

CERTIFICATE 1

CERTIFICATE 2

REPORT OF TRAINING AT DEVIANT STROKES

NAME: REUBEN CHAGAS FERNANDES

INSTITUTION: AGNEL POLYTECHNIC VERA

COMPANY: DEVIANT STROKES

MENTOR: AMRITA NERURKAR

ACKNOWLEDGEMENT

I am Reuben Chagas Fernandes student of Agnel Polytechnic Verna who would like to thank the Directorate of technical education goa for introducing me towards this internship program in the final year. It has been really wonderful. A special thanks to Simiao D Cunha principal of Agnel Polytechnic Verna and Sir Savio D'souza (HOD) and all the faculty members of the computer department of Agnel Polytechnic verna for making this internship training program a huge success the guidance and help offered by the faculty at the training will never be forgotten.

The new topics and the mentors will not be forgotten for their sincere help and guidance to learning new things. Above all I would like to give my sincere gratitude to Mr Anand Lotlikar the founder of Deviant Strokes. A huge thanks and blessing to the entire staff for helping me to complete my training.

Last but not the least I thank my parents and my fellow classmates for helping and cooperating with me during the training through which I was able to complete the assignment and tasks assigned to me and I'm happy to have been placed here for my training and will remember deviant strokes even then.

PREFACE

As part of the Diploma Course in Computer Engineering conducted by the Goa Board of technical Education Porvorim , Goa so as to fulfill the obligation that I have completed the 8 weeks training at Deviant Strokes .

The company deals with Corporate Brand Visuals, Marketing Collaterals, Websites , In-Store Branding, Web Applications, CRM (Web Based), Logo, Brochures, Brand Strategies, Social Media Marketing, SEO, and Digital Marketing. The main part of the training was an exposure to the industrial side , wherein we are trained on how certain things work while going for a job in an industry . We were also told to maintain and respect a good relation with our senior staff at the company. During the training I've given to the knowledge I have received during my training and have mentioned it in the following pages below . During this training I have learnt to develop apps using android studio and have also learnt some new web technologies like AJAX.

Table of contents:

Chapter 1: About Deviant Strokes

Chapter 2: Organisational Chart

Chapter 3: Description of the Departments in which students have undergone training

**Chapter 4: description of special jobs undertaken/observed by students
Department wise**

1. Creating an App using Android Studio

2. Creating a web page and connecting it to server and App

**Chapter 5 : unique features identified by students in the training
organisation if any**

**Chapter 6 : knowledge and skill gained by the students during training
period**

Chapter 7: Training summary

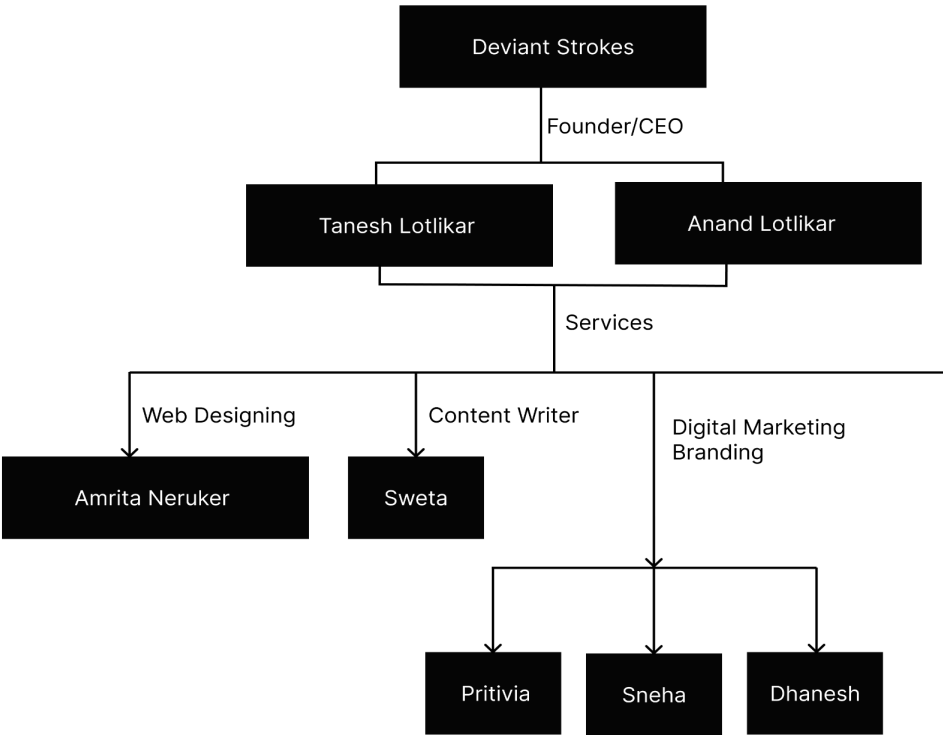
Chapter 1

Deviant Strokes

Deviant strokes is an advertising service founded in 2013 by Anand Lotlikar and Tanesh Lotlikar. It is located in Margao , Goa. It is a self financed organisation. They work with startups & companies to create, elevate & position their brand and they love to create new stuff. At Deviant Strokes they provide Corporate Brand Visuals, Marketing Collaterals, Websites , In-Store Branding, Web Applications, CRM (Web Based), Logo, Brochures, Brand Strategies, Social Media Marketing, SEO, and Digital Marketing

Chapter 2

Organisation Structure



Chapter 4: description of special jobs undertaken/observed by students

Department wise

Taftek systems is an app which is used to control Led lights through bluetooth or through the internet

AndroidManifest.xml

The Android manifest file helps to declare the permissions that an app must have to access data from other apps. The Android manifest file also specifies the app's package name that helps the Android SDK while building the app

These are the list of permissions the AndroidManifest file contains

INTERNET

Allows applications to open network sockets.

BLUETOOTH

Allows applications to connect to paired bluetooth devices.

ACCESS_COARSE_LOCATION

This Permission allows an app to access an approximate location.

ACCESS_FINE_LOCATION

This permission allows u Allows an app to access precise location.

