# WEB AND MOBILE PROGRAMMING

**Group: 07**
**<u>Students:</u>**
**Venkata Naga Sai Srikanth Gollapudi, sggc9@umsystem.edu**
**Varun Reddy Mangi, vmf9@umsystem.edu**

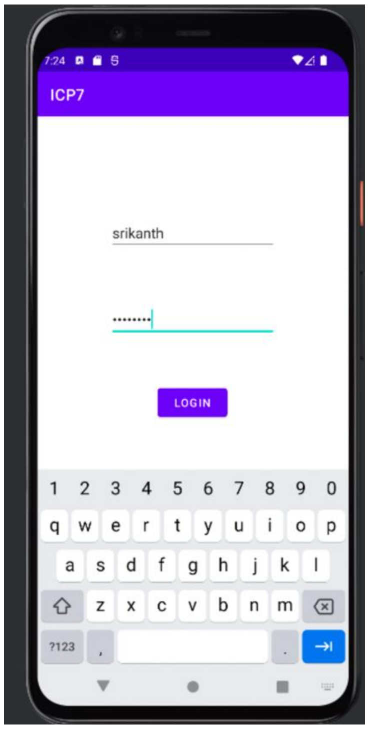
**Video link:** https://youtu.be/TuVsdzCnGH0
**PPT link:**
https://github.com/GollapudiSrikanth/WebMobileSpring2022/tree/main/Mobile/ICP_Presentation

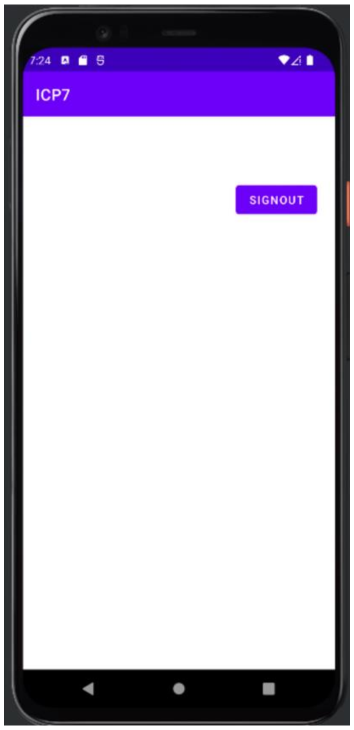
## ICP-8: Android Studio Login Page

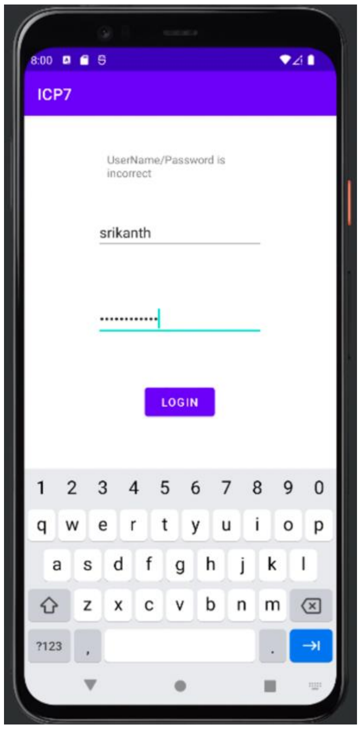
1. Created a project and installed Google pixel 5 with Android OS version of 12.
2. Now, we have created a interface for login page which contains Username and password fields and also a login button.
3. The above fields are created using EditText option with type as user and password.
4. Login button is created using button option.
5. All the business logic is witten in the MainActivity.java.
6. On entering valid details and clicking on login button, user will be directed to the home page.
7. The home page contains only a logout button. If the user clicks the logout button, then the user will be directed to the login page.
8. If user enters invalid details, an error message called "Username/Password is incorrect" will be displayed on the screen.

From this ICP we are able to implement a design from UI Wire Frames.

