section of an application (known as the "unit") meets its design and behaves as intended. In procedural programming, a unit could be an entire module, but it is more commonly an individual function or procedure. In object-oriented programming, a unit is often an entire interface, such as a class, but could be an individual method.By writing tests first for the smallest testable units, then the compound behaviors between those, one can build up comprehensive tests for complex applications.

* 1. **Integration testing**

Integration testing is any type of software testing that seeks to verify the interfaces between components against a software design. Software components may be integrated in an iterative way or all together ("big bang"). Normally the former is considered a better practice since it allows interface issues to be located more quickly and fixed.

Integration testing works to expose defects in the interfaces and interaction between integrated components (modules). Progressively larger groups of tested software components corresponding to elements of the architectural design are integrated and tested until the software works as a system.

**Validation testing**

Check all the requirements like functional, behavioral and performance requirement are validate against the construction software.

* 1. **System testing**

System testing is testing conducted on a complete integrated system to evaluate the system's compliance with its specified requirements.System testing takes, as its input, all of the integrated components that have passed integration testing. The purpose of integration testing is to detect any inconsistencies between the units that are integrated together (called assemblages). System testing seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.[citation needed] The actual result is the behavior produced or observed when a component or system is tested

**Alpha testing**

Alpha testing is simulated or actual operational testing by potential users/customers or an independent test team at the developers' site. Alpha testing is often employed for off-the-shelf software as a form of internal acceptance testing before the software goes to beta testing.[56]

**Beta testing**

Beta testing comes after alpha testing and can be considered a form of external user acceptance testing. Versions of the software, known as beta versions, are released to a limited audience outside of the programming team known as beta testers. The software is released to groups of people so that further testing can ensure the product has few faults or bugs. Beta versions can be made available to the open public to increase the feedback field to a maximal number of future users and to deliver value earlier, for an extended or even indefinite period of time (perpetual beta).

**Acceptance testing**:

User acceptance testing and Alpha and beta testing are described in the next testing types section. Operational acceptance is used to conduct operational readiness of a product, service or system as part of a quality management system. OAT is a common type of non-functional software testing, used mainly in software development and software maintenance projects.

**Module testing**

Module testing is defined as a software testing type, which checks individual subprograms, subroutines, classes, or procedures in a program. Instead of testing whole software program at once, module testing recommends testing the smaller building blocks of the program.

Module testing is largely a white box oriented. The objective of doing Module, testing is not to demonstrate proper functioning of the module but to demonstrate the presence of an error in the module.

##### Test Approach:

**Black box testing**

* Incorrect or missing functions
* Interface errors
* Errors in external database errors
* Performance errors
* Initialization and terminate errors.

##### White Box Testing:

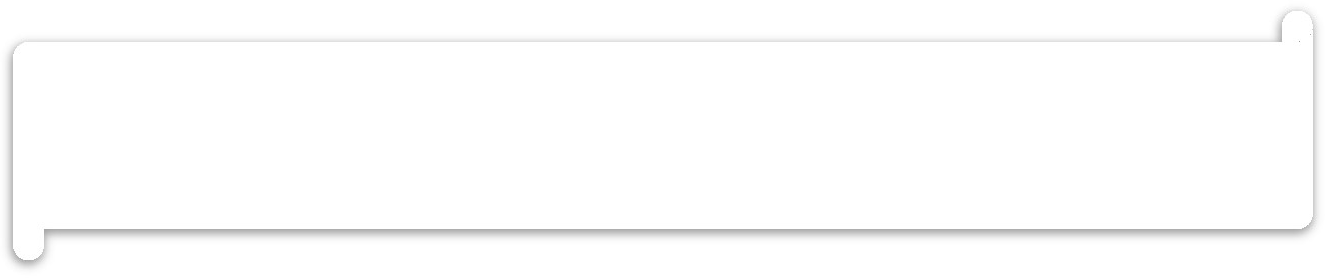
* + - Check whether all independent paths within a module at least once.
    - Exercise all logic are true and false
    - Exercise all the loops within boundaries within their boundaries.
    - Exercise in the internal data structure to ensure validity
    - Ensure whether the possible validity look ups have been provided to validate data entry.