BLUETOOTH_ADMIN

Allows applications to discover and pair bluetooth devices.

BLUETOOTH_SCAN

Required to be able to discover and pair nearby Bluetooth devices.

BLUETOOTH_ADVERTISE

Required to be able to advertise to nearby Bluetooth devices.

BLUETOOTH_CONNECT

Required to be able to connect to paired Bluetooth devices.

ACCESS_NETWORK_STATE

Allows applications to access information about networks

Links.java

This file is an interface which contains all the links to the php file. Each of these Php Files are used to communicate with the app and the server. We use android Volley an HTTP library that makes networking very easy and fast, for Android apps

public interface Links {

```
String InsertData = "http://tafteksystems.com/insertdata.php";  
String ReadData = "http://tafteksystems.com/readlogin.php";  
String InsertMachine ="http://tafteksystems.com/insertmachine.php";  
String UpdateMachine = "http://tafteksystems.com/updatemachine.php";  
String ReadMachine = "http://tafteksystems.com/readmachine.php";  
String UpdateData = "http://tafteksystems.com/updateprofile.php";  
String DeleteData = "http://tafteksystems.com/deletprofile.php";  
String UpdateImage = "http://tafteksystems.com/updateimage.php";  
}
```

The database class uses this interface

Mainactivity.java (Main Page)



When we open the app the main activity page is displayed. The main activity Page displays two buttons on the screen and asks the user to select an option the user can choose **Automation** or **Home**. Then we Access the shared Preferences

```
SharedPreferences sp = getSharedPreferences("User", Context.MODE_PRIVATE);  
loginInfo = sp.getBoolean("log",false);
```

Shared Preferences are used to store data locally and are available across different activities. Here we are finding a Shared preferences User which is Stored securely using MODE_PRIVATE.

Variable LoginInfo will store a boolean of true and false if log of shared preferences does not return any value then by default the variable logininfo will be set to false If the Home or Automation button is pressed a function will be executed which will check if logInfo is true which means user data exists locally and then it will create an Intent. If logInfo is false it will take u to **login.class** instead of **HomeDetails.class**

```
Intent homepage = new Intent(MainActivity.this, HomeDetails.class);
```

```
startActivity(homepage);  
finish();
```

Intents are used for navigating among various activities within the same application. Here a new Intent is created to open another activity **HomeDetails.class** then **startActivity()** executes this intent and then opens a new activity. The **finish()** function is executed when the user clicks on the back button. When the user clicks the back button the current activity will be destroyed and user will be taken back to your previous activity

HomeDetails.java

This file contains the code for the navbar by default it displays the **Home.class** which is Home.java. The navbar contains options like profile , connect , settings and logout

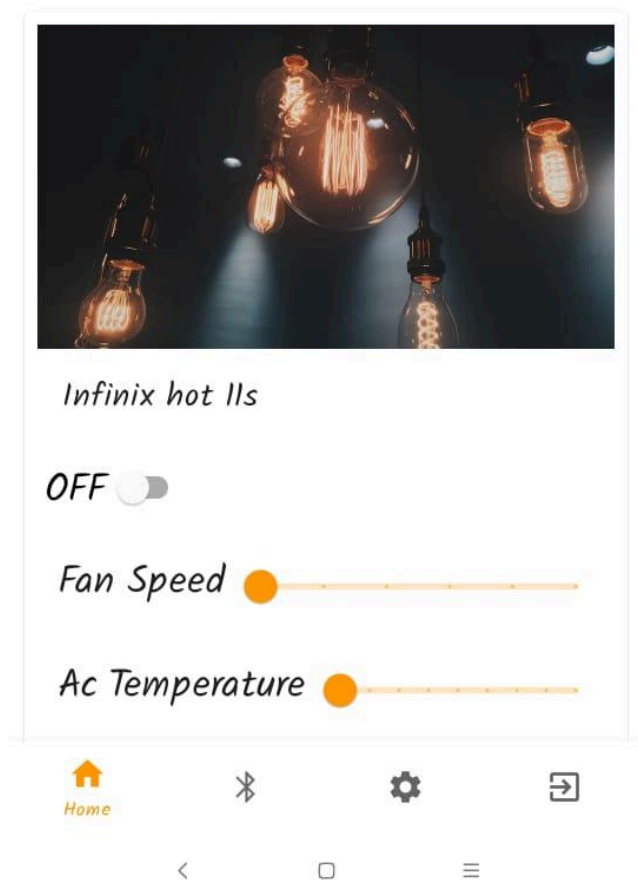
```
Home home = new Home();  
getSupportFragmentManager().beginTransaction().setCustomAnimations(an  
droidx.navigation.ui.R.animator.nav_default_enter_anim,  
androidx.navigation.ui.R.animator.nav_default_exit_anim).replace(R.id.frag  
mentContainerView,home).commit();
```

A Fragment is a piece of an application's user interface or behaviour that can be placed in an Activity. In the code above It first Creates a new object to the Home class then uses **getSupportFragmentManager()** to display the fragment Home. An if Statement has been added to code if the user selects another option then the fragment belonging to that option is opened

Home.java



Paired Devices



This fragment contains all the details of the user. It also shows the users name and profile picture it gets the users profile name from the shared Preferences

```
Picasso.get().load(UProfile.toString()).networkPolicy(NetworkPolicy.NO_CACHE).memoryPolicy(MemoryPolicy.NO_CACHE).transform(new CircleTransform()).into(profile);
```

The above code is used to fetch the user's profile from the server and display it to the user. Picasso is a powerful image downloading and caching library for Android. Then **ReadMachine()** from Database class is called and then ReadMachine() calls a function **Showinfo()** which is in Home class

```
ArrayList<String> Name = new ArrayList<>();  
ArrayList<String> Add = new ArrayList<>();  
ArrayList<String> Status = new ArrayList<>();  
System.out.println(jsonArray);  
for (int i=0;i<jsonArray.length();i++)  
{  
    try {  
        Name.add(jsonArray.getJSONObject(i).getString("devicename"));  
        Add.add(jsonArray.getJSONObject(i).getString("macid"));  
        Status.add(jsonArray.getJSONObject(i).getString("status"));  
    } catch (Exception e) {  
        e.printStackTrace();  
    }  
}
```