**OUTPUT:**


7:24
ICP7
srikanth
LOGIN


7:24
ICP7
SIGNOUT


8:00
ICP7
UserName/Password is incorrect
srikanth
LOGIN

**ICP-9: API CALLING**

We have created a pizza ordering application using android studio.

1. This app basically helps users to order pizzas of their liking.

2. The interface contains Text edits which asks details from the user (user name and the user email id).

3. We have added 4 different toppings from which a user can select any by just clicking on the check box.

4. User can increase or decrease the quantity of the toppings by just clicking on + or – buttons.

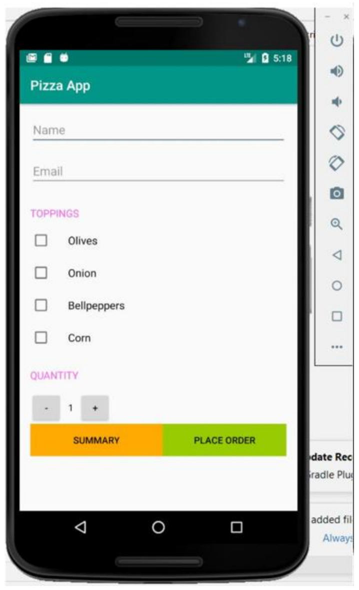
5. We have also included summary and place order buttons.

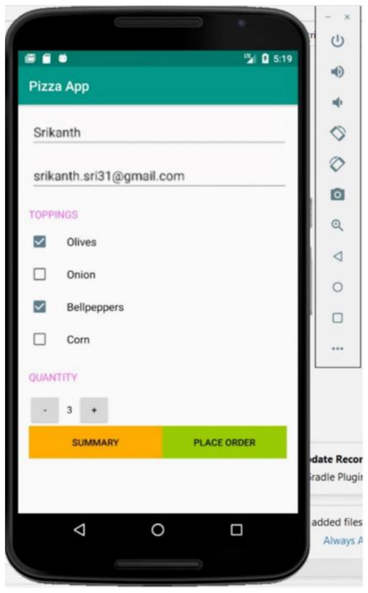
6. On clicking summary button, the application displays the order summary.

7. On selecting place order, the application sends a summary mail to the respective email id.

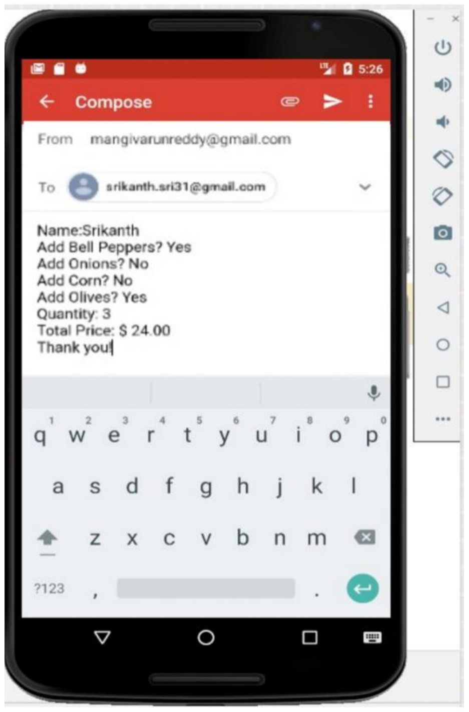
From this ICP we are able to create and Pizza application and able to order a pizza with report sent to customer email ID.