##### Top down integration testing offers several advantages.

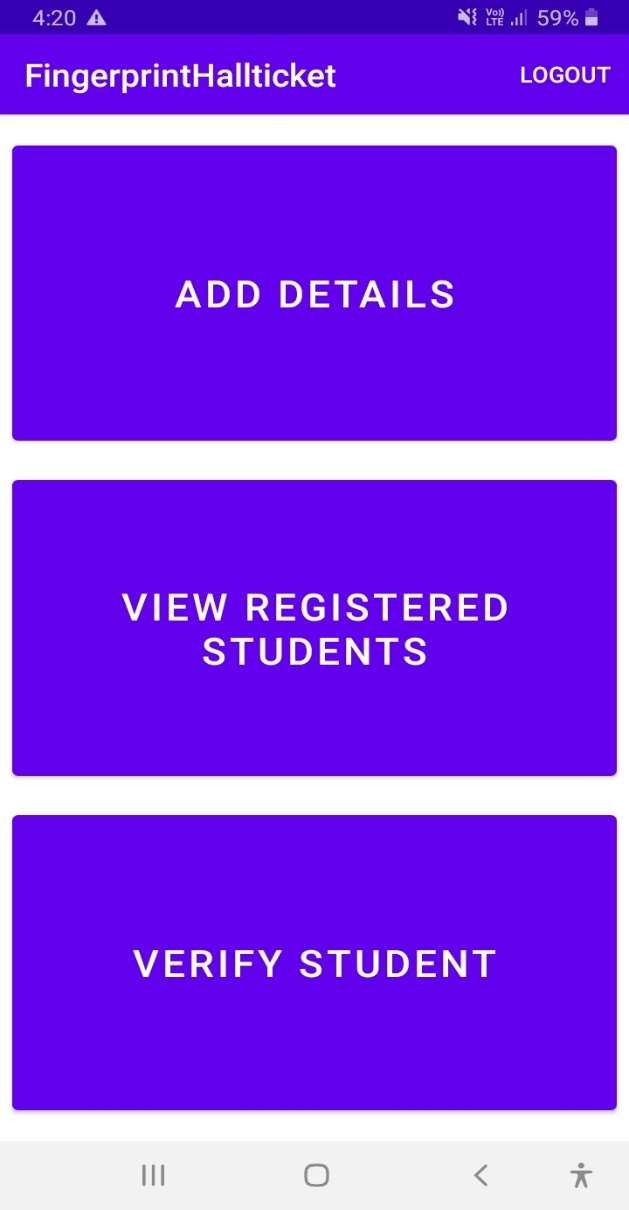
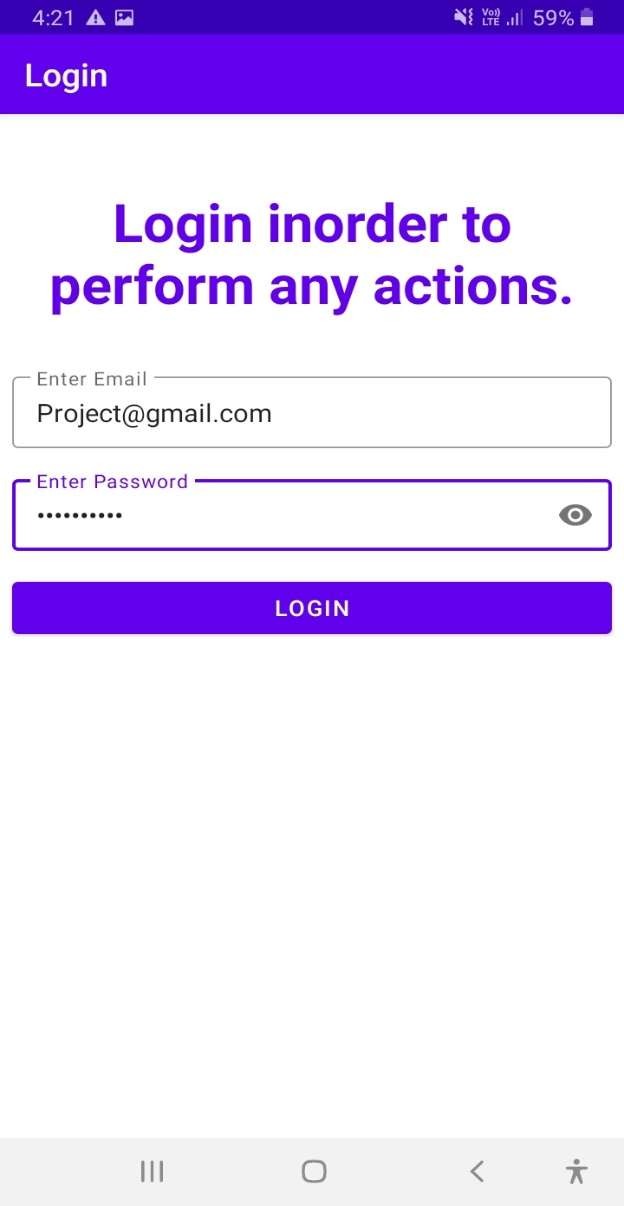
* + - System integration is distributed throughout implementation phase.
    - Top level interface are tested first most often.
    - The top level routines provide a natural test harness for lower level routines.
    - Errors are localized to the new modules and interface being added.

