**1. INTRODUCTION**

**1.1 Project Description**

“Auditorium Management System [hOLO] “, A Real-time Application Specifically designed and developed to manage the booking of Auditorium. It allows the users to Book Auditoriums based on Time and Date which allows the users to manage their bookings and requirements in the auditorium. It also allows the users to cancel their Bookings.

The Application has 3 different Users:

1. Admin
2. Staff
3. Non-Teaching Staff / Technical Team.

Admin: Admin logs-in to the application using Credentials provided.

Functions of Admin:

1. Add User/Auditorium
2. Delete User/Auditorium
3. Book/Cancel Auditorium Bookings
4. Change Password
5. Update User
6. Retrieve all Bookings
7. Retrieve completed events

Staff: Staff logs-in to the application using Credentials provided.

Functions of Staff:

1. Book Auditorium
2. Change Password
3. View Bookings
4. Cancel Bookings

Non-Teaching Staff / Technical Team: They log-in to the application using Credentials provided.

Functions:

1. View All Bookings
2. Book Auditorium
3. Change Password

**1.2 Problem Statement**

“The main aim for developing the application is to try overcoming the traditional way of Booking by allowing the users to book Auditorium anytime and anywhere.”

**1.3 Existing System**

Currently The User has to mention his requirements in the Event Scheduler.

Drawbacks:

* User can book an auditorium if he/she is available inside the Institute.
* Manually mention their requirements.

**1.4 Proposed System**

The proposed system allows the users to book auditoriums over Internet anywhere and at any point of time. The application gathers all the requirements that the users wants to be present within the auditorium.

**1.5 Objectives**

* To overcome traditional method of booking using Online-Booking.
* To provide the end- user with timely access to resource.

**1.6 Benefits of the Proposed System**

* Processing Speed.
* Easy Booking.
* Convenience to Customers.
* Results in smooth functioning of organization.
* Satisfaction to Customers.
* Better performance in Customer Flow.

**2. System Requirement**

**2.1 Hardware Specification (Minimum)**

* Processor : Intel i3 minimum.
* Processor Speed : 2.8 GHz Processor
* RAM : 4 GB RAM minimum, 8 GB RAM recommended
* Hard Disk : 2 GB of available disk space minimum, 4 GB Recommended

(500 MB for IDE + 1.5 GB for Android SDK and Emulator

System image).

* Monitor : 1280 x 800 minimum screen resolution

**2.2 Software Specification**

* Operating System : Microsoft Windows 7/8/10 (32-bit or 64-bit). The Android

Emulator supports 64-bit Windows only.

* Front End : Android Studio.
* Back End : Firebase
* Language Used : Java
* Documentation : Microsoft Office 2013.

**2.3 Tools Survey**

**2.3.1 Android Studio**

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on Jet Brains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems. It is a replacement for the Eclipse Android Development Tools (ADT) as the primary IDE for native Android application development.

Android Studio was announced on May 16, 2013 at the Google I/O conference. It was in early access preview stage starting from version 0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014. The first stable build was released in December 2014, starting from version 1.0.

On May 7, 2019, Kotlin replaced Java as Google’s preferred language for Android app development. Java is still supported, as is C++.

**Features**

* 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.

Android Studio supports all the same programming languages of IntelliJ(and CLion) e.g. Java, C++, and more with extensions, such as Go; and Android Studio 3.0 or later supports Kotlin and "all Java 7 language features and a subset of Java 8 language features that vary by platform version." External projects backport some Java 9 features. While IntelliJ that Android Studio is built on supports all released Java versions, and Java 12, it's not clear to what level Android Studio supports Java versions up to Java 12 (the documentation mentions partial Java 8 support). At least some new language features up to Java 12 are usable in Android.

**2.3.2 Firebase**

Firebase evolved from Envolve, a prior startup founded by James Tamplin and Andrew Lee in 2011. Envolve provided developers an API that enables the integration of online chat functionality into their websites. After releasing the chat service, Tamplin and Lee found that it was being used to pass application data that were not chat messages. Developers were using Envolve to sync application data such as game state in real time across their users. Tamplin and Lee decided to separate the chat system and the real-time architecture that powered it. They founded Firebase as a separate company in September 2011 and it launched to the public in April 2012.

Firebase's first product was the Firebase Real-time Database, an API that synchronizes application data across iOS, Android, and Web devices, and stores it on Firebase's cloud. The product assists software developers in building real-time, collaborative applications.

In May 2012, a month after the beta launch, Firebase raised $1.1 million in seed funding from venture capitalists Flybridge Capital Partners, Greylock Partners, Founder Collective, and New Enterprise Associates. In June 2013, the company further raised $5.6 million in Series A funding from Union Square Ventures and Flybridge capital Partners.

In 2014, Firebase launched two products. Firebase Hosting and Firebase Authentication. This positioned the company as a mobile backend as a service.

In October 2014, Firebase was acquired by Google. A year later, in October 2015, Google acquired Divshot, an HTML5 web-hosting platform, to merge it with the Firebase team.

In May 2016, at Google I/O, the company's annual developer conference, Firebase introduced Firebase Analytics and announced that it was expanding its services to become a unified backend-as-a-service (BaaS) platform for mobile developers. Firebase now integrates with various other Google services, including Google Cloud Platform, AdMob, and Google Ads to offer broader products and scale for developers. Google Cloud Messaging, the Google service to send push notifications to Android devices, was superseded by a Firebase product, Firebase Cloud Messaging, which added the functionality to deliver push notifications to both iOS and web devices. In January 2017, Google acquired Fabric and Crashlytics from Twitter to add those services to Firebase.

In October 2017, Firebase has launched Cloud Firestore, a real-time document database as the successor product to the original Firebase Real-time Database.

**3. DESIGN SPECIFICATION**

**3.1 DATA DICTIONARY**

A data dictionary is a file or a set of files that contains a database's metadata. The data dictionary contains records about other objects in the database, such as data ownership, data relationships to other objects and other data.

The data dictionary is a crucial component of any relational database. Ironically, because of its importance, it is invisible to most database users. Typically, only database administrators interact with the data dictionary.

3.1.1 AUTH TABLE

|  |  |  |
| --- | --- | --- |
| **FIELDNAME** | **DATATYPE** | **CONSTRAINT** |
| UID | Varchar | Primary Key |
| Email | Varchar | Unique |
| Password | Varchar |  |

3.1.2 Users

|  |  |  |
| --- | --- | --- |
| **FIELDNAME** | **DATATYPE** | **CONSTRAINT** |
| Email | Varchar | references |
| Name | Varchar |  |
| Block | Varchar |  |
| Phone | Number |  |
| User type | Varchar |  |
| UID | Varchar | Foreign Key |

3.1.3 Galleries

|  |  |  |
| --- | --- | --- |
| **FIELDNAME** | **DATATYPE** | **CONSTRAINT** |
| Name | Varchar |  |
| Block | Varchar |  |
| Date | Date |  |
| Floor | Number |  |
| Image | Varchar | Unique |
| Pid | Varchar | Primary key |
| Seats | Number |  |
| Time | Date |  |

3.1.4 Bookings

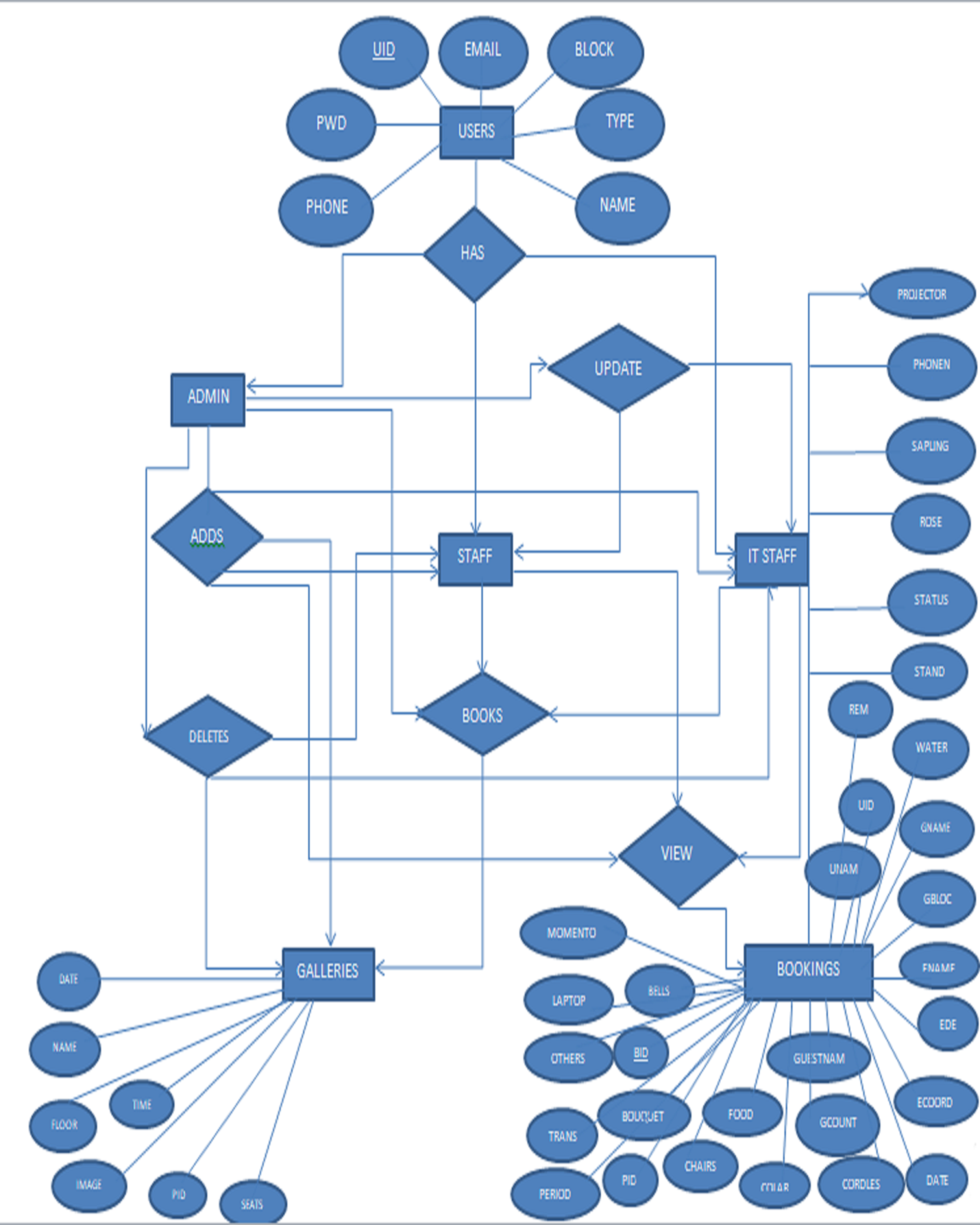
|  |  |  |
| --- | --- | --- |
| **FIELDNAME** | **DATATYPE** | **CONSTRAINT** |
| Bells | Varchar |  |
| BookingID | Varchar | Primary Key |
| Bouquet | Varchar |  |
| Chairs | Varchar |  |
| Colar | Varchar |  |
| Cordless | Varchar |  |
| Date | Date |  |
| EventCoordinator | Varchar |  |
| EventDepartment | Varchar |  |
| EventName | Varchar |  |
| FoodSnacks | Varchar |  |
| FromTime | Date |  |
| GalleryBlock | Varchar |  |
| GalleryFloor | Varchar |  |
| GalleryID | Varchar |  |
| GalleryName | Varchar |  |
| GuestCount | Number |  |
| GuestName | Varchar |  |
| Laptop | Varchar |  |
| Momento | Varchar |  |
| Others | Varchar | Nullable |
| PhoneNo | Number |  |
| Projector | Varchar |  |
| Remuneration | Varchar |  |
| Rose | Varchar |  |
| Sapling | Varchar |  |
| StandMike | Varchar |  |
| Status | Varchar |  |
| ToTime | Date |  |
| Transportation | Varchar | Nullable |
| UserID | Varchar |  |
| UserName | Varchar |  |
| WaterBottles | Varchar |  |

**3.2 ENTITY RELATIONSHIP DIAGRAM**

A graphical representation of the entities and the relationships between them. Entity relationship diagrams are a useful medium to achieve a common understanding of data among users and application developers.