**OUTPUT:**



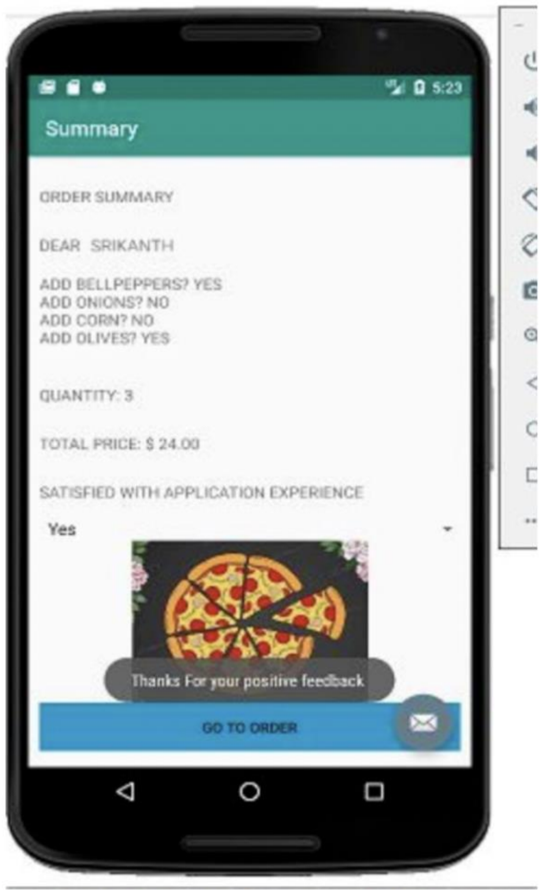


Srikanth
srikanth.sri31@gmail.com
✓ Olives
☐ Onion
✓ Bellpeppers
☐ Corn
- 3 +



From mangivarunreddy@gmail.com
To srikanth.sri31@gmail.com

Name:Srikanth
Add Bell Peppers? Yes
Add Onions? No
Add Corn? No
Add Olives? Yes
Quantity: 3
Total Price: $ 24.00
Thank you!