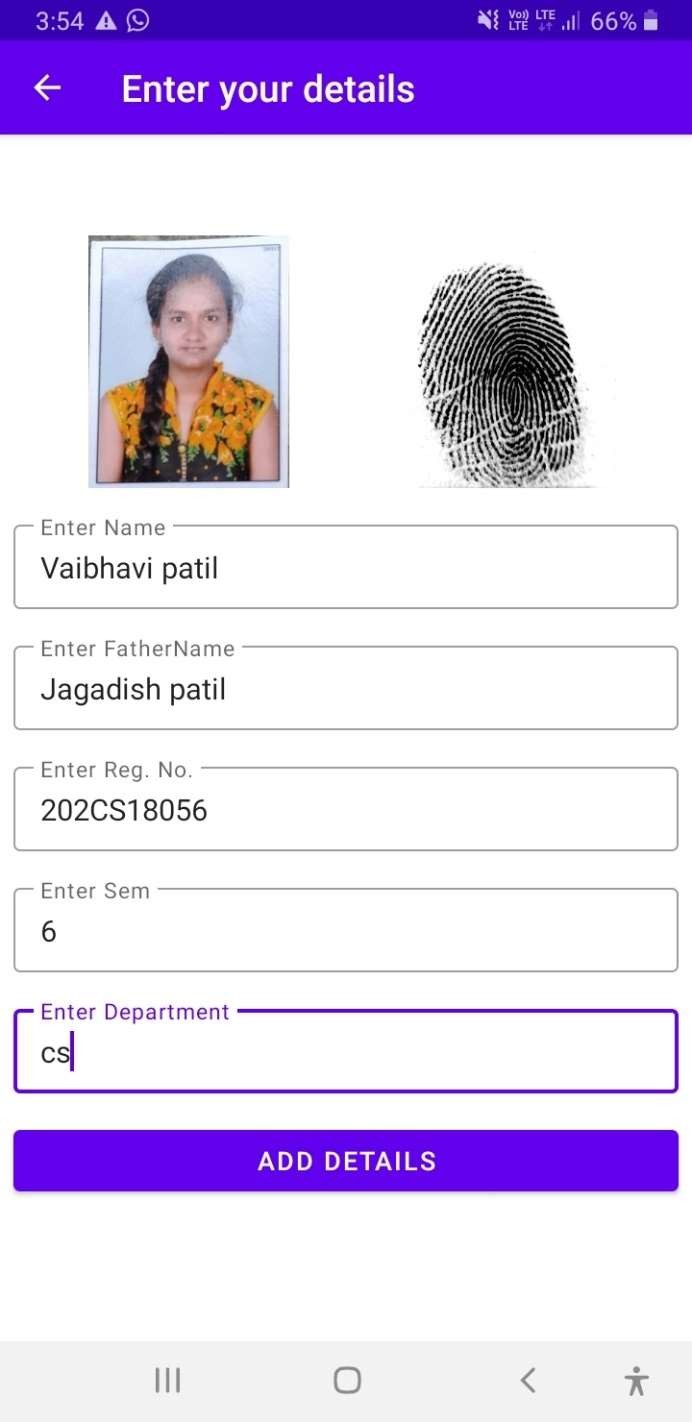