In data modeling, an entity-relationship model (ERM) is a representation of structured data; Entity - relationship modeling is the process of generating these models. The end-product of the modeling process is an entity-relationship diagram (ERD), a type of Conceptual Data Model or Semantic Data Model.

|  |  |  |
| --- | --- | --- |
| **Symbol** | **Name** | **Definition** |
| C:\Users\Harsha\Pictures\Attribute.png | Attribute | An attribute describes the property of an entity. An attribute is represented as Oval in an ER diagram. |
| C:\Users\Harsha\Pictures\DerivedAttribute.png | Derived  Attribute | A derived attribute is one whose value is dynamic and derived from another attribute. It is represented by **dashed oval** in an ER Diagram. |
| C:\Users\Harsha\Pictures\Entity.png | Entity | An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is an object, a component of data. An entity set is a collection of similar entities. These entities can have attributes that define its properties |
| C:\Users\Harsha\Pictures\MultivaluedAttribute.png | Multivalued  Attribute | An attribute that can hold multiple values is known as multivalued attribute. It is represented with **double ovals** in an ER Diagram. |
| C:\Users\Harsha\Pictures\RelationshipShape.png | Relationship | A relationship is represented by diamond shape in ER diagram, it shows the relationship among entities. |

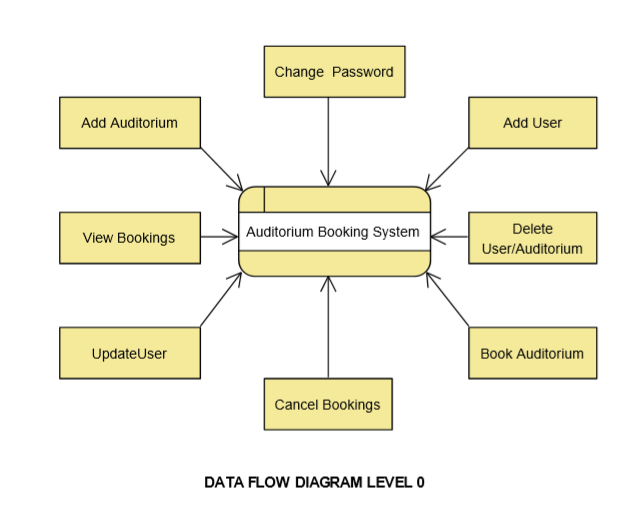
****

**Fig 3.2.1 Entity relationship Diagram**

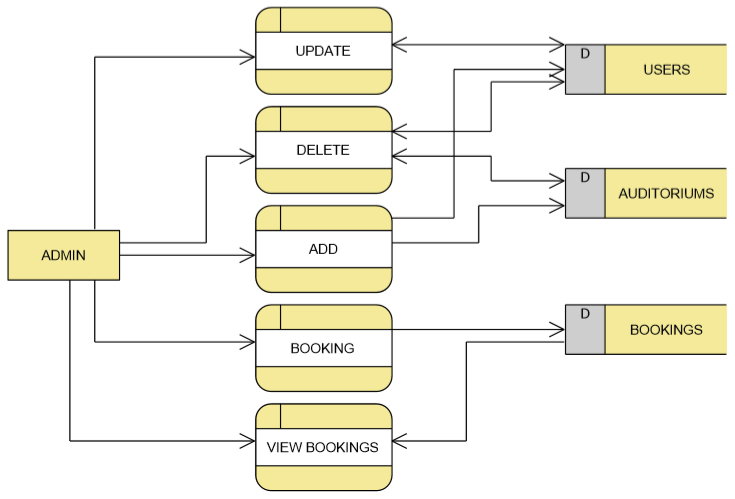
**3.3 DATA FLOW DIAGRAM**

The data flow diagrams are pictorial or graphical representation of the outline of the system study. The data flow diagram covers all the processes and data storage area which takes place during any transaction in the system. The data flow diagrams are functionally divided into context level, Zero level and First level data flow diagrams.

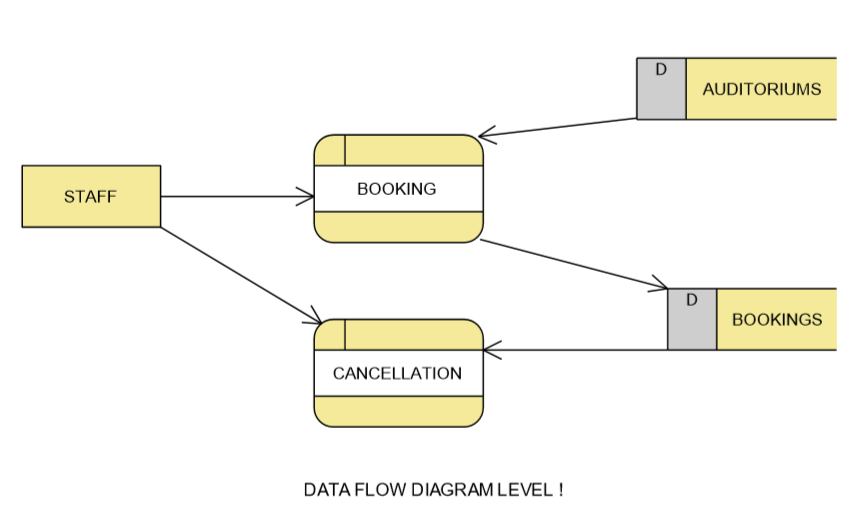
|  |  |  |
| --- | --- | --- |
| **Symbol** | **Name** |  |
|  | Process | The process (function, transformation) is part of a system that transforms inputs to outputs. The symbol of a process is a circle, an oval, a rectangle or a rectangle with rounded corners (according to the type of notation). The process is named in one word, a short sentence, or a phrase that is clearly to express its essence. |
|  | External  Entity | An **external entity** sends or receives data from the system. It can represent a person, a machine, an organization etc., that is **external** to the system being modeled. |
|  | Data  Flow | Data flow (flow, dataflow) shows the transfer of information (sometimes also material) from one part of the system to another. The symbol of the flow is the arrow. |
|  | Data  Store | These are files or repositories that hold information for later use, such as a database table or a membership form. |

****

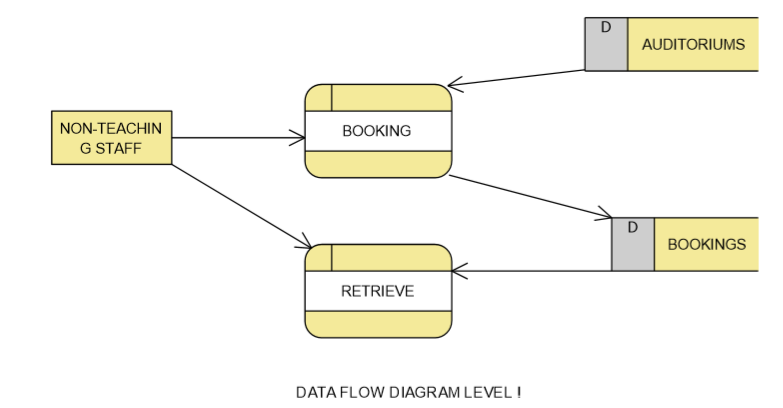
**Fig 3.3.1 Data Flow Diagram Level 0**

****

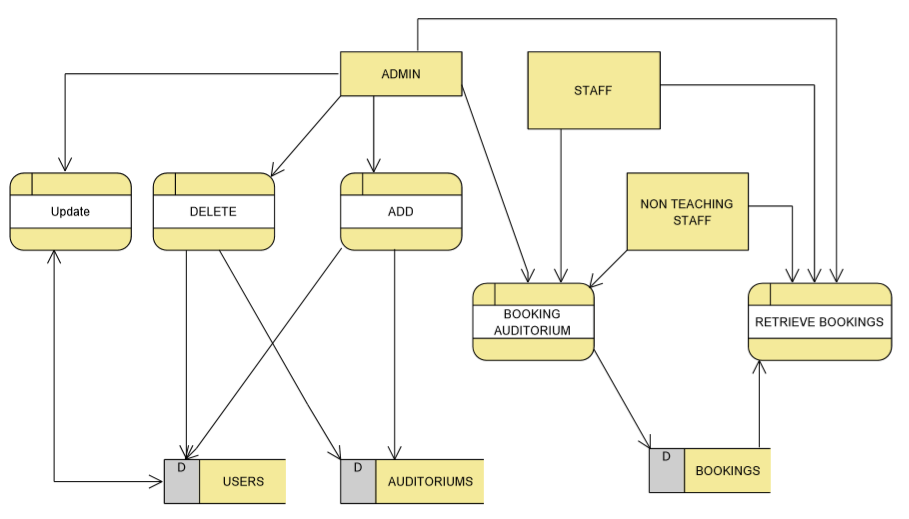
**Fig 3.3.2 Data Flow Diagram Level 1**

****

**Fig 3.3.3 Data Flow Diagram Level 1**

****

**Fig 3.3.4 Data Flow Diagram Level 1**



**Fig 3.3.5 Data Flow Diagram Level 2**

**4. CODING**

**LOGIN PAGE**

**XML CODE**

<?**xml version="1.0" encoding="utf-8"**?>  
<**androidx.constraintlayout.widget.ConstraintLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="@drawable/image"  
 tools:context=".MainActivity"**>  
  
 <**androidx.constraintlayout.widget.Guideline  
 android:id="@+id/guideline"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 app:layout\_constraintGuide\_begin="20dp"  
 app:layout\_constraintGuide\_percent="0.05"** />  
  
 <**androidx.constraintlayout.widget.Guideline  
 android:id="@+id/guideline2"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 app:layout\_constraintGuide\_begin="20dp"  
 app:layout\_constraintGuide\_percent="0.95"** />  
  
 <**androidx.constraintlayout.widget.Guideline  
 android:id="@+id/guideline3"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 app:layout\_constraintGuide\_begin="20dp"  
 app:layout\_constraintGuide\_percent="0.10"** />  
  
 <**androidx.constraintlayout.widget.Guideline  
 android:id="@+id/guideline4"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 app:layout\_constraintGuide\_begin="20dp"  
 app:layout\_constraintGuide\_percent="0.25"** />  
  
 <**androidx.constraintlayout.widget.Guideline  
 android:id="@+id/guideline5"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 app:layout\_constraintGuide\_begin="20dp"  
 app:layout\_constraintGuide\_percent="0.50"** />  
  
 <**androidx.constraintlayout.widget.Guideline  
 android:id="@+id/guideline6"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 app:layout\_constraintGuide\_begin="20dp"  
 app:layout\_constraintGuide\_percent="0.75"** />  
  
 <**androidx.constraintlayout.widget.Guideline  
 android:id="@+id/guideline7"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 app:layout\_constraintGuide\_begin="20dp"  
 app:layout\_constraintGuide\_percent="0.90"** />  
  
 <**ImageView  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:src="@mipmap/app"  
 android:alpha="0.4"**/>  
  
 <**com.google.android.material.textfield.TextInputLayout  
 android:id="@+id/username"  
 style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/email"  
 android:textColorHint="@color/white"  
 app:hintTextColor="@color/white"  
 android:backgroundTint="@color/white"  
 app:boxStrokeColor="@color/white"  
 app:layout\_constraintTop\_toBottomOf="@id/guideline3"  
 android:layout\_marginLeft="20dp"  
 android:layout\_marginRight="20dp"  
 android:layout\_marginTop="100dp"**>  
  
 <**com.google.android.material.textfield.TextInputEditText  
 android:id="@+id/emailname"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:textColor="@color/white"  
 android:textSize="18sp"  
 android:drawableStart="@drawable/accounticon"  
 android:drawableLeft="@drawable/accounticon"  
 android:textStyle="bold"  
 android:drawablePadding="10dp"  
 android:inputType="textEmailAddress"** />  
  
 </**com.google.android.material.textfield.TextInputLayout**>  
  
 <**com.google.android.material.textfield.TextInputLayout  
 android:id="@+id/pass"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 app:passwordToggleEnabled="true"  
 android:hint="@string/pass"  
 android:backgroundTint="@color/white"  
 android:textColorHint="@color/white"  
 app:hintTextColor="@color/white"  
 style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"  
 app:boxStrokeColor="@color/white"  
 app:layout\_constraintTop\_toBottomOf="@id/username"  
 android:layout\_margin="20dp"  
 app:passwordToggleTint="@color/white"**>  
  
 <**com.google.android.material.textfield.TextInputEditText  
 android:id="@+id/passs"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:textColor="@color/white"  
 android:textSize="18sp"  
 android:drawableLeft="@drawable/passicon"  
 android:textStyle="bold"  
 android:drawablePadding="10dp"  
 android:drawableStart="@drawable/passicon"  
 android:inputType="textPassword"**/>  
  
 </**com.google.android.material.textfield.TextInputLayout**>  
  
 <**Button  
 android:id="@+id/loginbtn"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginStart="20dp"  
 android:layout\_marginEnd="20dp"  
 android:layout\_marginTop="15dp"  
 android:ems="10"  
 android:text="@string/login"  
 android:textColor="@color/white"  
 android:textStyle="bold"  
 android:textSize="25sp"  
 android:background="@drawable/roundbutton"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@id/llayou"**/>  
  
 <**RelativeLayout  
 android:id="@+id/llayou"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 app:layout\_constraintTop\_toBottomOf="@id/pass"  
 app:layout\_constraintStart\_toEndOf="@id/guideline"  
 app:layout\_constraintEnd\_toStartOf="@id/guideline2"  
 android:layout\_margin="20dp"**>  
  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentRight="true"  
 android:text="@string/forgot"  
 android:textColor="@color/white"  
 android:textSize="16sp"  
 android:backgroundTint="#DB7093"  
 android:textStyle="bold"  
 android:id="@+id/forgotpass"**/>  
  