The **Showinfo()** function contains an ArrayList of Name and Add and Status. An ArrayList class uses a dynamic array for storing the elements. It is like an array, but there is no size limit. We can add or remove elements anytime. jsonArray contains all the data in a Json format and gets each devicename , mac id and status and using the .add function adds it to the Arraylist after adding it to the Arraylist a Recyclerview is used to display the data. A RecyclerView is a ViewGroup added to the android studio as a successor of the GridView and ListView. We then call the BluetoothHome class and then pass the Arraylist values to the constructor

BluetoothHome.java

This class is used to display all the Bluetooth Devices belonging to the user. It displays all the data in a recyclerview

It first calls **BluetoothAdapter.getDefaultAdapter()**; to get some bluetooth information of the user's phone. In the recyclerview it shows all the device names

and their status whether it is true or false. When the user presses on the switch a function is invoked and will update the status of the switch then it will get the mac id of this device name.

```
bluetoothAddress = holder.status.getTag().toString();
```

The BluetoothAddress variable contains the status (true or false).

BluetoothDevice

```
hc=myBluetooth.getRemoteDevice(bluetoothAddress.replaceAll(" ",""));
```

Variable hc store the mac address of another bluetooth device

```
btSocket = hc.createInsecureRfcommSocketToServiceRecord(myUid);
```

```
btSocket.connect();
```

Btsocket will create a socket connection to that device

```
if(btSocket!= null) {
```

```
    btSocket.getOutputStream().write(info.toString().getBytes());
```

```
    btSocket.close();
```

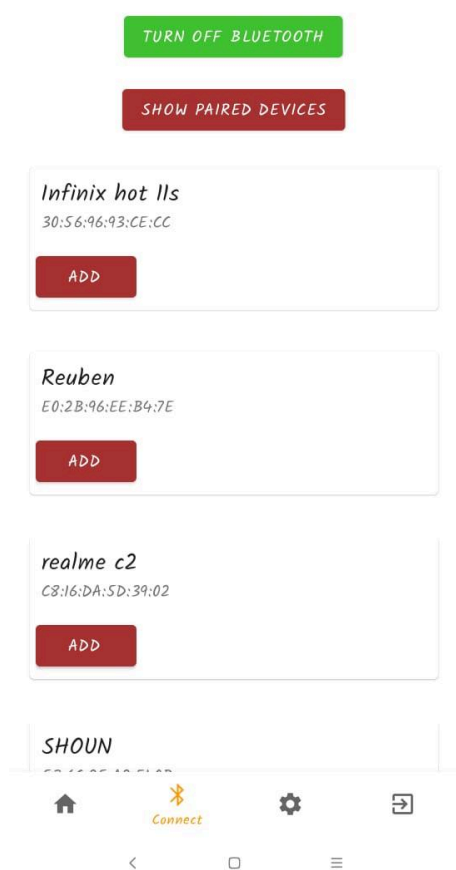
```
}
```

Info.toString contains the status (0 or 1). If btSocket is not null then send the status (0 or 1) to the other LED bluetooth device and then end the connection. The Bluetooth connection code is placed inside a thread to prevent load on the app At the same time the function UpdateMachine is called to make Changes To the user's status in the database

BluetoothConnect.java

The bluetooth connect class is used to display all the paired device information in a recyclerview. Its constructor is assigned with the name, address, user id and context

Connect.java

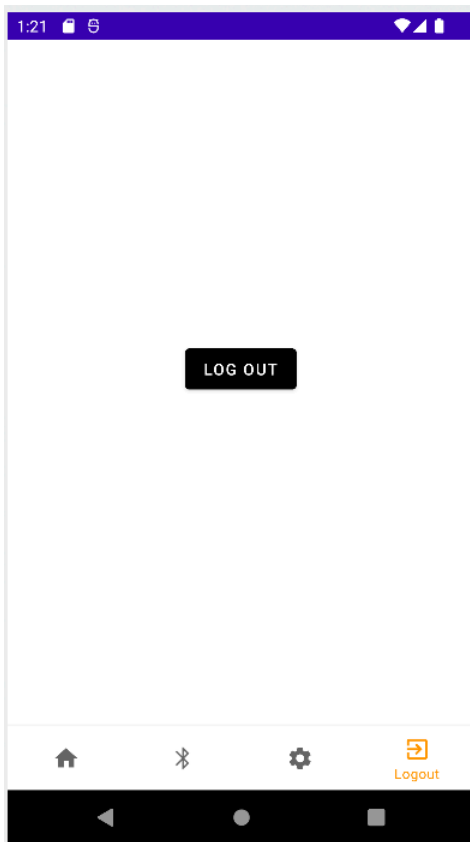


This fragment is used to see all the paired bluetooth devices and register them to the database. You can also turn on and turn off bluetooth from the app. If your phone doesn't support bluetooth you will receive a message that your phone doesn't support bluetooth. If the device has bluetooth then the user can turn on and turn off bluetooth. To turn off Bluetooth on our device we use

Adapter.disable() function and to turn on Bluetooth we use the **Adapter.enable()**

If the app doesn't have any permission to use bluetooth then the app will ask the user for permission. If the users clicks on show paired device then the app will display all paired device and a button to add your device to the database on click the add button your device is registered to the database

Logout.java



This page is used to allow the user to logout from the app. To remove the user's data locally we have to use SharedPreferences

```
SharedPreferences sharedPreferences =  
getActivity().getSharedPreferences("User", Context.MODE_PRIVATE);  
sharedPreferences.edit().clear().apply();
```