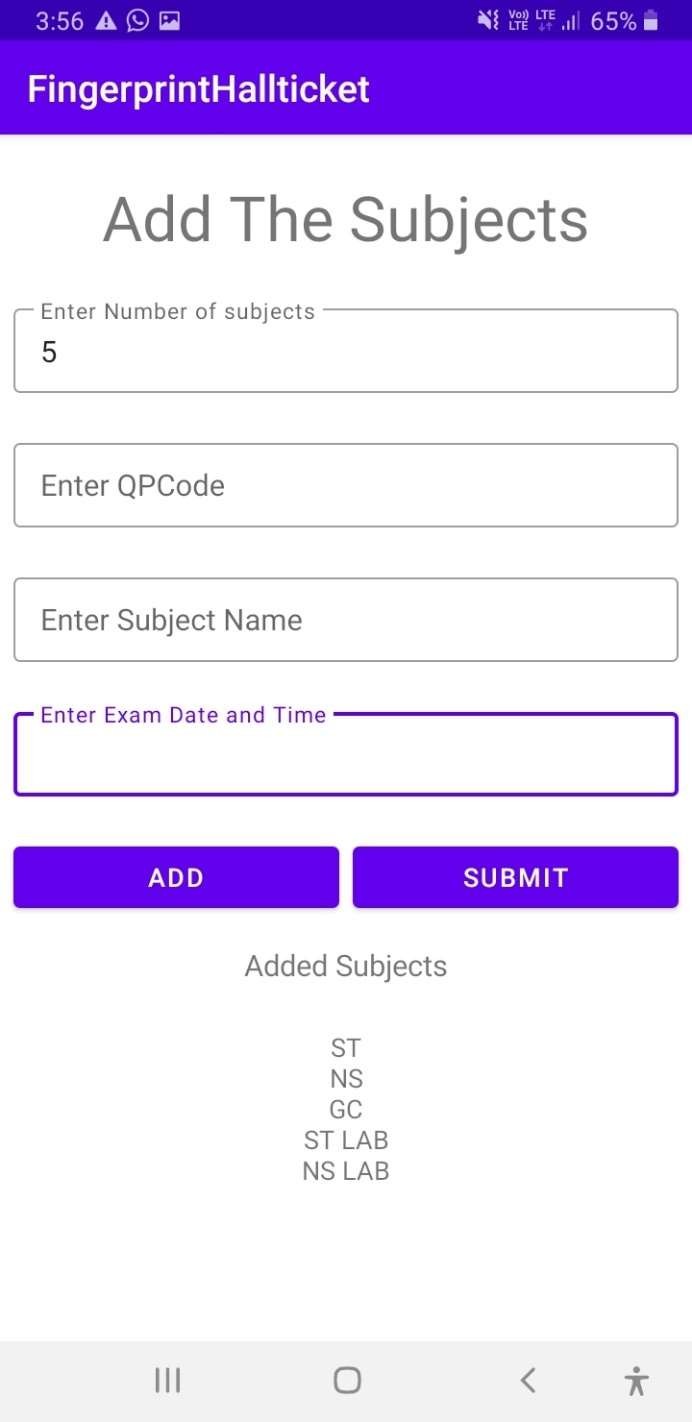