 </**RelativeLayout**>  
  
</**androidx.constraintlayout.widget.ConstraintLayout**>

**JAVA CODE**

**package** com.example.project;  
  
**import** androidx.annotation.NonNull;  
**import** androidx.appcompat.app.AppCompatActivity;  
  
**import** android.app.AlertDialog;  
**import** android.app.ProgressDialog;  
**import** android.content.Context;  
**import** android.content.DialogInterface;  
**import** android.content.Intent;  
**import** android.net.ConnectivityManager;  
**import** android.net.NetworkInfo;  
**import** android.os.Bundle;  
**import** android.util.Patterns;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**import** com.google.android.gms.tasks.OnCompleteListener;  
**import** com.google.android.gms.tasks.Task;  
**import** com.google.android.material.textfield.TextInputEditText;  
**import** com.google.firebase.auth.AuthResult;  
**import** com.google.firebase.auth.FirebaseAuth;  
**import** com.google.firebase.database.DataSnapshot;  
**import** com.google.firebase.database.DatabaseError;  
**import** com.google.firebase.database.DatabaseReference;  
**import** com.google.firebase.database.FirebaseDatabase;  
**import** com.google.firebase.database.ValueEventListener;  
  
**import** java.util.regex.Matcher;  
**import** java.util.regex.Pattern;  
  
**public class** MainActivity **extends** AppCompatActivity {  
 TextInputEditText **lusername**,**lpassword**;  
 Button **lloginbtn**;  
 TextView **lforgot**;  
 ProgressDialog **progressBar**;  
 FirebaseAuth **fAuth**;  
 DatabaseReference **rootref**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
  
 **lusername**=(TextInputEditText)findViewById(R.id.***emailname***);  
 **lpassword**=(TextInputEditText)findViewById(R.id.***passs***);  
 **lloginbtn**=findViewById(R.id.***loginbtn***);  
 **lforgot**=findViewById(R.id.***forgotpass***);  
 **progressBar**=**new** ProgressDialog(MainActivity.**this**);  
  
 **fAuth**= FirebaseAuth.*getInstance*();  
  
 **lforgot**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v)  
 {  
 **if** (!isconnected()){  
 Toast.*makeText*(MainActivity.**this**, **"Please Ensure Internet Connection"**, Toast.***LENGTH\_SHORT***).show();  
 }**else if** (isconnected()){  
 Intent intent=**new** Intent(MainActivity.**this**,changepassword.**class**);  
 intent.addFlags(Intent.***FLAG\_ACTIVITY\_NEW\_TASK***|Intent.***FLAG\_ACTIVITY\_CLEAR\_TASK***);  
 startActivity(intent);  
 }  
 }  
 });  
  
 **lloginbtn**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **final** String email = **lusername**.getText().toString().trim();  
 **final** String pass = **lpassword**.getText().toString().trim();  
  
 **if** (!*validator*(pass)){  
 **lpassword**.setError(**"Minimum 8 Characters,One Alphabetic Character,One Small case Character,One Digit and one special symbol Required"**);  
 **lpassword**.setText(**""**);  
 **lpassword**.requestFocus();  
 }**else if** (email.isEmpty()) {  
 **lusername**.setError(**"Email Required"**);  
 **lusername**.setText(**""**);  
 **lusername**.requestFocus();  
 **return**;  
 }**else if** (!Patterns.***EMAIL\_ADDRESS***.matcher(email).matches()) {  
 **lusername**.setError(**"Enter Valid Email Address"**);  
 **lusername**.setText(**""**);  
 **lusername**.requestFocus();  
 **return**;  
 }**else if** (pass.isEmpty()) {  
 **lpassword**.setError(**"Please enter Password"**);  
 **lpassword**.setText(**""**);  
 **lpassword**.requestFocus();  
 **return**;  
 }**else**{  
 **progressBar**.setTitle(**"Logging In"**);  
 **progressBar**.setMessage(**"Please Wait, While we verify Credentials"**);  
 **progressBar**.setCanceledOnTouchOutside(**false**);  
 **progressBar**.show();  
  
 **fAuth**.signInWithEmailAndPassword(email,pass).addOnCompleteListener(**new** OnCompleteListener<AuthResult>() {  
 @Override  
 **public void** onComplete(@NonNull Task<AuthResult> task) {  
 **if** (task.isSuccessful())  
 {  
 **final** String id;  
 id=**fAuth**.getCurrentUser().getUid();  
  
 **rootref**=FirebaseDatabase.*getInstance*().getReference().child(**"Users"**).child(id);  
  
 **rootref**.addValueEventListener(**new** ValueEventListener() {  
 @Override  
 **public void** onDataChange(@NonNull DataSnapshot dataSnapshot) {  
 **if** (dataSnapshot.exists()) {  
 String usertype, username, usermail;  
 usertype = dataSnapshot.child(**"type"**).getValue().toString();  
 username = dataSnapshot.child(**"Name"**).getValue().toString();  
 usermail = dataSnapshot.child(**"Email"**).getValue().toString();  
  
 **if** (usertype.equals(**"ADMIN"**)) {  
 Intent intent = **new** Intent(MainActivity.**this**, Admin.**class**);  
 intent.addFlags(Intent.***FLAG\_ACTIVITY\_NEW\_TASK*** | Intent.***FLAG\_ACTIVITY\_CLEAR\_TASK***);  
 intent.putExtra(**"usermail"**, usermail);  
 intent.putExtra(**"username"**, username);  
 startActivity(intent);  
 **lusername**.setText(**""**);  
 **lpassword**.setText(**""**);  
 **lusername**.requestFocus();  
 **progressBar**.dismiss();  
  
 } **else if** (usertype.equals(**"STAFF"**)) {  
 Intent intent = **new** Intent(MainActivity.**this**, Staff.**class**);  
 intent.addFlags(Intent.***FLAG\_ACTIVITY\_NEW\_TASK*** | Intent.***FLAG\_ACTIVITY\_CLEAR\_TASK***);  
 intent.putExtra(**"usermail"**, usermail);  
 intent.putExtra(**"username"**, username);  
 startActivity(intent);  
 **lusername**.setText(**""**);  
 **lpassword**.setText(**""**);  
 **lusername**.requestFocus();  
 **progressBar**.dismiss();  
 } **else if** (usertype.equals(**"IT STAFF"**)) {  
 Intent intent = **new** Intent(MainActivity.**this**, Technical.**class**);  
 intent.addFlags(Intent.***FLAG\_ACTIVITY\_NEW\_TASK*** | Intent.***FLAG\_ACTIVITY\_CLEAR\_TASK***);  
 intent.putExtra(**"username"**, username);  
 intent.putExtra(**"usermail"**, usermail);  
 startActivity(intent);  
 **lusername**.setText(**""**);  
 **lpassword**.setText(**""**);  
 **lusername**.requestFocus();  
 **progressBar**.dismiss();  
 }  
 }**else if** (!dataSnapshot.exists()){  
 Toast.*makeText*(MainActivity.**this**, **"User does not Exists,Please Contact Admin ..."**, Toast.***LENGTH\_SHORT***).show();  
 **progressBar**.dismiss();  
 }  
  
 }  
  
 @Override  
 **public void** onCancelled(@NonNull DatabaseError databaseError) {  
  
 }  
 });  
 }  
 **else** {  
 Toast.*makeText*(MainActivity.**this**, **"User does not exists "**, Toast.***LENGTH\_SHORT***).show();  
 **lusername**.setText(**""**);  
 **lpassword**.setText(**""**);  
 **lusername**.requestFocus();  
 **progressBar**.dismiss();  
 }  
 }  
 });  
 }  
 }  
 });  
  
 **if** (!isconnected())  
 {  
 Toast.*makeText*(MainActivity.**this**,**"Please Ensure Internet Connection"**,Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
  
 **public boolean** isconnected()  
 {  
 ConnectivityManager connectivityManager = (ConnectivityManager) **this**.getSystemService(Context.***CONNECTIVITY\_SERVICE***);  
 NetworkInfo networkInfo = connectivityManager.getActiveNetworkInfo();  
 **return** networkInfo != **null**;  
 }  
  
 **public static boolean** validator(String password){  
 Pattern pattern;  
 Matcher matcher;  
 **final** String Pass\_pattern=**"^(?=.\*[0-9])(?=.\*[A-Z])(?=.\*[a-z])(?=.\*[@#!$%^&+=])(?=\\S+$).{8,}$"**;  
 pattern=Pattern.*compile*(Pass\_pattern);  
 matcher=pattern.matcher(password);  
 **return** matcher.matches();  
 }  
  
}

**CHANGE PASSWORD**

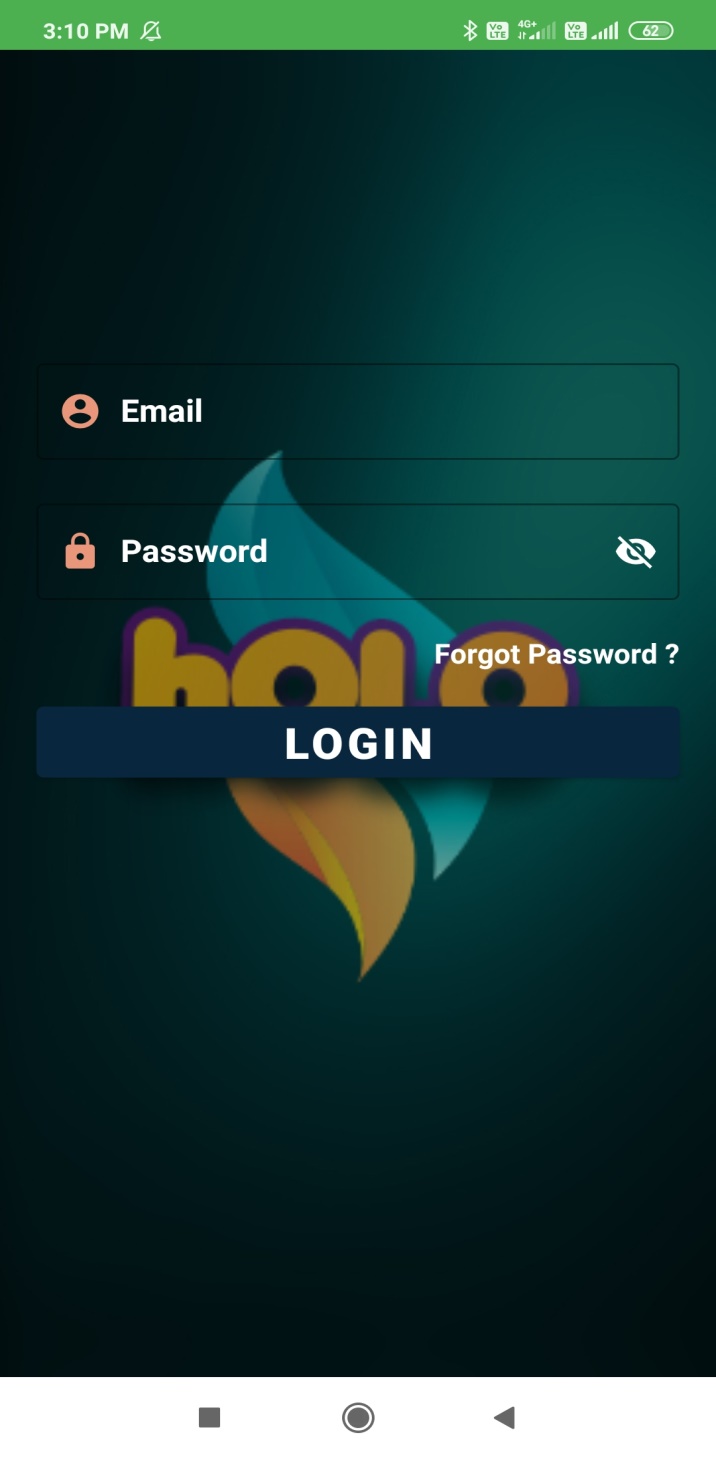
**XML CODE**

*<?***xml version="1.0" encoding="utf-8"***?>*<**androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".changepassword"  
 android:background="@drawable/image"**>  
  
 <**androidx.constraintlayout.widget.Guideline  
 android:id="@+id/guideline"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 app:layout\_constraintGuide\_begin="20dp"  
 app:layout\_constraintGuide\_percent="0.05"** />  
  
 <**androidx.constraintlayout.widget.Guideline  
 android:id="@+id/guideline2"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 app:layout\_constraintGuide\_begin="20dp"  
 app:layout\_constraintGuide\_percent="0.95"** />  
  
 <**androidx.constraintlayout.widget.Guideline  
 android:id="@+id/guideline3"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 app:layout\_constraintGuide\_begin="20dp"  
 app:layout\_constraintGuide\_percent="0.10"** />  
  
 <**androidx.constraintlayout.widget.Guideline  
 android:id="@+id/guideline4"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 app:layout\_constraintGuide\_begin="20dp"  
 app:layout\_constraintGuide\_percent="0.25"** />  
  