Summary

ORDER SUMMARY

DEAR SRIKANTH

ADD BELLPEPPERS? YES
ADD ONIONS? NO
ADD CORN? NO
ADD OLIVES? YES

QUANTITY: 3

TOTAL PRICE: $ 24.00

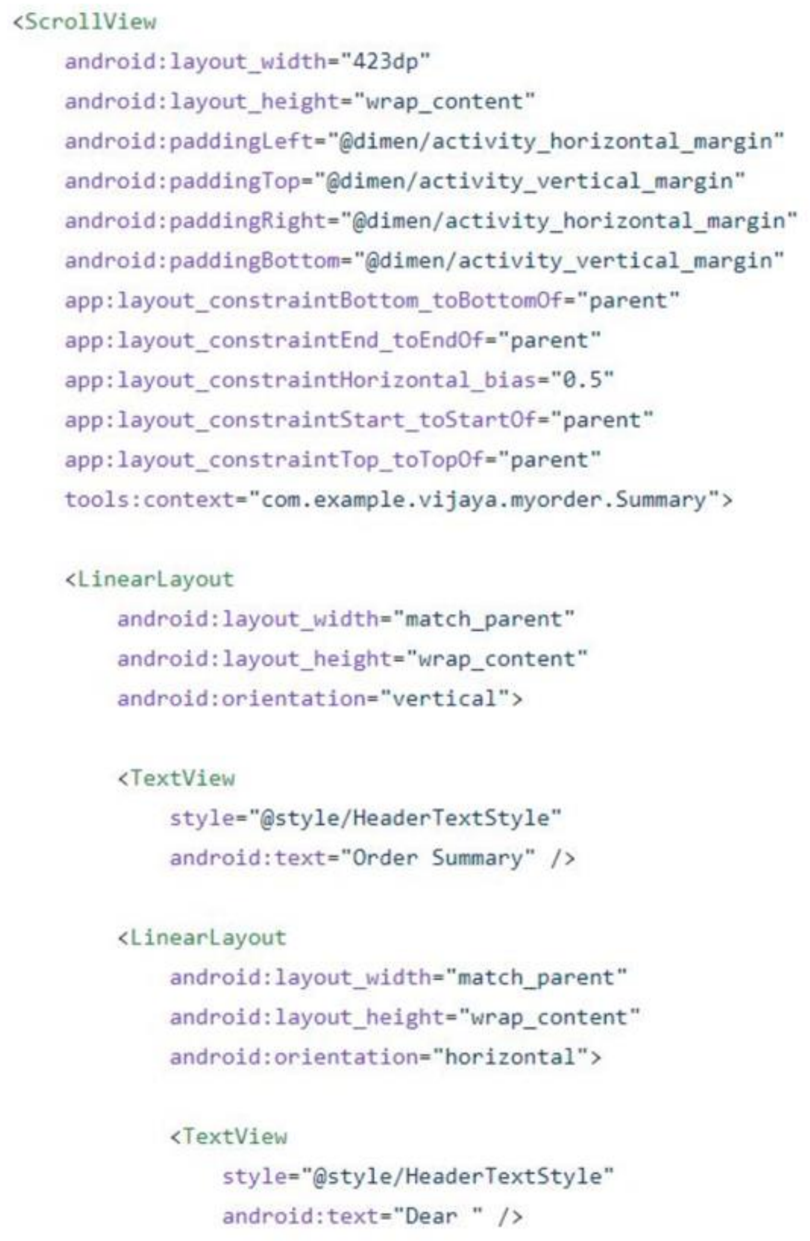
SATISFIED WITH APPLICATION EXPERIENCE

Thanks For your positive feedback

Summary page logic in the PizzaApp

```
Intent redirect = new Intent(MainActivity.this, Summary.class);
redirect.putExtra("OrderSummaryMessage", orderSummaryMessage);
redirect.putExtra("order", order);
redirect.putExtra("price", price1);
redirect.putExtra("quant", quant);
redirect.putExtra("name", name);
```

Order page design code in the PizzaApp adding scroll bar to the application



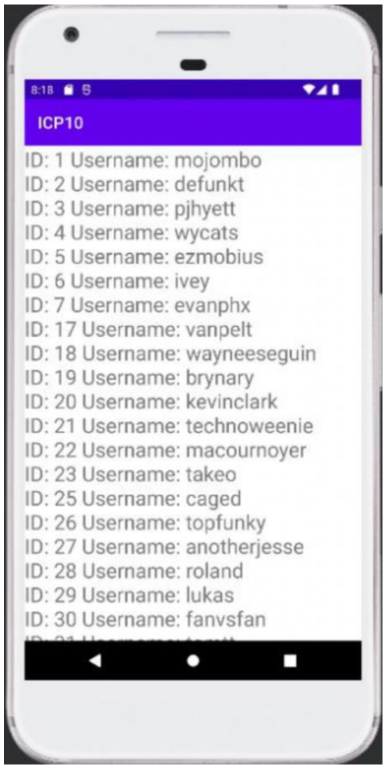
```
<ScrollView
    android:layout_width="423dp"
    android:layout_height="wrap_content"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    tools:context="com.example.vijaya.myorder.Summary">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">

        <TextView
            style="@style/HeaderTextStyle"
            android:text="Order Summary" />

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal">

            <TextView
                style="@style/HeaderTextStyle"
                android:text="Dear " />
```

**ICP-10: Consuming the Rest Api**

1. Created the project using Android studio and added two dependencies which are retrofit2.9.0, converter-gson:2.9.0.
2. Created a java class file "User.java" which acts as a data transfer object which consists of id and username with getters and setters.
3. In the layout added a text field with id of 'text1' and added a nested scroll view to the text1 which gives the way to scroll up and down.
4. In the onCreate function in the MainActivity.java,retrofit api is used consumed the github rest api to fetch the users data.
5. On Successful call of api, response is collected in the onResponse method and casted to User class which is collected in the list.
6. The collected list is then displayed in the text view which is present in the MainAcitivity.xml and displayed to the user.