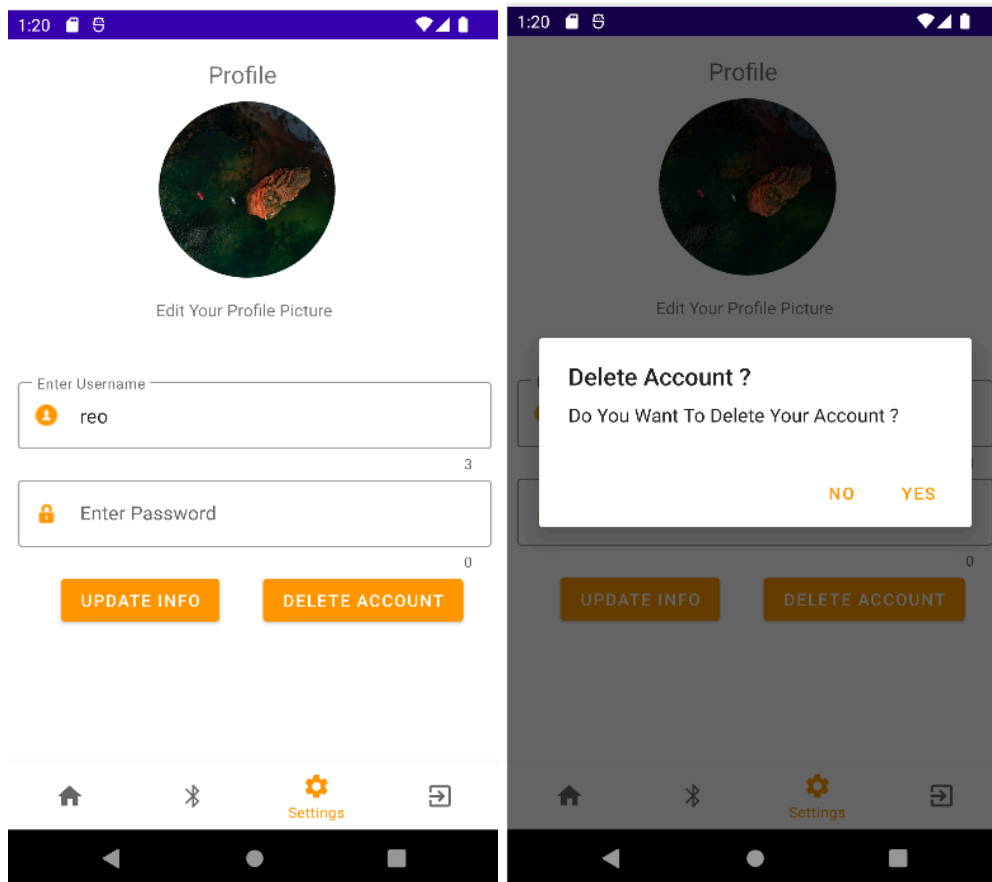
First we get the sharedPreferences User then use the **edit().clear().apply()** to clear all the user data

Then an Intent is created to Redirect the user to Main page which is

MainActivity.class

```
Intent intent = new Intent(getContext(),MainActivity.class);  
startActivity(intent);
```

Settings.java



In this page the user can change his current profile picture, username and password. The user can also delete his account or log out from the app. If the user presses on 'Edit your profile picture' a permission will be asked to allow access to images and if you allow this permission your gallery will open and you can select any one image as your new profile picture. If the user clicks on the delete button then an alert box will appear on the screen to select yes or no.

AlertDialog can be used to display the dialog message with OK and Cancel buttons. It can be used to interrupt and ask the user about his/her choice to continue or discontinue. we create an alert box by using the

AlertDialog.builder() function

```
AlertDialog.Builder builder = new AlertDialog.Builder(getContext());
```

We set then set the message using setMessage()

```
builder.setMessage("Do You Want To Delete Your Account ?");
```

We use **setTitle()** to set title of the dialog

```
builder.setTitle("Delete Account ?");
```

Set Cancelable false for when the user clicks on the outside the Dialog Box then it will remain show


```
builder.setCancelable(false);
```

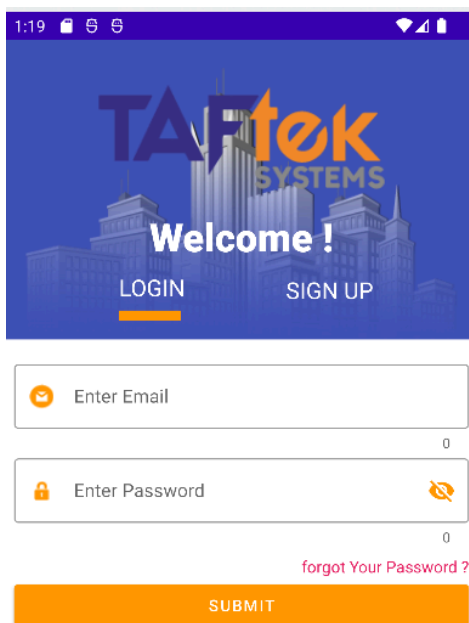
If the User Presses On Yes then **DeleteProfile()** from database class is called else if no is pressed the dialog box is closed

```
builder.setPositiveButton("Yes",(DialogInterface.OnClickListener)  
(dialogInterface, i) -> {  
    new Database(getContext()).DeleteProfile();  
});  
builder.setNegativeButton("No",(DialogInterface.OnClickListener)  
(dialogInterface, i) -> {  
    dialogInterface.cancel();  
});
```

We use this statement to show the dialog box

```
AlertDialog alertDialog = builder.create();  
alertDialog.show();
```