 <**androidx.constraintlayout.widget.Guideline  
 android:id="@+id/guideline5"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 app:layout\_constraintGuide\_begin="20dp"  
 app:layout\_constraintGuide\_percent="0.50"** />  
  
 <**androidx.constraintlayout.widget.Guideline  
 android:id="@+id/guideline6"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 app:layout\_constraintGuide\_begin="20dp"  
 app:layout\_constraintGuide\_percent="0.75"** />  
  
 <**androidx.constraintlayout.widget.Guideline  
 android:id="@+id/guideline7"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 app:layout\_constraintGuide\_begin="20dp"  
 app:layout\_constraintGuide\_percent="0.90"** />  
  
 <**TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 app:layout\_constraintTop\_toBottomOf="@id/guideline3"  
 android:id="@+id/resetname"  
 android:text="Reset Password"  
 android:layout\_marginTop="10dp"  
 android:textSize="25sp"  
 android:textStyle="bold"  
 android:textColor="@color/white"  
 android:background="@color/colorPrimary"  
 android:textAlignment="center"**/>  
  
 <**TextView  
 android:id="@+id/textview"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 app:layout\_constraintTop\_toBottomOf="@id/resetname"  
 android:text="@string/resetlinkname"  
 android:textColor="@color/white"  
 android:textSize="18sp"  
 android:textAlignment="center"  
 app:layout\_constraintStart\_toEndOf="@id/guideline"  
 android:textStyle="bold"  
 app:layout\_constraintEnd\_toStartOf="@id/guideline2"  
 android:layout\_marginTop="40dp"  
 android:layout\_marginStart="20dp"  
 android:layout\_marginEnd="20dp"**/>  
  
 <**com.google.android.material.textfield.TextInputLayout  
 android:id="@+id/reset"  
 style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="@string/email"  
 android:textColorHint="@color/white"  
 app:boxStrokeColor="@color/white"  
 app:layout\_constraintTop\_toBottomOf="@id/textview"  
 android:layout\_marginTop="25dp"  
 android:layout\_marginLeft="20dp"  
 android:layout\_marginRight="20dp"**>  
  
 <**com.google.android.material.textfield.TextInputEditText  
 android:id="@+id/resetemail"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:textColor="@color/white"  
 android:textSize="18sp"  
 android:textStyle="bold"  
 android:drawableStart="@drawable/accounticon"  
 android:drawableLeft="@drawable/accounticon"  
 android:inputType="textEmailAddress"**/>  
  
 </**com.google.android.material.textfield.TextInputLayout**>  
  
 <**Button  
 android:id="@+id/done"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 app:layout\_constraintTop\_toBottomOf="@id/reset"  
 android:layout\_marginRight="20dp"  
 android:layout\_marginLeft="20dp"  
 android:text="Send Link"  
 android:textColor="@color/white"  
 android:textStyle="bold"  
 android:backgroundTint="@color/colorPrimary"  
 android:background="@drawable/roundbutton"  
 android:textSize="20sp"  
 android:layout\_marginTop="20dp"**/>  
  
  
</**androidx.constraintlayout.widget.ConstraintLayout**>

**JAVA CODE**

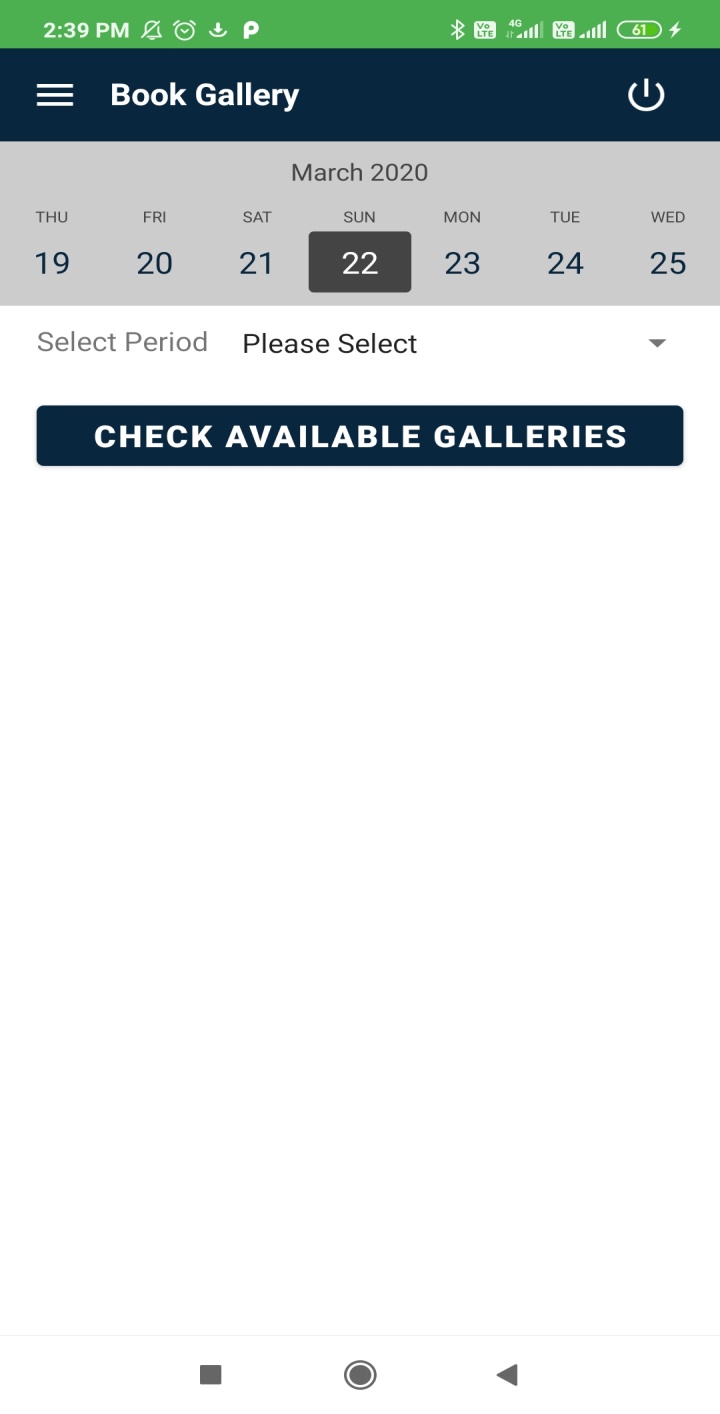
**package** com.example.project;  
  
**import** android.content.Intent;  
**import** android.os.Bundle;  
**import** android.util.Patterns;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.Toast;  
  
**import** androidx.annotation.NonNull;  
**import** androidx.appcompat.app.AppCompatActivity;  
  
**import** com.google.android.gms.tasks.OnCompleteListener;  
**import** com.google.android.gms.tasks.Task;  
**import** com.google.android.material.textfield.TextInputEditText;  
**import** com.google.firebase.auth.FirebaseAuth;  
  
**public class** changepassword **extends** AppCompatActivity {  
  
 TextInputEditText **resetemail**;  
 Button **reset**;  
 FirebaseAuth **fAuth**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_changepassword***);  
  
 **resetemail**=(TextInputEditText)findViewById(R.id.***resetemail***);  
 **reset**=(Button)findViewById(R.id.***done***);  
  
 **fAuth**= FirebaseAuth.*getInstance*();  
  
 **reset**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **final** String email=**resetemail**.getText().toString().trim();  
 **if** (email.isEmpty())  
 {  
 **resetemail**.setError(**"Enter Email Address"**);  
 **resetemail**.setText(**""**);  
 **resetemail**.requestFocus();  
 **return**;  
 }  
 **if**(!Patterns.***EMAIL\_ADDRESS***.matcher(email).matches())  
 {  
 **resetemail**.setError(**"Enter Valid Email Address"**);  
 **resetemail**.requestFocus();  
 **resetemail**.setText(**""**);  
 **return**;  
 }  
  
 **fAuth**.sendPasswordResetEmail(email).addOnCompleteListener(**new** OnCompleteListener<Void>() {  
 @Override  
 **public void** onComplete(@NonNull Task<Void> task) {  
 Toast.*makeText*(changepassword.**this**,**"Reset Link Sent to Email Address"**, Toast.***LENGTH\_SHORT***).show();  
 **resetemail**.setText(**""**);  
 Intent intent=**new** Intent(changepassword.**this**,MainActivity.**class**);  
 intent.addFlags(Intent.***FLAG\_ACTIVITY\_NEW\_TASK***|Intent.***FLAG\_ACTIVITY\_CLEAR\_TASK***);  
 startActivity(intent);  
 }  
 });  
 }  
 });  
 }  
}

**5. IMPLEMENTATION**

****

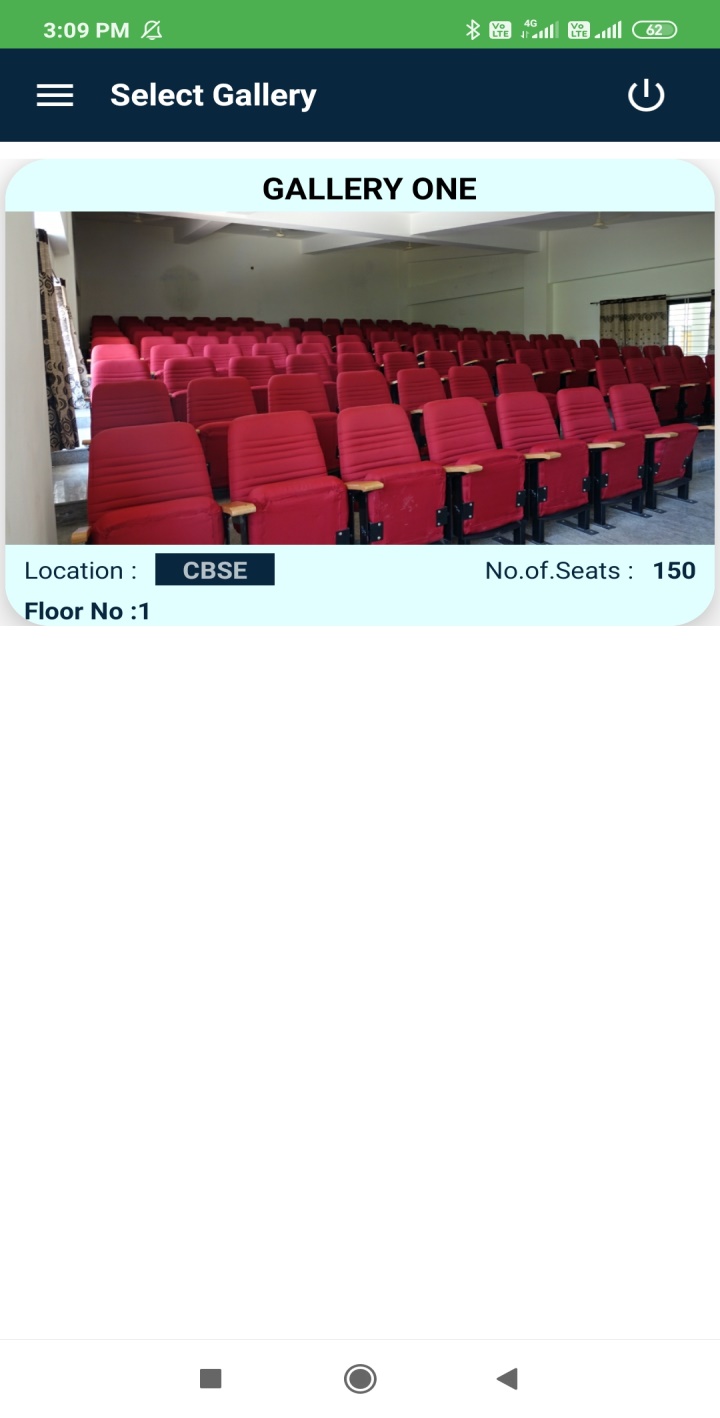
**Fig 5.1 LOGIN PAGE**

The entire user group login to the application using their Credentials provided to them.