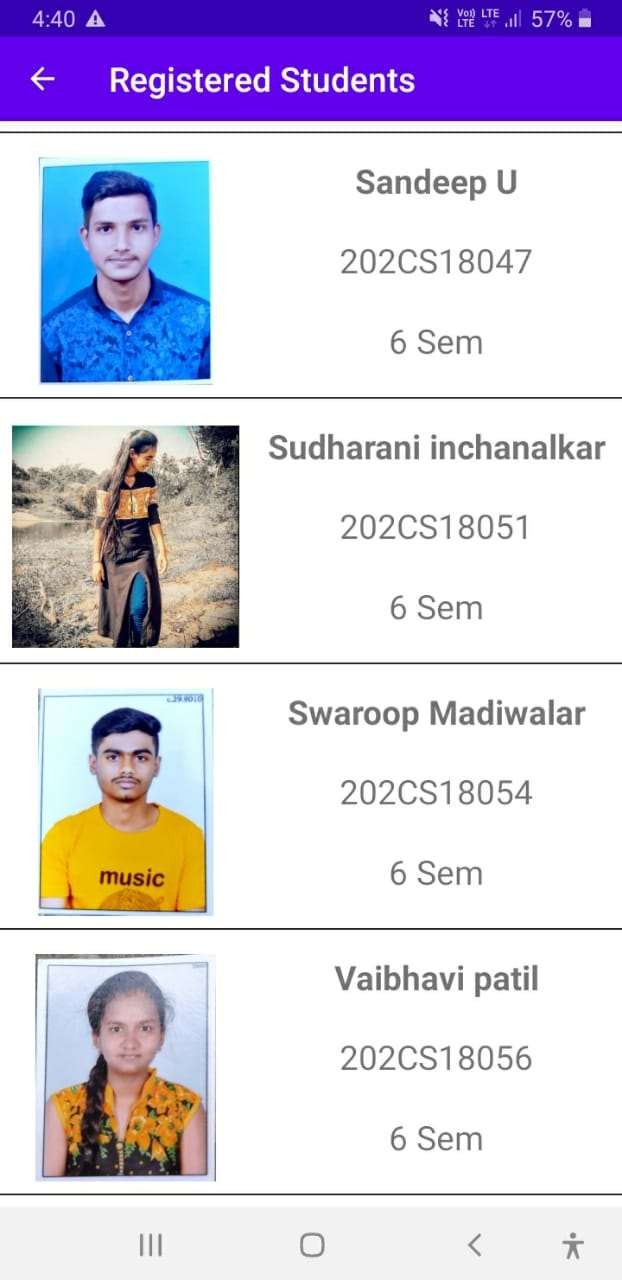
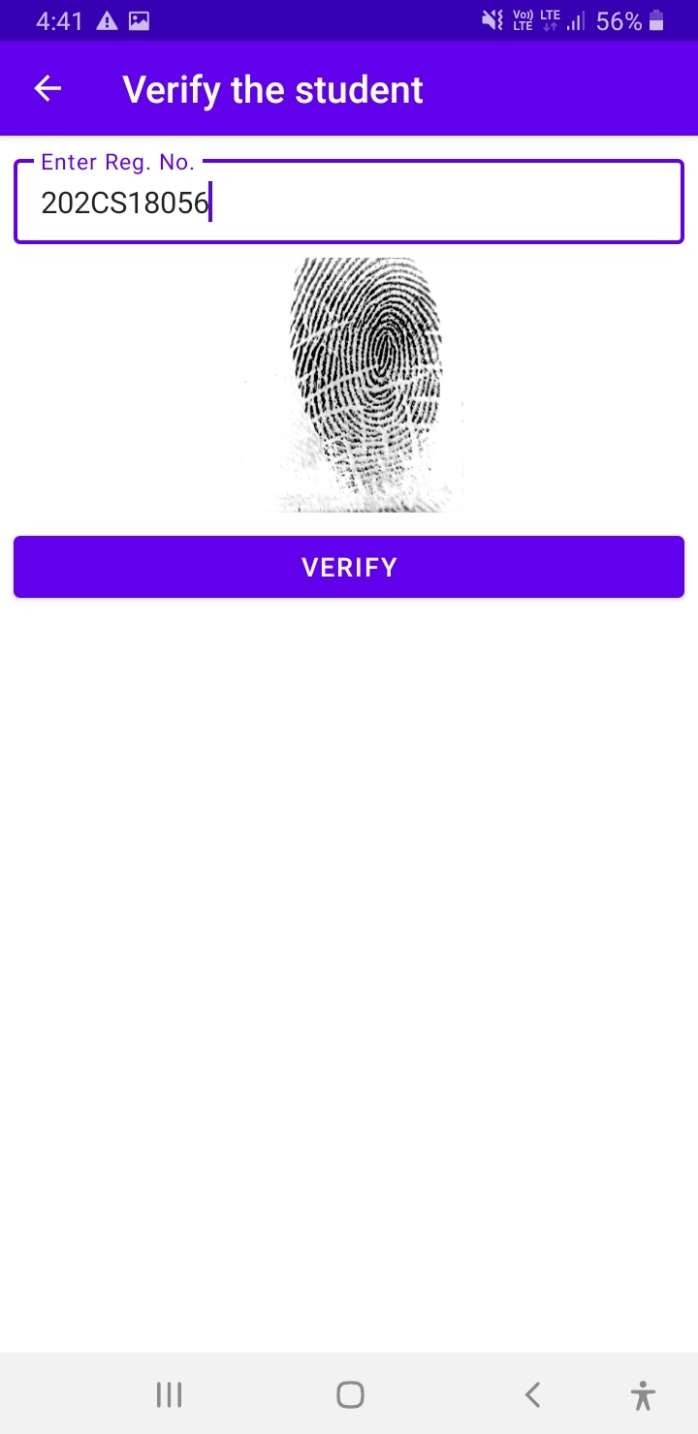
**CHAPTER-8**