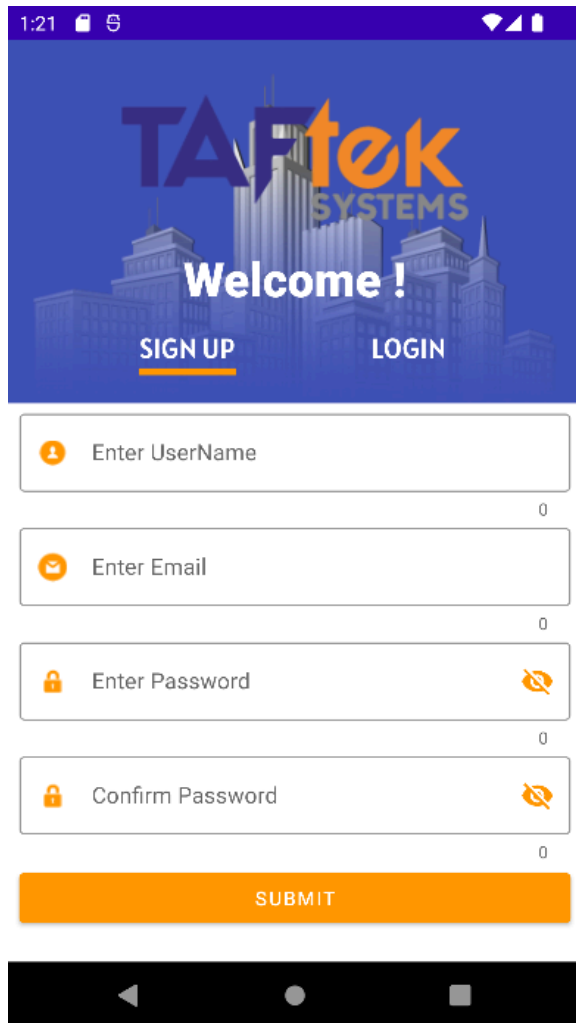
Login.java



If the user's account data is not found in the SharedPreferences then We will display the login Page to the user. The login page contains the app logo and two

options sign Up and Login in. in the Login page the user is asked to Enter His/Hers email address and password. When the user clicks on the submit button the function **SearchDb()** from Database is called. This function sends the data to the server and checks if the user's password and email is correct. If the user's password is correct then the home page (home.java) is displayed.

Signup.java



The screenshot shows a mobile application interface for a signup page. At the top, there is a blue header with the text "TAFTEK SYSTEMS" and "Welcome!". Below the header, there are two buttons: "SIGN UP" (underlined) and "LOGIN". The main form consists of four input fields, each with a label and a small icon on the left, and a small "0" below the field. The fields are: "Enter UserName" (with a person icon), "Enter Email" (with an envelope icon), "Enter Password" (with a lock icon and a toggle icon on the right), and "Confirm Password" (with a lock icon and a toggle icon on the right). Below the form is a large orange "SUBMIT" button. At the bottom, there is a black navigation bar with three icons: a back arrow, a home circle, and a recent apps square.

If the user doesn't have an existing account he/she can click on the signup option. The sign up page asks the user to enter his username, email, confirm password and password. If the user does not fill any one of the forms then **.setError();** is used to display a message with an exclamation icon and a red text to inform the user that he hasn't filled the form. The form will also check that the user's password and confirm password match.

If the user fills all the details then `insertData()` from Database Class is called which will store the user's details to the database on the server

```
if(!password.getText().toString().isEmpty()
    && !confirmPassword.getText().toString().isEmpty()
    && !email.getText().toString().isEmpty()
    && !username.getText().toString().isEmpty())
{
    new
    Database(this).insertData(password.getText().toString(),email.getText().toStri
ng(),username.getText().toString(),ps);
}
```

Database.java

This class contains all the necessary functions to communicate with the database. We use android volley to make http requests. Each function makes an http request to a given url which is a php file on the server. This php files searches certain details on the database and returns an output based on the input

XML

XML stands for Extensible Mark-up Language. We use XML in android studio to create the front end of the app. Here are some of the common used XML statement used by me for the app

Button

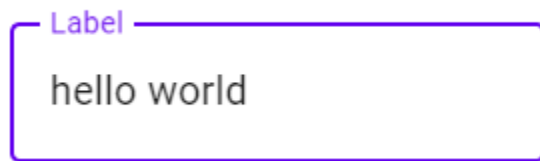
The following code is used to create a button with an Id button and background colour of green

<Button

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="BUTTON"
android:id="@+id/button"
android:background="#4CAF50"/>
```

TextInputLayout

This XML code is from material design used to create a form input



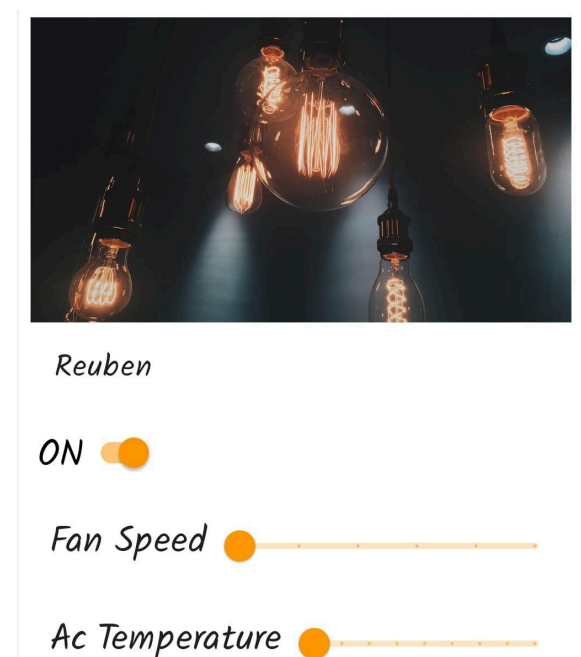
```
<com.google.android.material.textfield.TextInputLayout
    android:id="@+id/textField"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="label">
```

TextinputEditText

The text input edit text contains the message entered by the user in the form

```
<com.google.android.material.textfield.TextInputEditText
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
/>
</com.google.android.material.textfield.TextInputLayout>
```

CardView



The above design is created using a **MaterialCardView**. Cards contain content and actions that relate information about a subject. Inside the card we are showing an image and the device name , the status of the device true or false and two sliders to show the fan speed and ac temperature. We use a **textView** to display the text and a **switchMaterial** to display a switch. The **RangeSlider** is used for adjusting the fan and ac speed. We use an **ImageView** to display an image

```
<com.google.android.material.card.MaterialCardView
```

```
    android:id="card"
```

```
    android:layout_width="match_parent"
```

```
    android:layout_height="wrap_content"
```

```
    android:layout_margin="8dp">
```

```
    <LinearLayout
```

```
        android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content"
```