****

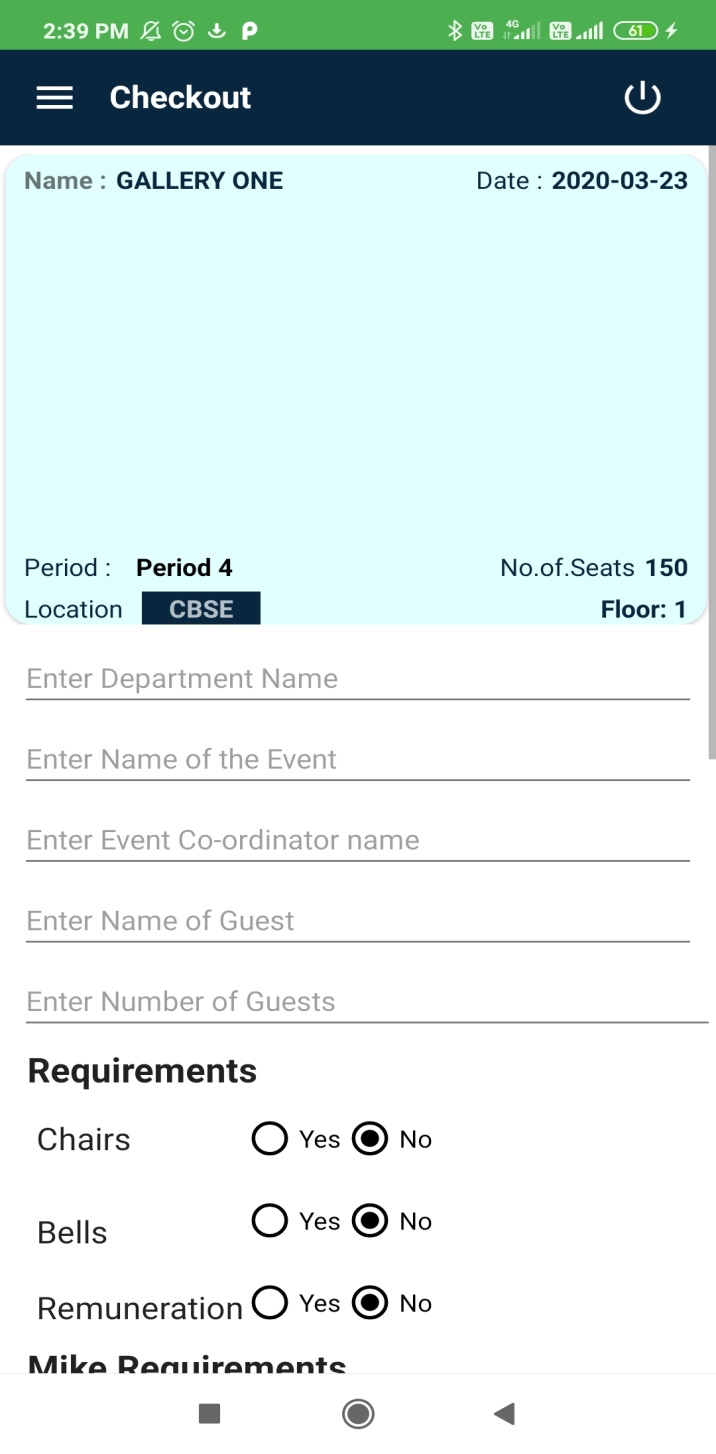
**Fig 5.2 BOOK GALLERY PAGE 1**

All the users select date and respected period for booking an event and once the users click on “CHECK AVAIALABLE GALLERIES” the application redirects to another page to select the Auditorium.

****

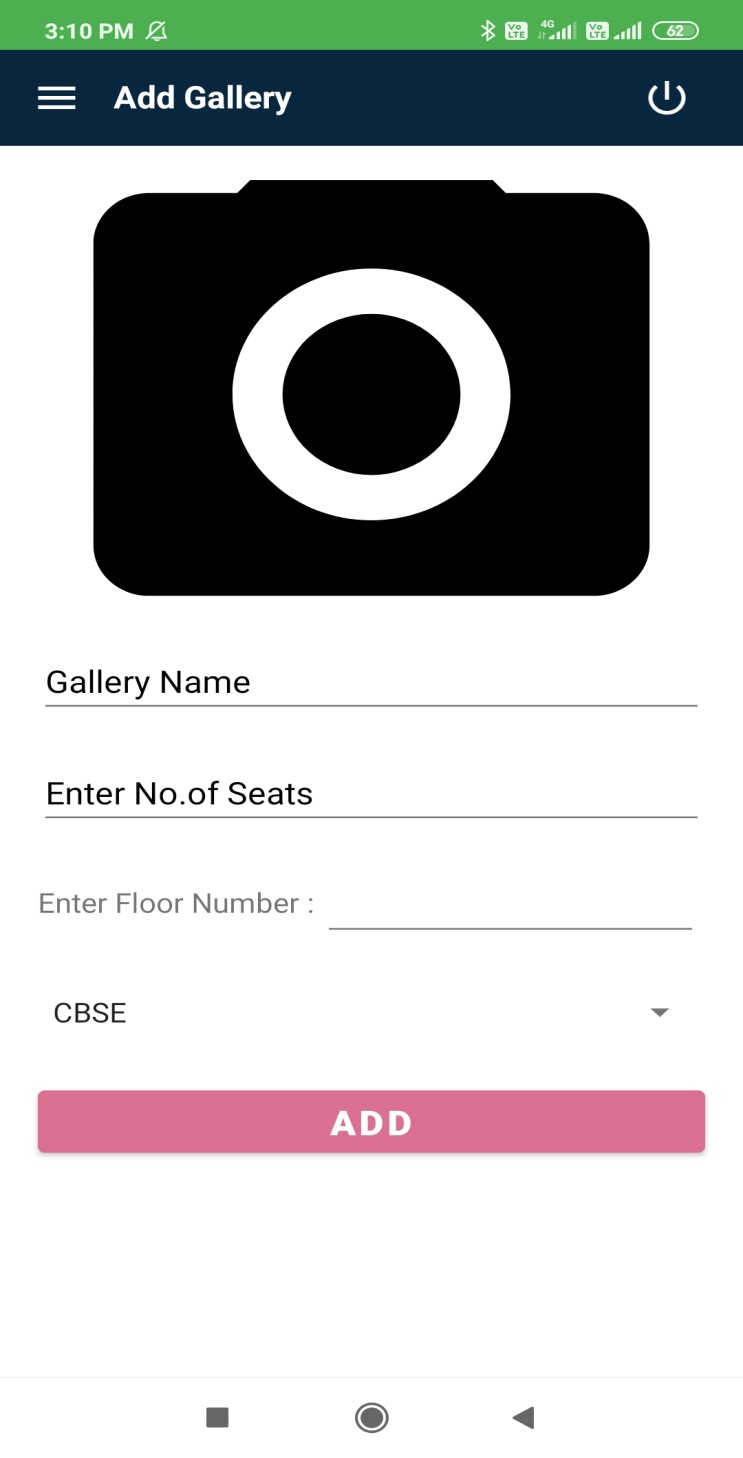
**Fig 5.3 BOOK GALLERY PAGE 2**

This is where the users select Auditoriums. Once the users clicks on Galleries.it checks whether the gallery is available during the date and period that is selected by the user. If the auditorium is available it is re-directed to another page else if not available an error message is shown.

****

**Fig 5.4 BOOK GALLERY PAGE 3**

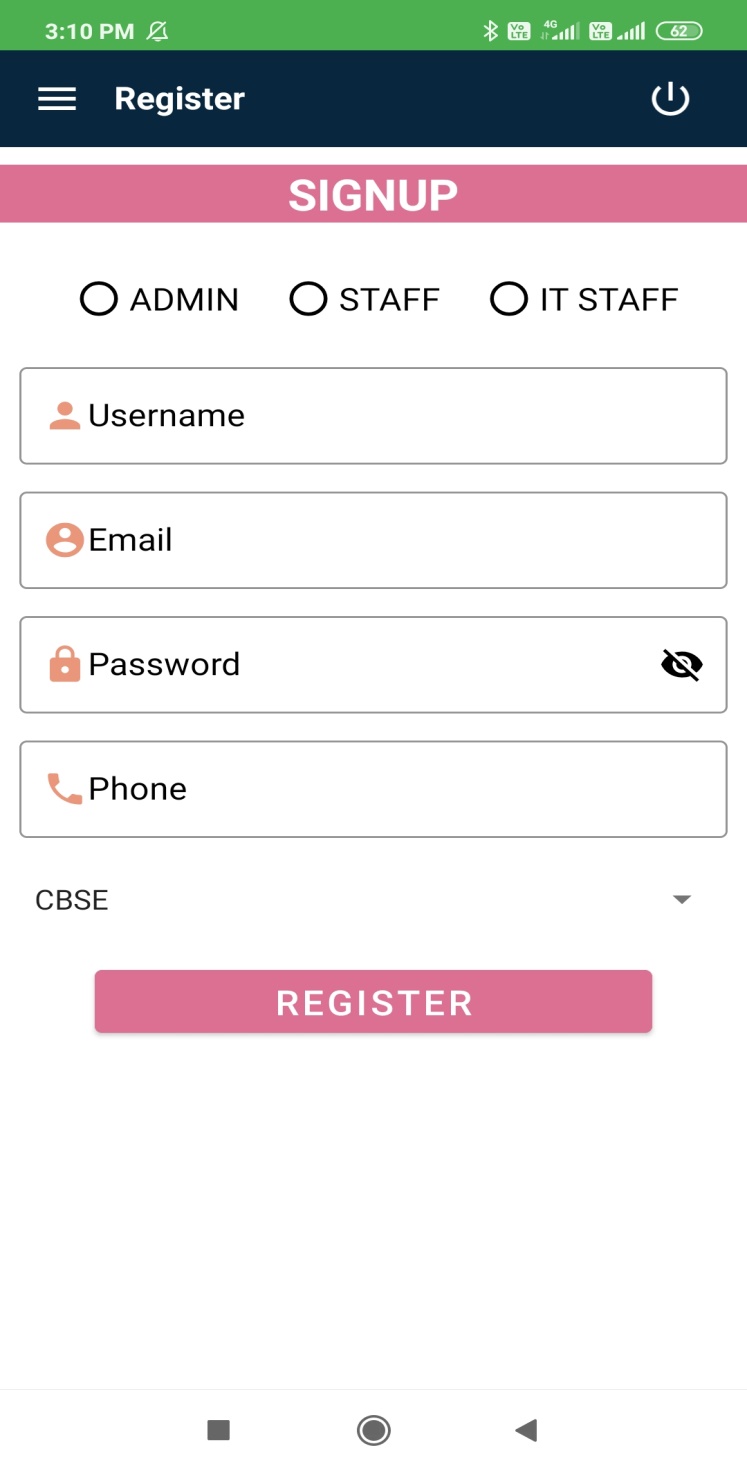
This is the final page for booking an Auditorium. Here, the user is required to enter the basic event details and the necessary requirements of the Auditoriums.

****

**Fig 5.5 ADD GALLERY PAGE**

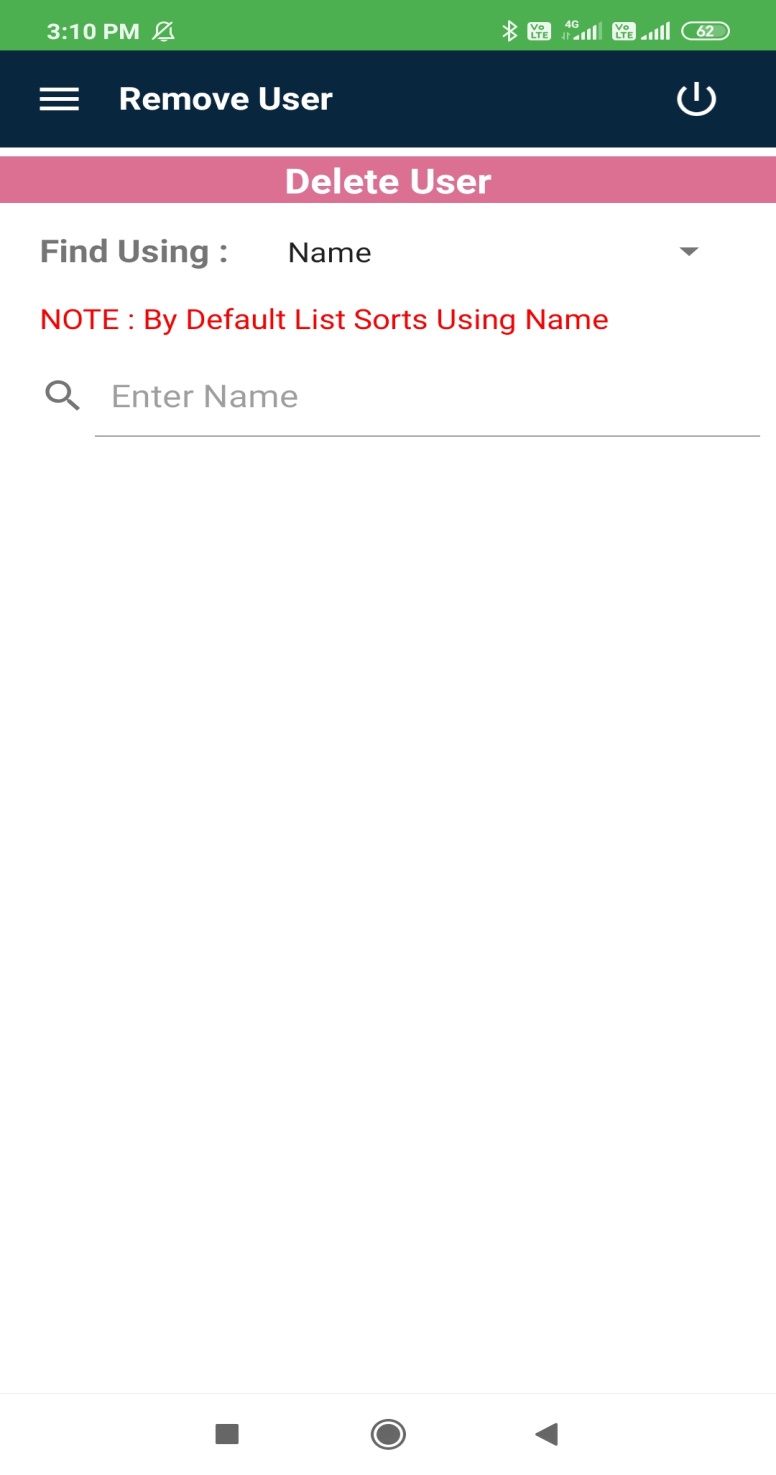
This page is used to add a new Auditorium into the Database. It gathers basic information of the

Auditorium.