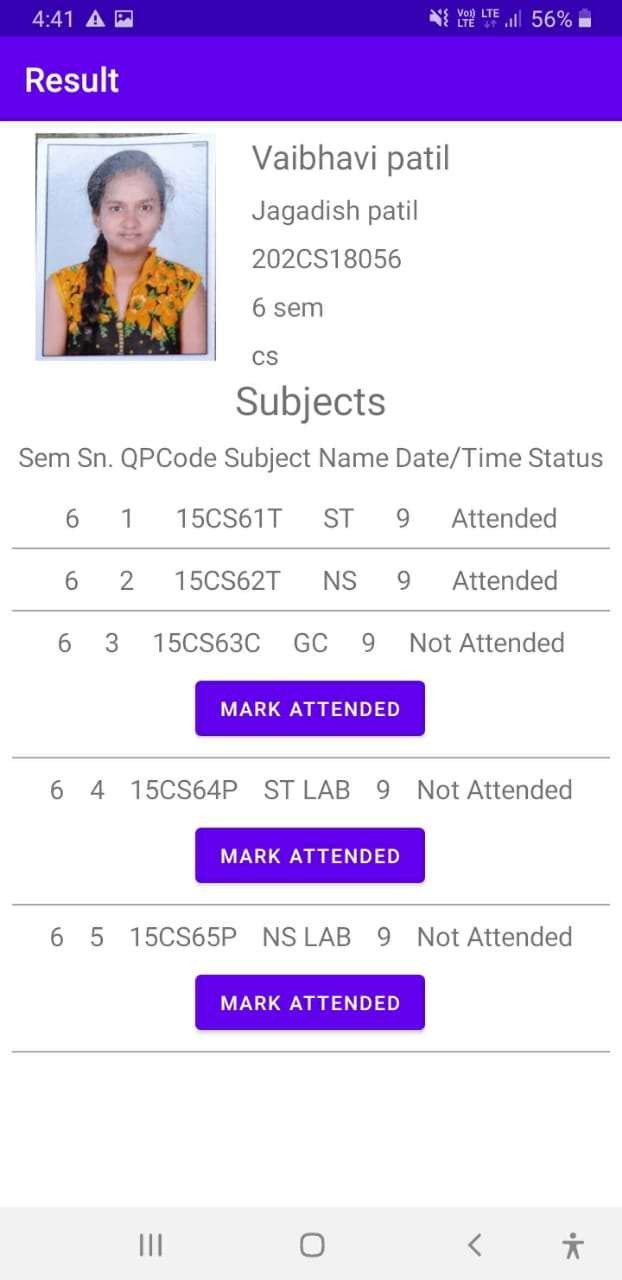


**INPUT AND OUTPUT SCREEN**



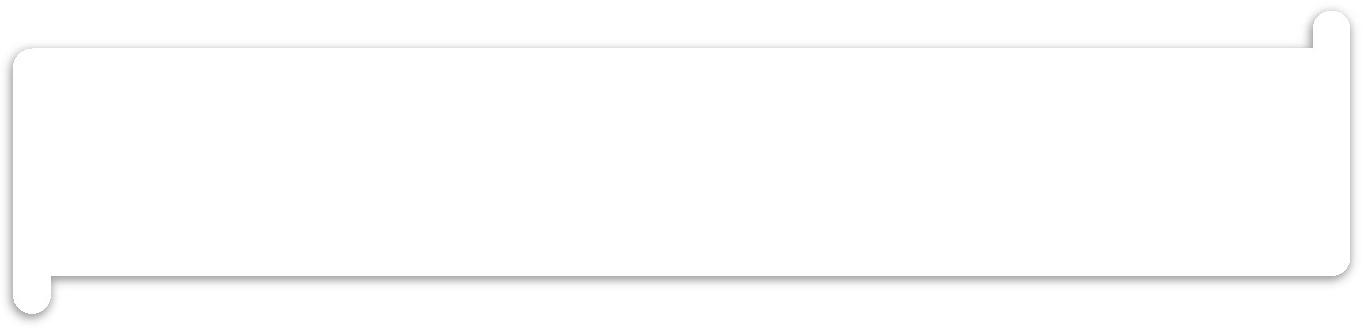
 





**CHAPTER-9**



**LIMITATIONS OF THE PROJECT**

**Limitation of project:**

* + - * Need to depend on Internet
      * Requirement of hardware
      * Little Time consuming while registering.
      * Need fast database server.



**CHAPTER-10**



**FUTURE SCOPE OF THE PROJECT**

# Scope and future enhancement:

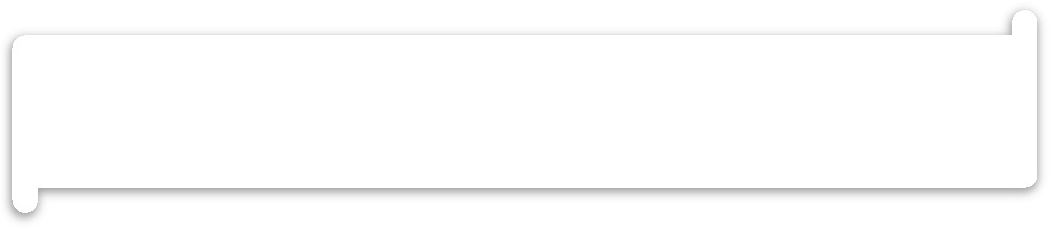
It work for android devices. Fingerprint hall ticket system stores the student’s information like name, register number, semester, department in the database. The subjects for which the students have to attend the