```
        android:orientation="vertical">
```

```
    <!-- Media -->
```

```
    <ImageView
```

```
        android:layout_width="match_parent"
```

```
        android:layout_height="194dp"
```

```
        app:srcCompat="@drawable/media"
```

```
        android:scaleType="centerCrop"
```

```
        android:contentDescription="@string/content_description_media"
```

```
    />
```

```
    <LinearLayout
```

```
        android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content"
```

```
        android:orientation="vertical"
```

```
        android:padding="16dp">
```

```
    <!-- Title -->
```

```
    <TextView
```

```
        android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
```

```
        android:text="Reuben"
        android:textAppearance="?attr/textAppearanceHeadline6"
    />
```

```
</LinearLayout>
```

```
<!-- Buttons -->
```

```
<LinearLayout
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_margin="8dp"
    android:orientation="horizontal">
```

```
<com.google.android.material.switchmaterial.SwitchMaterial
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginEnd="8dp"
    android:checked="true"
    android:text="ON"/>
```

```
<TextView
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Fan Speed"
/>
```

```
<com.google.android.material.slider.RangeSlider
```

```
    android:id="FAN"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:valueFrom="0.0"
    android:valueTo="5.0" />
```

```
<TextView
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Ac Temperature"
/>
```

```
<com.google.android.material.slider.RangeSlider
    android:id="@+id/AC"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:valueFrom="17.0"
    android:valueTo="30.0" />

</LinearLayout>
</LinearLayout>
</com.google.android.material.card.MaterialCardView>
```

PHP

Config.php

This file is used to connect to the database and also create tables if they don't exist in the database. This file is called in every php file

DeleteProfile.php

This php file is executed when the deleteProfile() function from the database class is executed. This file deletes the users id and machine ids from the database then uses the **unlink()** function to delete the users profile image stored on the database

insertData.php

This php file is used to create a user account. It takes in the users email, username and password and then checks if the user exists in the database. If this email already exists in the database then the app will tell the user that his account already exists. If their email doesn't exist in the database then we create a new account first we hash the password using the **hash()** function. We can directly use **sha1()** and **md5()** for hashing. The following statement is used for hashing the user's password **hash("ripemd320", sha1(md5(\$password)))**; then an unique id is created using the **uniqid()**. This function generates a unique ID based on the

microtime (the current time in microseconds). We enter all the users details to the database using a sql statement

```
mysqli_query($conn,"INSERT INTO  
userinfo(userid,username,email,password,profile) VALUES  
('$uuid','$username','$email','$pass','http://tafteksystems.com/profile/default.j  
pg'));
```

By default we set the profile image as <http://tafteksystems.com/profile/default.jpg>

InsertMachine.php

This php file takes in the userid, machineid (mac address) and device name we are first checking if this device is registered in the database. If this device is registered then the app will show a message that the device is already created if the device does not exists in the database then we create a machine key which is created using **uniqid()** and is then hashed with various algorithms

```
$hid = hash("ripemd320", sha1(md5(uniqid("",true)))));
```

Then the data is inserted to the database

```
$insertData = mysqli_query($conn, "INSERT INTO  
controll(userid,macid,status,devicename,devicekey)  
VALUES('$userid','$mid','false','$machinename','$hid');");
```

After inserting the data to the database we send a message back to the app that the device is registered successfully

```
$response["message"] = "Device Registered Successfully";
```

readLogin.php

This php file takes in the email and password of the user's email and then checks if this account exists in the database. If the user email exists then hash the password entered by the user and check if this hash password matches with the hashed password in the database. If the match is not correct then display to the user that the password is invalid. If the password matches with the one in the database then send the user id and email to the app and also send an alphabet **"B"** . if the

app receives this alphabet **B** then the app will show logged in successfully and the app will display the next page

Readmachine.php

This php file will take in a user id and will check with the database all the devices belonging to a particular user.

```
$in=mysqli_query($conn,"SELECT * FROM controll WHERE userid='$uid'");  
while($row = mysqli_fetch_array($in))  
{  
    array_push($arr,$row);  
}  
$response["message"] = $arr;  
echo json_encode($response);
```

All the device information will be pushed to an array **\$arr** and will be sent back to the app in a json format

updateImage.php

This php file is used for updating the user's profile picture. This file takes in the profile picture and userid then uses **base64_decode()** to decode the profile picture data send and uses **file_put_contents()** to store the profile data on the server and then create a url for the profile picture and update this url to the database

```
file_put_contents("profile/".$uid.".jpg",base64_decode($profile));  
$profileurl = 'http://tafteksystems.com/profile/'.$uid.'.jpg';  
$response["profileurl"] = $profileurl;  
$in = mysqli_query($conn,"UPDATE userinfo SET profile = '$profileurl' WHERE  
userid = '$uid';");
```

UpdateMachine.php

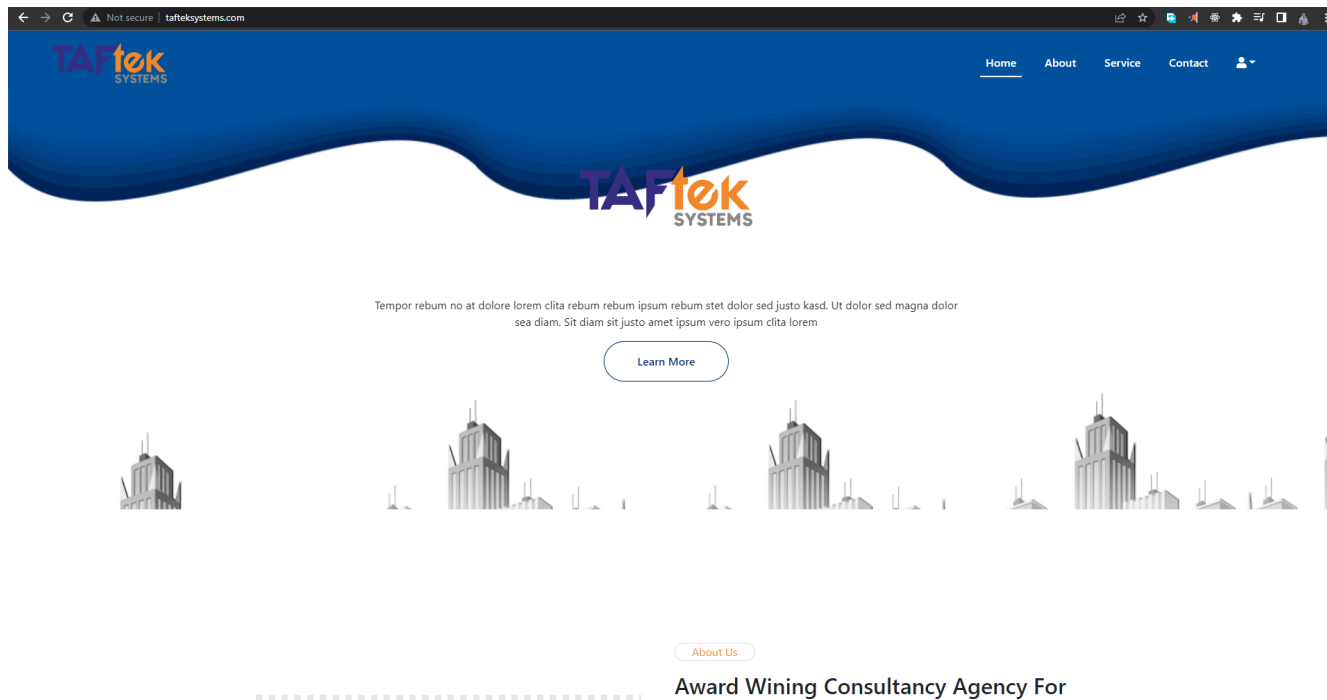
This php file is used for updating the status of the device. It takes in userid , status and machineid and then updates the status of the device belonging to a certain device id and user id and then sends a message back to the app that the data is updated successfully

updateProfile.php

This php file is used for updating the users username and password in the database. It takes in the user id and updates the username and password and sends a success message back to the app