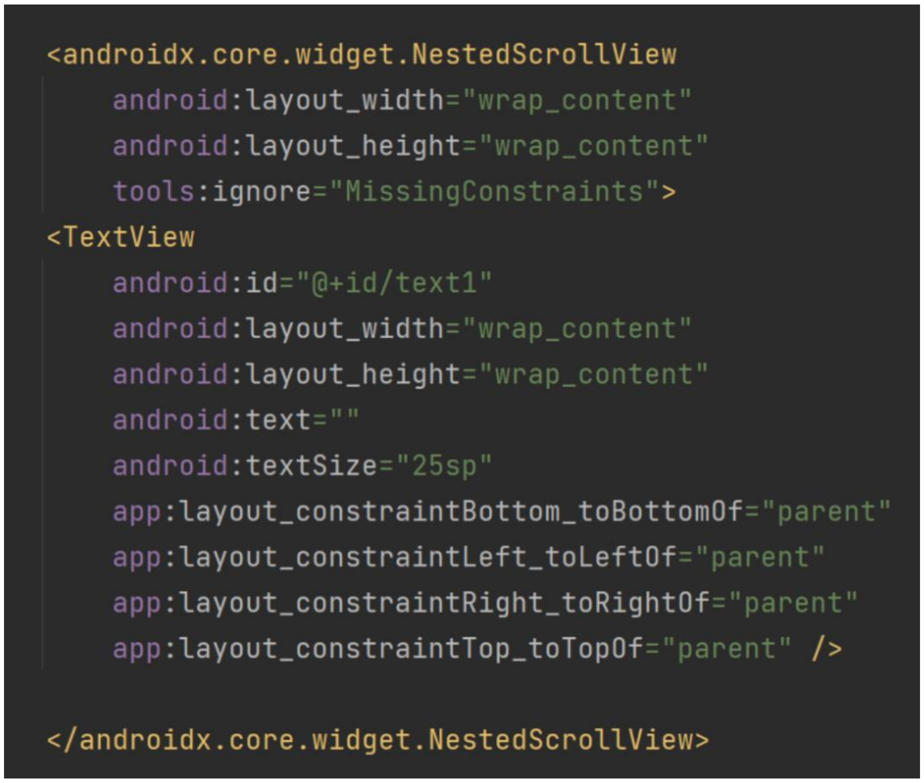
From this ICP we are able to consume rest api in the mobile application and got to know the dependencies of it

**OUTPUT:**



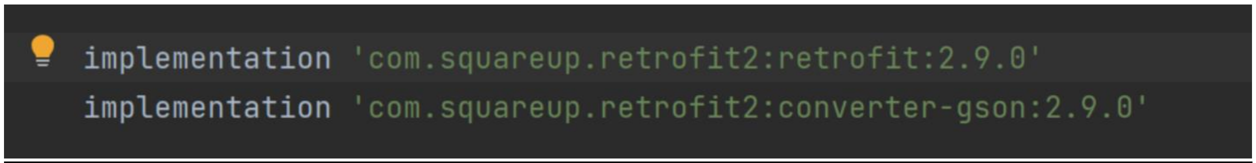
```
ID: 1 Username: mojombo
ID: 2 Username: defunkt
ID: 3 Username: pjhyett
ID: 4 Username: wycats
ID: 5 Username: ezmobius
ID: 6 Username: ivey
ID: 7 Username: evanphx
ID: 17 Username: vanpelt
ID: 18 Username: wayneeseguin
ID: 19 Username: brynary
ID: 20 Username: kevinclark
ID: 21 Username: technoweenie
ID: 22 Username: macournoyer
ID: 23 Username: takeo
ID: 25 Username: caged
ID: 26 Username: topfunky
ID: 27 Username: anotherjesse
ID: 28 Username: roland
ID: 29 Username: lukas
ID: 30 Username: fanvsfan
```