****

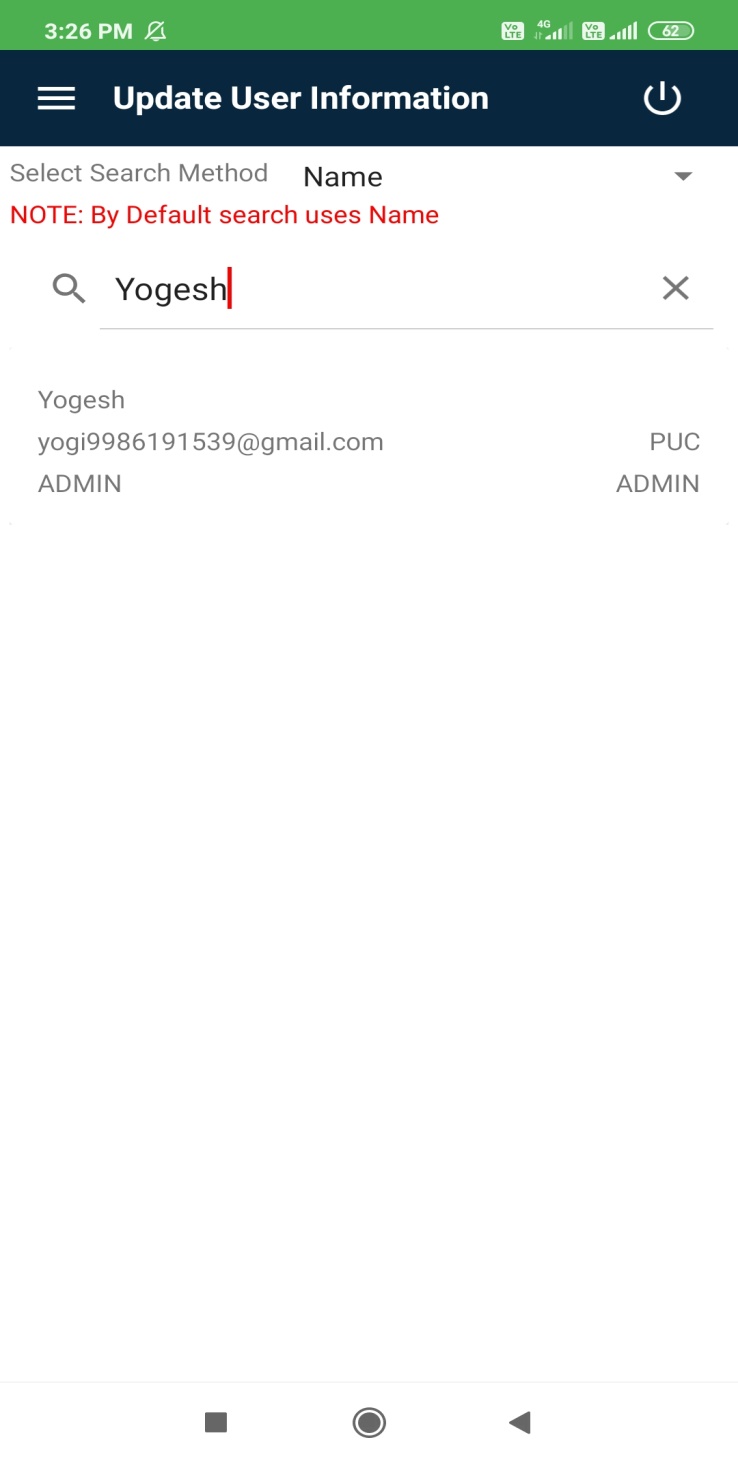
**Fig 5.6 ADD USER PAGE**

This page is used to add new Users to the database.

****

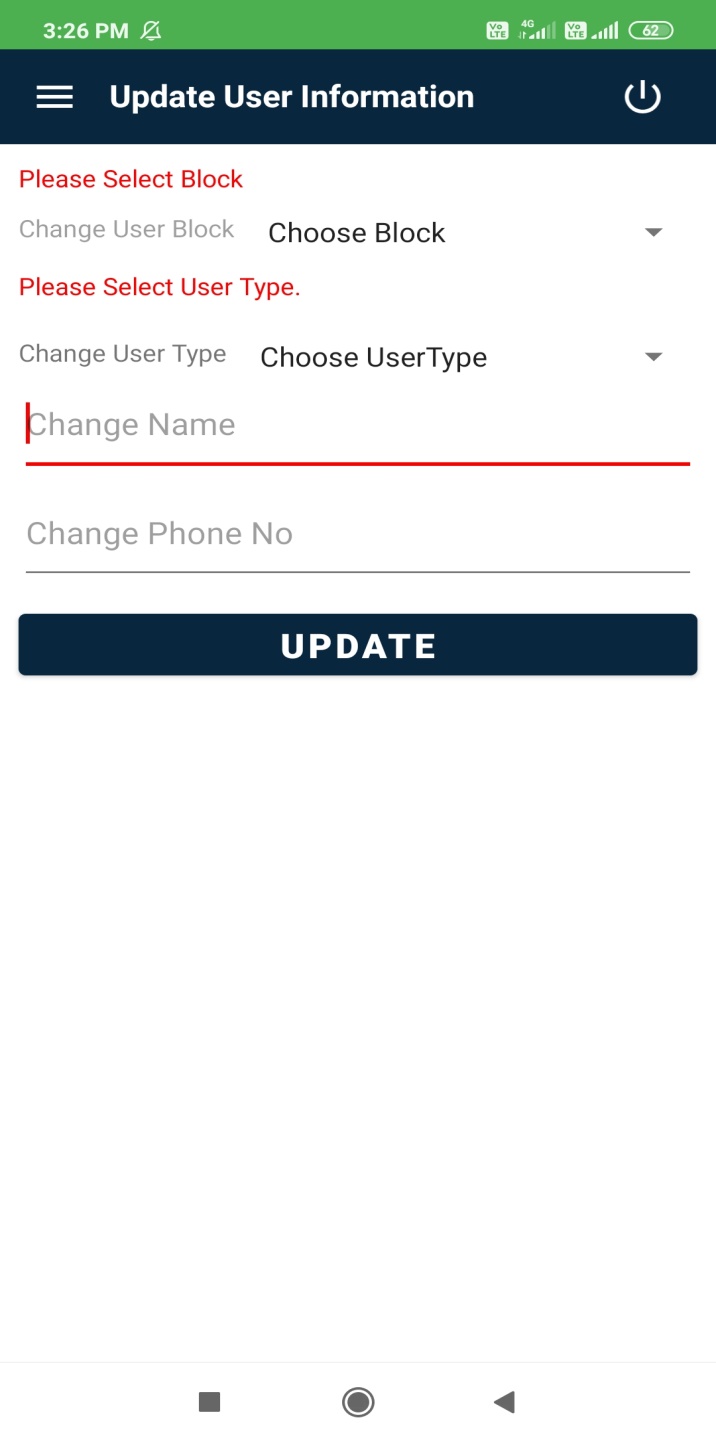
**Fig 5.7 REMOVE USER PAGE**

This page is used to search / traverse all the user’s in the database and then delete the User Information by selecting the desired record that is displayed.



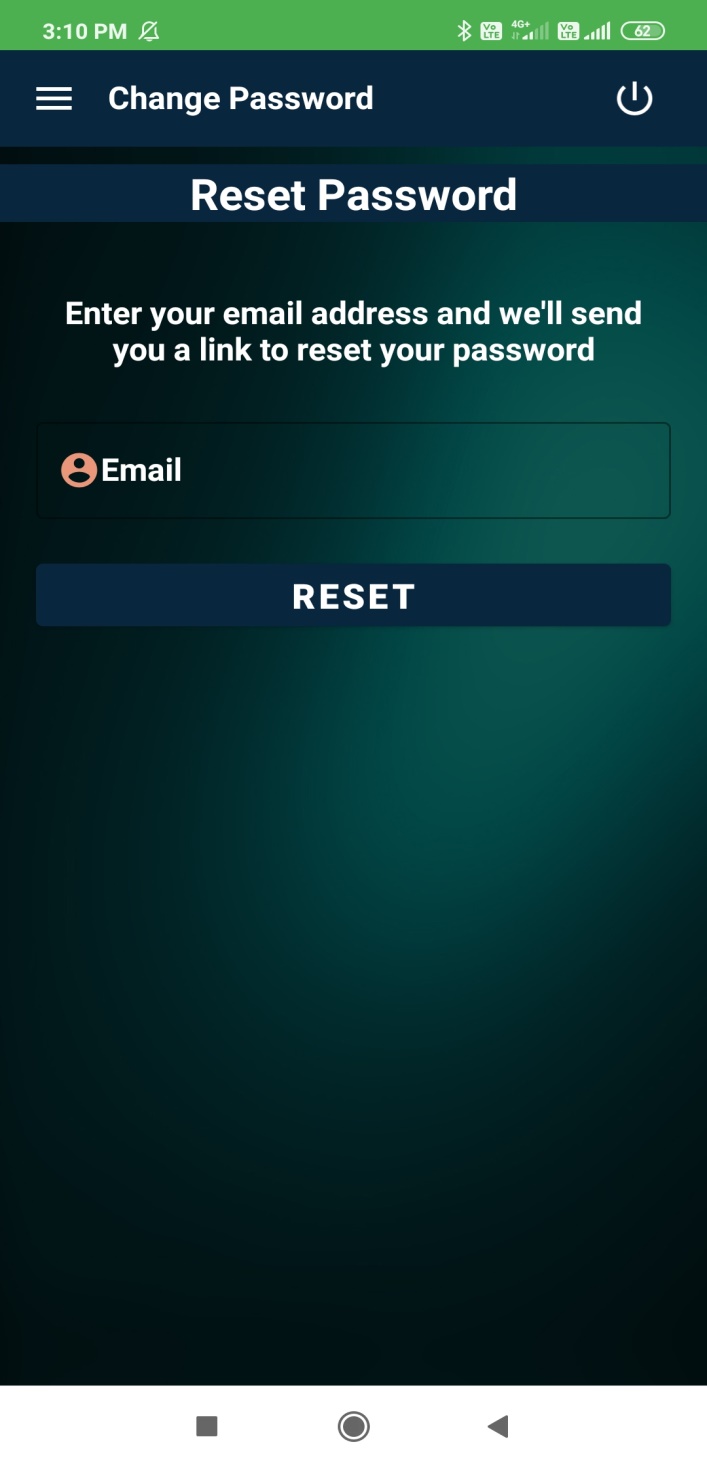
**Fig 5.8 UPDATE USER INFORMATION PAGE 1**

This page initially starts with traversing all the database users and displaying the records. Once the records are found the user clicks on the required user’s information. On clicking the information the Admin is redirected to a new page.