```
$uid = $_POST["userid"];  
if($_POST["username"]  
{  
    $uname = $_POST["username"];  
    mysqli_query($conn,"UPDATE userinfo SET username = '$uname' WHERE userid  
= '$uid';");  
    $response["message"] = "Entered Username";  
    $response["newuser"] = $uname;  
}  
if($_POST["password"])  
{  
    $password = $_POST["password"];  
    $pass = hash("ripemd320", sha1(md5($password)));  
    mysqli_query($conn,"UPDATE userinfo SET password = '$pass' WHERE userid  
= '$uid';");  
    $response["message"] = "Entered Username and password";  
}
```


Website




AJAX was used for the Websites Login and signup page. These webpages were made by our mentor using Wordpress. She told us to do the login and sign up using php.

LOGIN

SIGN UP

o@gmail.com

Email is correct



Wrong Password Entered

Forgot Your Password?

Login

Each time the user types something on the form. A javascript code will be executed and this executed code will check with the database and update the message on the form

```
function showHint(str, type) {  
  if (str.length == 0) {  
    document.getElementById("emailHelp").innerHTML = "Enter Your Email";  
    return;  
  } else {  
    var xmlhttp = new XMLHttpRequest();  
    xmlhttp.onreadystatechange = function () {  
      if (this.readyState == 4 && this.status == 200) {  
        document.getElementById("emailHelp").innerHTML = this.responseText;  
      }  
    };  
    xmlhttp.open("GET", `loginforwebsite.php?Email=${str}`, true);  
    xmlhttp.send();  
  }  
}
```

The above code will open **loginforwebsite.php** and will send the email filled in the form to the php. file this php file will then check if the email exists in the database and will return either Email not found or Email is correct. If the ReadyState is 4 and Status is 200 then it will display the message sent by the php file to the webpage. A readyState 4 means that the request had been sent, the server had finished returning the response and the browser had finished downloading the response content. A status of 200 OK success status response code indicates that the request has succeeded

```
if($_GET["Email"]){  
  $email = $_GET["Email"];  
  $in = mysqli_query($conn, "SELECT * FROM userinfo WHERE email = '$email'");  
  $row = mysqli_fetch_assoc($in);  
  if(mysqli_num_rows($in) > 0)  
  {  
    if($row["email"] == $email)
```