examination are also stored in the database while registering that student. While attending the examination the student has to get verified by placing their fingerprint which they previously gave for registration. If the student is valid then they are allowed to attend the examination and the subject which they are attending are marked as attended and stored back in the database.

This project needs future enhancements like –

* + - Editing the student’s information.
    - Automatic attendance marking based on time.
    - Printing the hall ticket for future uses.



**CHAPTER-11**



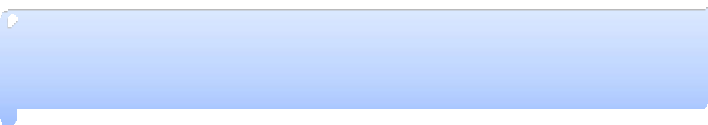
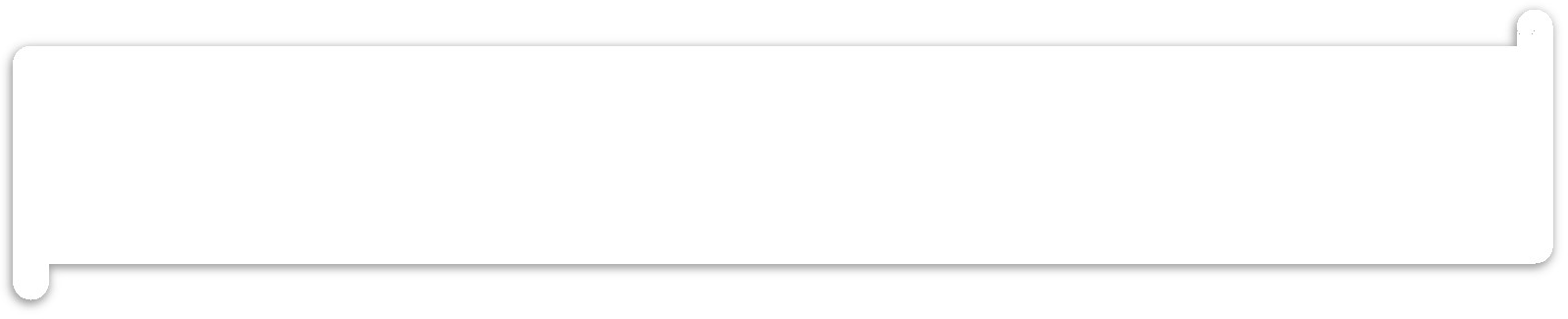
**CONCLUSION**