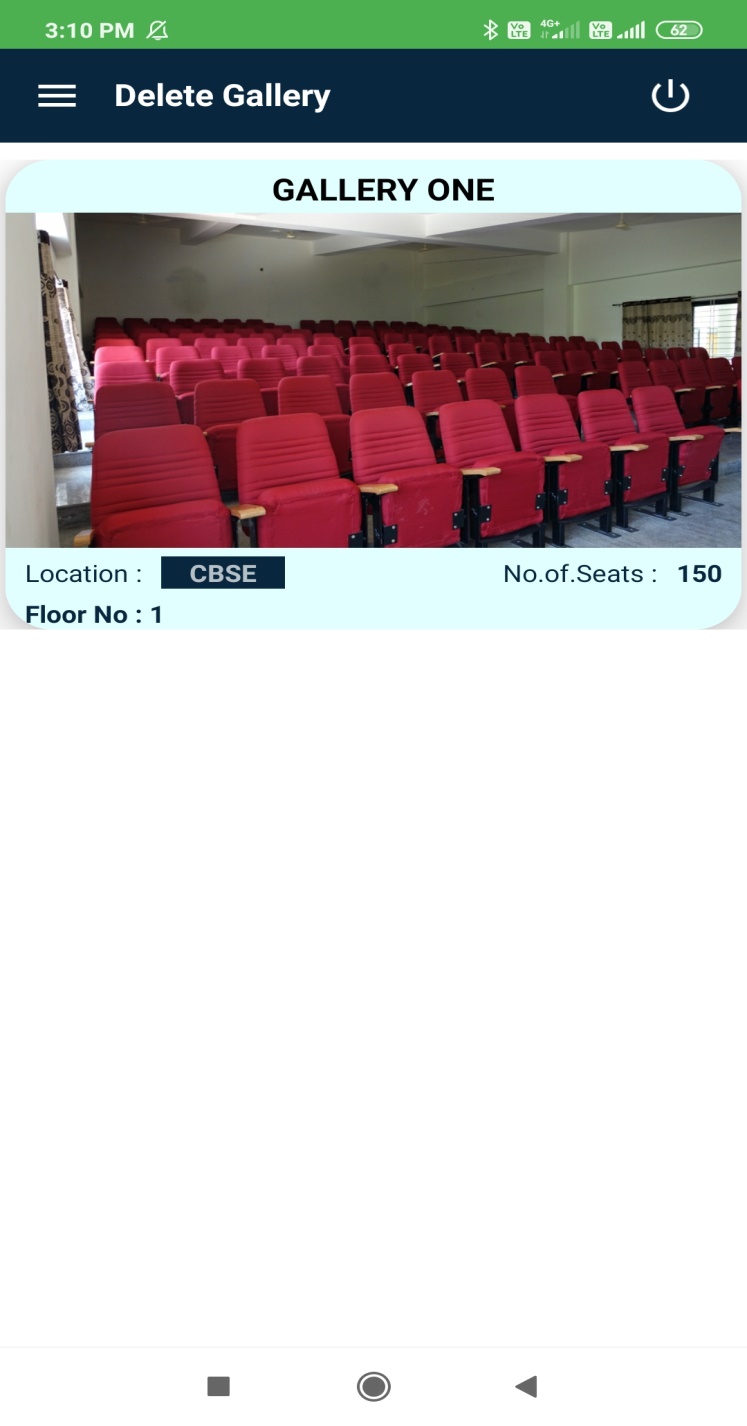
**Fig 5.9 UPDATE USER INFORMATION PAGE 2**

In this page the Admin enters /selects changes to be made to the user selected in the previous page.

****

**Fig 5.10 CHANGE PASSWORD**

Here, the user is required to enter his/ her own email address. Once the email is entered and the users click on “RESET”. The user will receive an email with a link to create new password.

****

**Figs 5.11 DELETE GALLERY PAGE**

In this page all the Auditoriums in the database are listed. The Admin clicks on the Auditorium to be deleted. On clicking the gallery a dialog appears requesting permission to delete the Auditorium permanently from the database.

**6. TESTING**

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not.

Testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

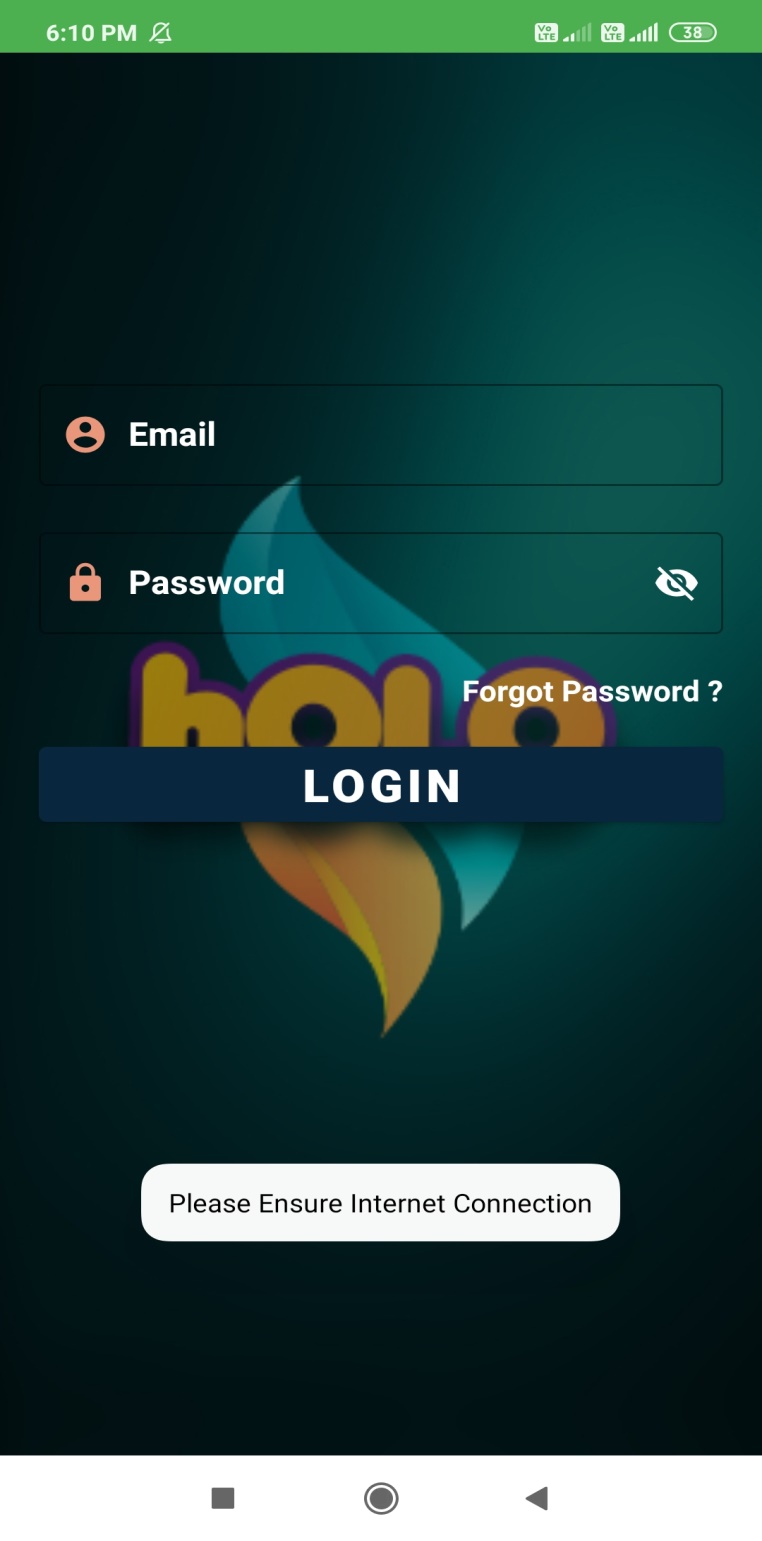
This tutorial will give you a basic understanding on software testing, its types, methods, levels, and other related terminologies.

**6.1 UNIT TESTING**

In [computer programming](https://en.wikipedia.org/wiki/Computer_programming), unit testing is a [software testing](https://en.wikipedia.org/wiki/Software_testing) method by which individual units of [source code](https://en.wikipedia.org/wiki/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.

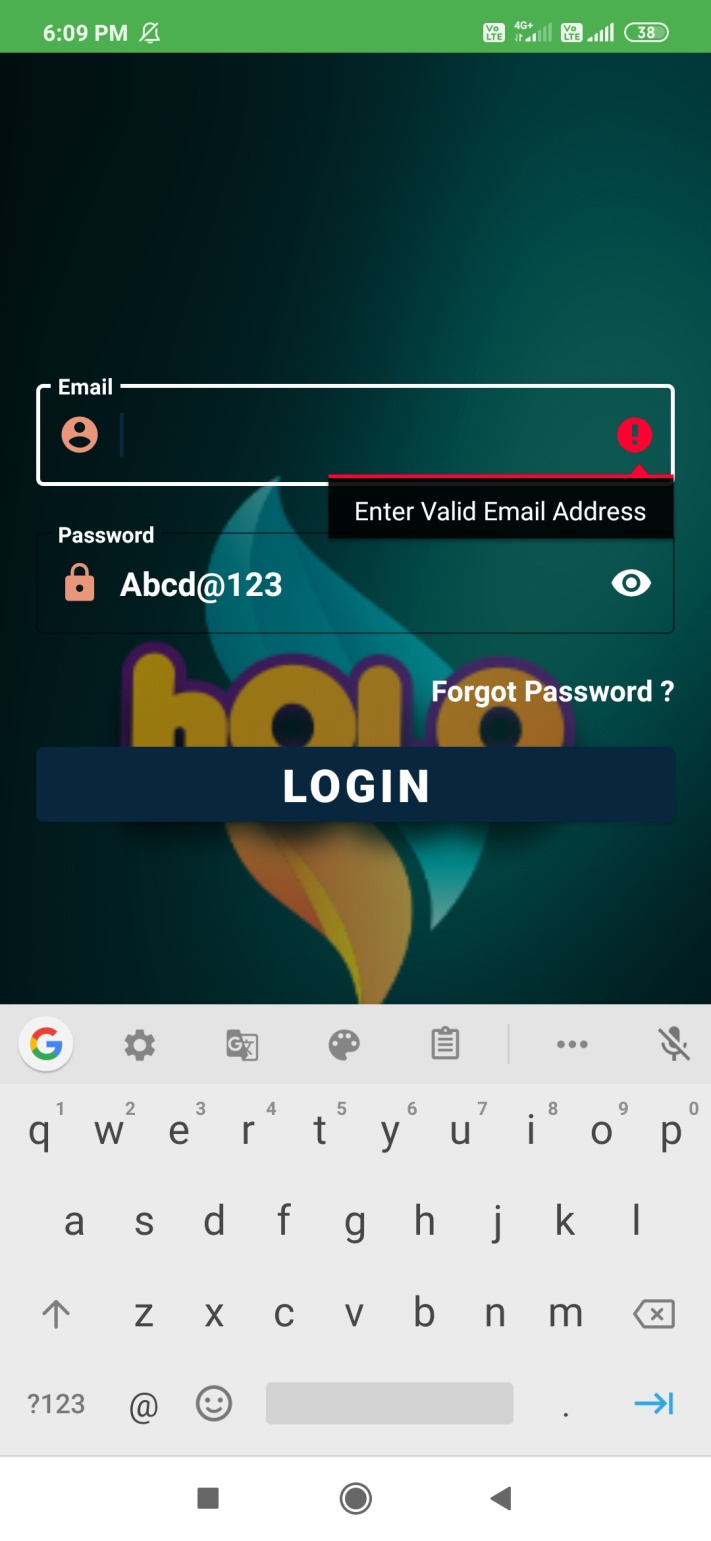
**6.2 INTEGRATION TESTING**

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. Integration testing is conducted to evaluate the compliance of a system or component with specified functional requirements. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.



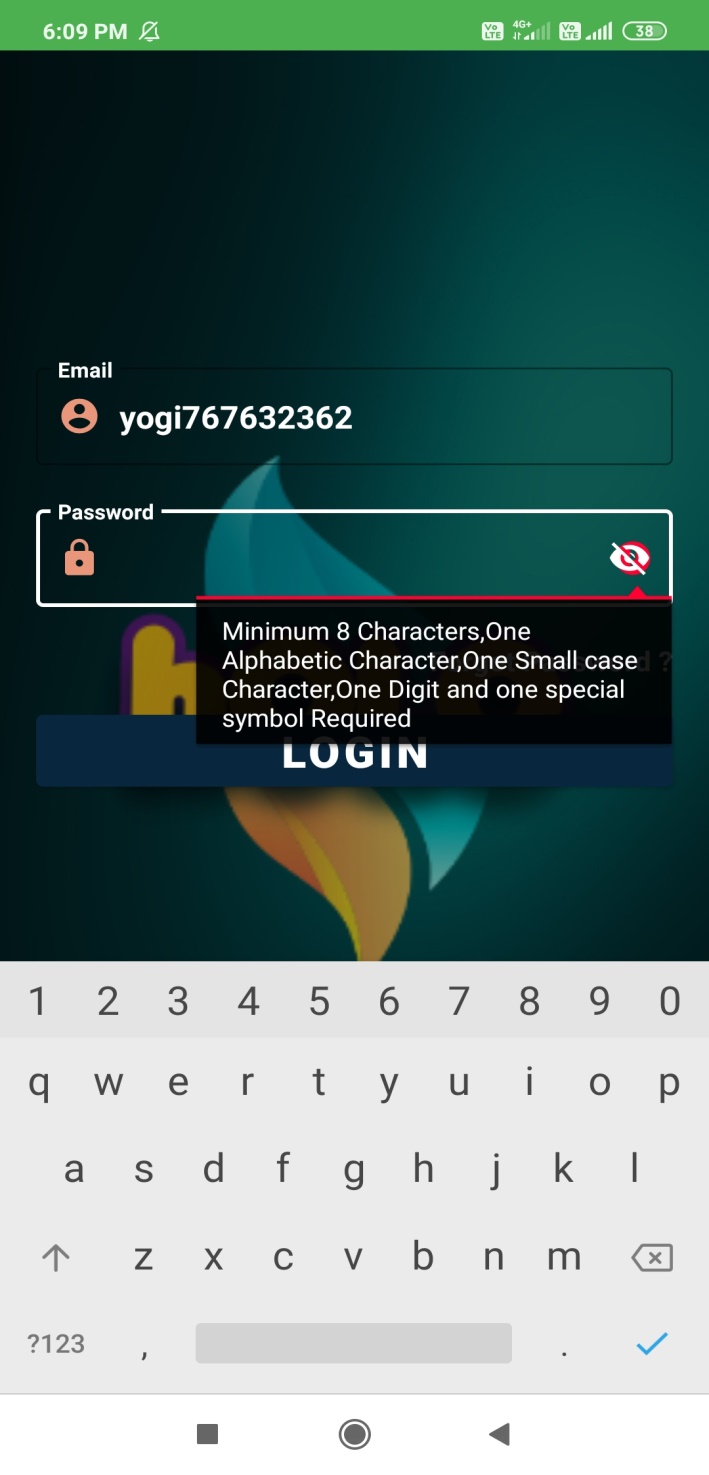
**Fig 6.1 LOGIN PAGE**

This page is the beginning of the application. The application shows an error message if internet connection is not established.