```

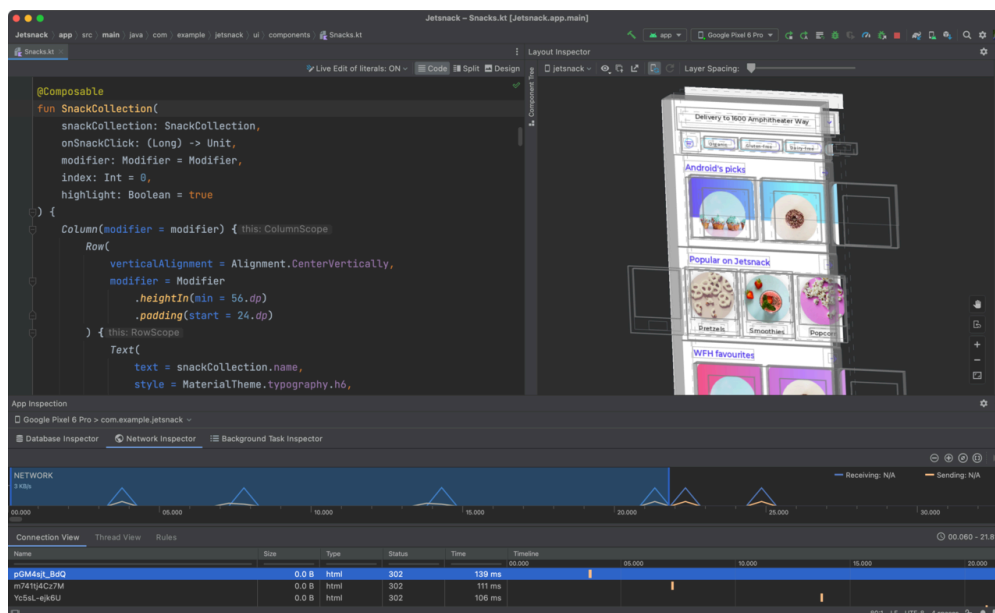
    {
        $_SESSION["email"] = $row["email"];
        echo "<p style='color:green;'>Email is correct</p>";
    }
}
else {
    echo "<p style='color:red;'>Email is Not correct</p>";
}
}
}

```

This same principle is used for password and signup page

Chapter 6 : knowledge and skill gained by the students during training period

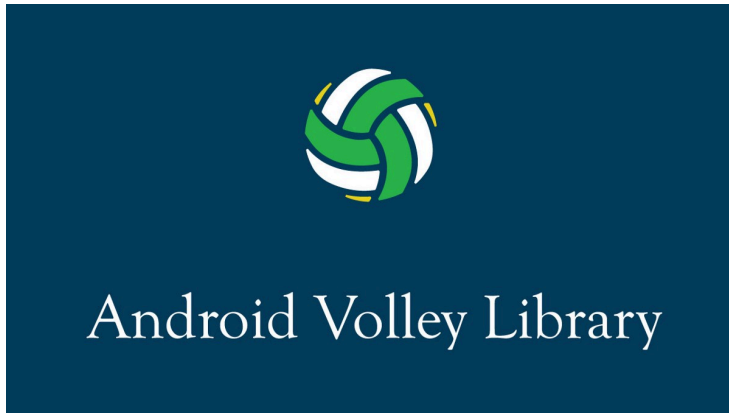
During the training period of 8 weeks i have learnt to use various tools They are **Android Studio**



Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems. It is a replacement for the Eclipse Android Development Tools (E-ADT) as the primary IDE for native Android application development. We can create app using java or kotlin in 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 Version 7 language features and a subset of Java 8 language features that vary by platform version.

Android Volley

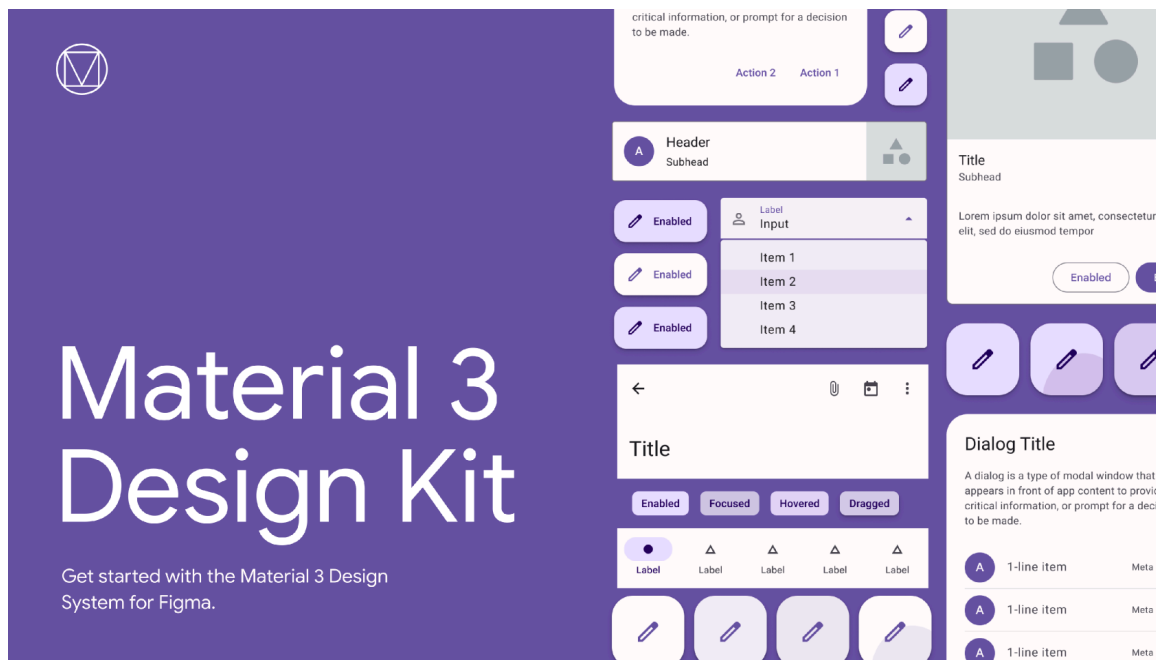


Volley is an HTTP library that makes networking very easy and fast, for Android apps. It was developed by Google and introduced during Google I/O 2013. It was developed because there is an absence in Android SDK, of a networking class capable of working without interfering with the user experience. Although Volley is a part of the Android Open Source Project(AOSP), Google announced in January 2017 that Volley will move to a standalone library.

Features of Volley:

1. Effective request cache and memory management
2. Extensibility and customization of the library to our needs
3. Cancelling the requests

Material Design



Material Design (codenamed Quantum Paper) is a design language developed by Google in 2014. Material Design uses more grid-based layouts, responsive animations and transitions, padding, and depth effects such as lighting and shadows. Google announced Material Design on June 25, 2014, at the 2014 Google I/O conference.

The main purpose of Material Design is the creation of a new visual language that combines principles of good design with technical and scientific innovation

AJAX

Ajax (Asynchronous JavaScript and XML) is a set of web development techniques that uses various web technologies on the client-side to create asynchronous web applications. With Ajax, web applications can send and retrieve data from a server asynchronously (in the background) without interfering with the display and behaviour of the existing page.

- Update a web page without reloading the page
- Request data from a server - after the page has loaded
- Receive data from a server - after the page has loaded
- Send data to a server - in the background

Chapter 7: Training summary

The training was a new experience for me to explore because it was the first time I had gone for Industrial training . It was a good experience and we were made to work on time. Before joining the training I had some experience with java programming but after completing the industrial training i learnt to create an app using android studio with java and xml. I also learnt various new functions and concepts of android studio. With material design and xml i learnt how to add buttons , forms, checkbox and images to the ui. I learnt concepts like recyclerview, Fragment and new java concepts like lambda, foreach loop and i learnt how to use android volley a library which allows us to communicate with the database from the app. I used material design to create beautiful designs for the app. I had to use AJAX to show the login/signup message without reloading the website. PHP was used to act as a communication between the database and the app/website And mysql was also used for storing data