**CONCLUSION:**

* The aim of this project is to develop a student identification system using biometric features.
* This system is used to identify the student based on fingerprint recognition.
* The fingerprint is captured using Mantra mfs100 scanner.
* The fingerprint obtained is processed and matched using mfs100 sdk on android device.



**CHAPTER-12**



**REFERENCES AND BIBLIOGRAPHY**

**BIBLOGRAPHY**

By using some books we created application for our project through some references of the books and as well as online resources.

##### REFERENCE

* + Software Engineering“ Ivan Somerville, sixth edition
  + System Analysis and design”, Eilmasri & navathe

#### Website

* https://howtofirebase.com/what-is-firebase-fcb8614ba442
* https://[www.geeksforgeeks.org/constraintlayout-in-android/](http://www.geeksforgeeks.org/constraintlayout-in-android/)
* https://[www.learnhowtoprogram.com/android/introduction-to-android/introduction-](http://www.learnhowtoprogram.com/android/introduction-to-android/introduction-) to-xml-and-android-layouts
* https://firebase.google.com/docs/database
* https://firebase.google.com/docs/storage
* https://firebase.google.com/docs/auth
* https://github.com/bumptech/glide

**LOG SHEET**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SL.NO** | **DATE** | **TASK** | **PROGRESS OF TASK** | **STAFF SIGN** |
| 1 | 27/12/2019 | Collection of data Library | COMPLETED |  |
| 2 | 01/01/2020 | Collection of data Internet | COMPLETED |  |
| 3 | 15/01/2020 | Collection of data Portals | COMPLETED |  |
| 4 | 20/01/2020 | Writing code for Project | COMPLETED |  |
| 5 | 30/01/2020 | Writing code for Project | COMPLETED |  |
| 6 | 03/02/2020 | Writing code for Project | COMPLETED |  |
| 7 | 10/02/2020 | Analysis of the developing project | COMPLETED |  |
| 8 | 15/02/2020 | Writing code for the database | COMPLETED |  |
| 9 | 24/02/2020 | Writing code for the Project connectivity | COMPLETED |  |
| 10 | 30/02/2020 | Writing code for the modules | COMPLETED |  |
| 11 | 02/03/2020 | Connecting the modules with the database | COMPLETED |  |
| 12 | 05/03/2020 | Analysis of the developing project | COMPLETED |  |
| 13 | 07/03/2020 | Modification of the web view | COMPLETED |  |
| 14 | 12/03/2020 | Writing code for the admin privileges | COMPLETED |  |
| 15 | 15/03/2020 | Updating database for admin | COMPLETED |  |
| 16 | 16/03/2020 | Updating database for menu | COMPLETED |  |
| 17 | 17/03/2020 | Updating database for feedback | COMPLETED |  |
| 18 | 17/03/2020 | Review-2 | COMPLETED |  |
| 19 | 18/03/2020 | Consulting the guide for suggestions | COMPLETED |  |
| 20 | 22/03/2020 | Implementing the guidelines given by the  guide | COMPLETED |  |
| 21 | 22/03/2020 | Report writing | COMPLETED |  |
| 22 | / /2020 | Report submission | COMPLETED |  |

**CIE ASSESMENT FOR FINAL REVIEW**

|  |  |  |  |
| --- | --- | --- | --- |
| **SL.NO** | **PARTICULARS** | **MAX MARKS** | **MARKS OBTAINED** |
| **1** | LOG OF ACTIVITY (plan and schedule) | **05** |  |
| **2** | REPORT | **10** |  |
| **3** | PRESENTATION | **10** |  |
|  | **TOTAL** | **25** |  |

##### SIGNATURE OF GUIDE SIGNATURE OF HOD

**PRESENTATION**