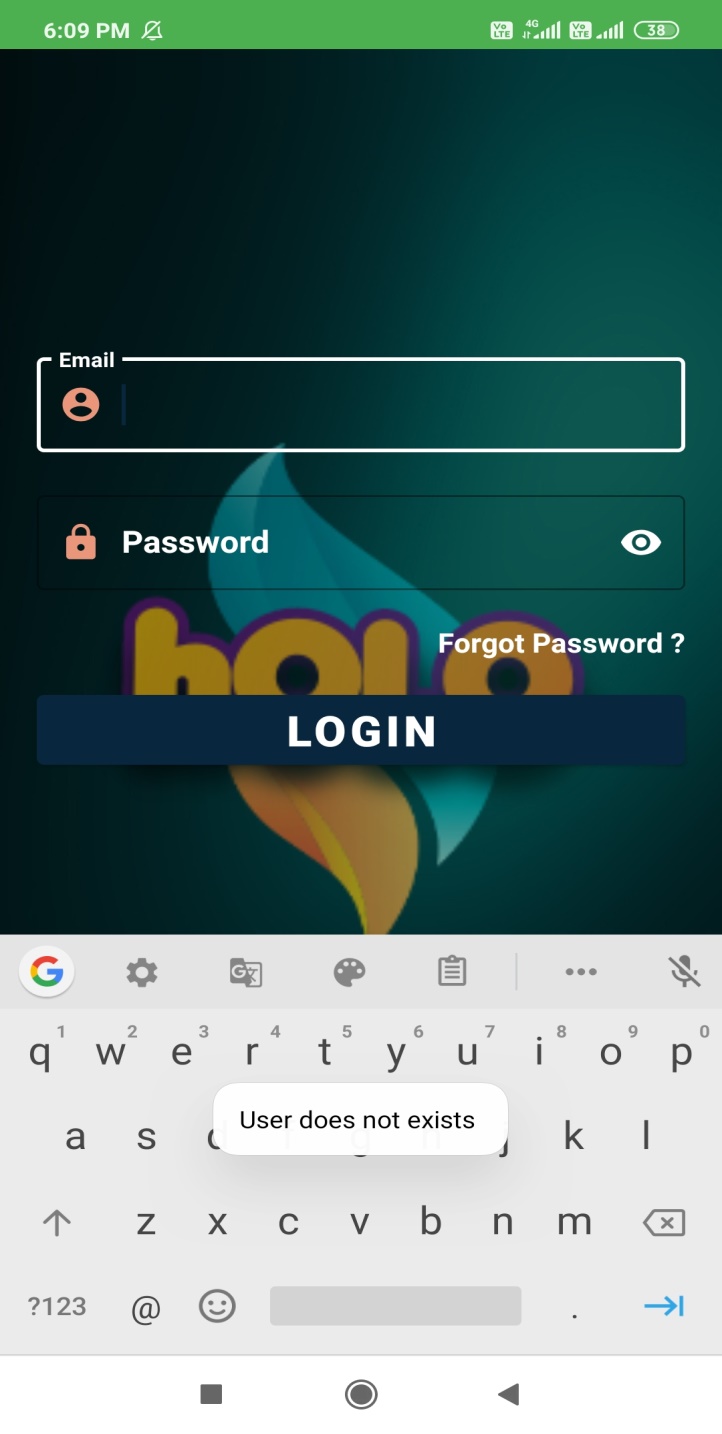
**Fig 6.2 LOGIN PAGE 2**

This page shows an error if email address is not entered in a correct format.



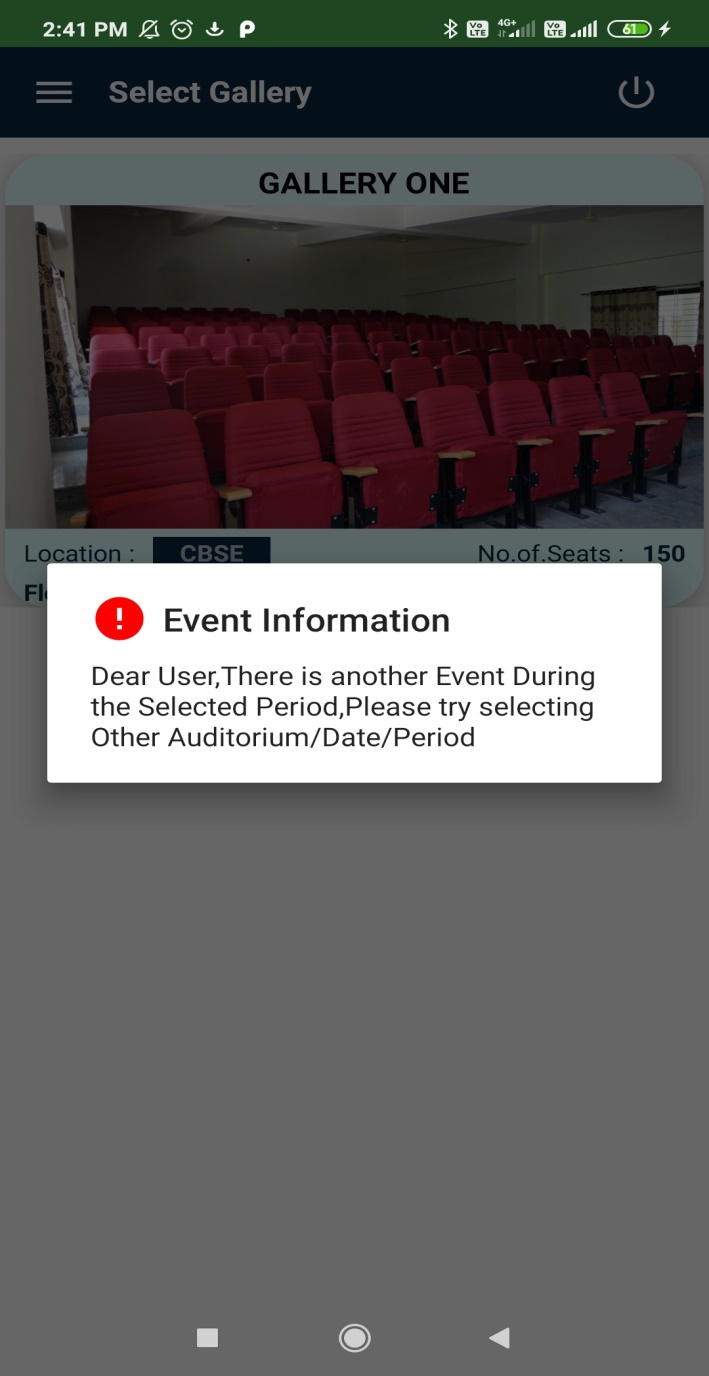
**Fig 6.3 LOGIN PAGE 3**

This page shows error if password does not meet the conditions or requirements specified.



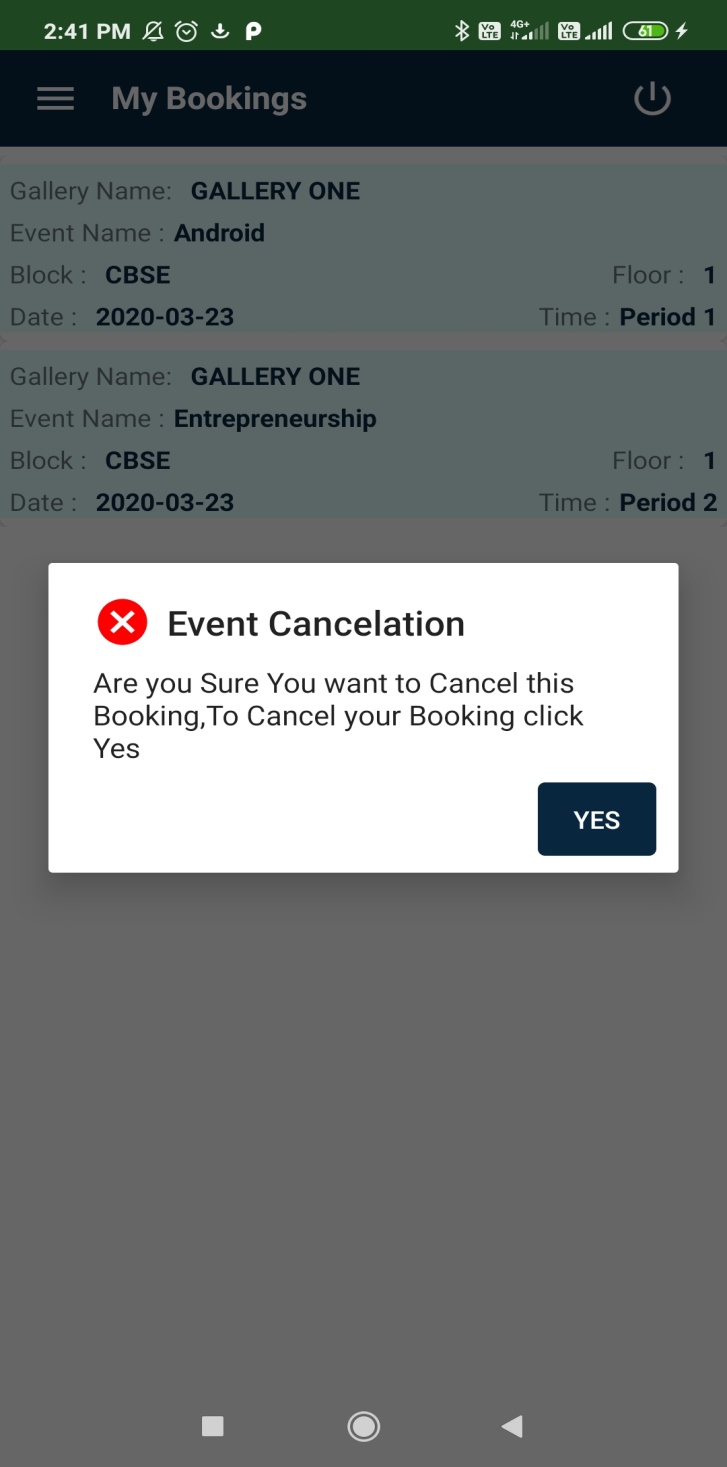
**Fig 6.4 LOGIN PAGE 4**

This page shows error if there is not record with respect to the entered credentials.



**Fig 6.5 BOOK GALLERY PAGE**

This page shows error on clicking the Auditoriums if an event is already running during the selected slot.



**Fig 6.6 MY BOOKINGS PAGE**

This page lists all the bookings done by the user. on clicking the event a dialog as in the image appears requesting or asking the user’s permission to cancel the event.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl. No** | **Description** | **Expected Result** | **Actual Result** | **Status** |
| 1 | Wrong user name and password entered by a person in login. | System must display error message. | System displayed error message and login failed. | Pass |
| 2. | Numeric fields are filled with characters. | System must not accept characters | System is not accepting characters | Pass |
| 3. | String data fields are filled with numbers. | System must not accept numbers | System is not accepting numbers | Pass |
| 4. | Changes made should be reflecting the database. | Update operation should be efficient. | Update operation is performed based on the fields changed by the admin. | Pass |
| 5. | Check Email-id are filled in proper format or Not. | System must not accept invalid email | The text box is cleared if invalid and error appears | Pass |
| 6. | More than 10 numbers are filled in phone number field | System must not accept | System does not accept more than 10 values | Pass |
| 7. | No Internet Connection | Application must display error | A Message appears telling “Please Ensure Internet Connection” | Pass |
| 8. | Event booking | Application must display error if events already exists | An error message is displayed on clicking the auditorium | pass |

**7. CONCLUSION**

* Easy Booking.
* Reduces Manual Booking and improves customer satisfaction.
* Enables faster performance thus helping the management reduce their operation cost.

**8. FUTURE ENHANCEMENTS**

* Improve bookings facility.

**9. BIBILIOGRAPHY**

* Stackoverflow.com
* Tutorialspoint.com
* Tps://github.com/alihussania
* Tps://github.com/jhonnyx2012
* “Learn Android Studio Build Android Apps Quickly and Efficiently “ By Adam Gerber and Clifton Craig.