adding Scroll Bar to the textview



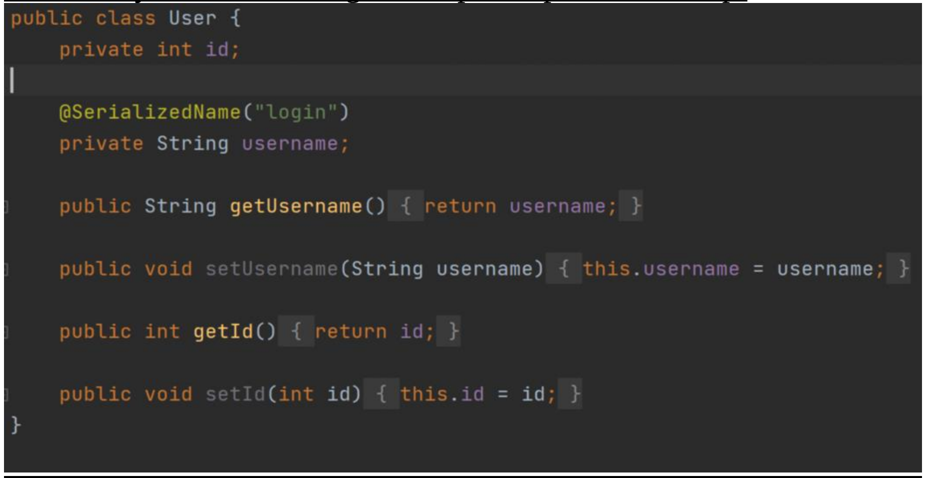
```
<androidx.core.widget.NestedScrollView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    tools:ignore="MissingConstraints">
<TextView
    android:id="@+id/text1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text=""
    android:textSize="25sp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

</androidx.core.widget.NestedScrollView>
```

Dependencies added to the project in the build.gradle file



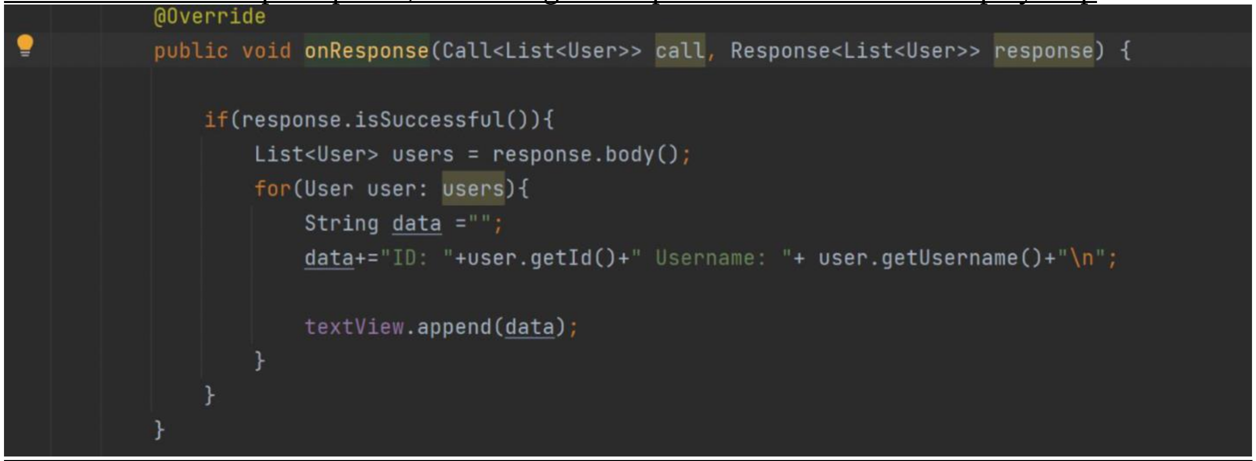
```
implementation 'com.squareup.retrofit2:retrofit:2.9.0'
implementation 'com.squareup.retrofit2:converter-gson:2.9.0'
```

User Entity used for casting the output response of the api



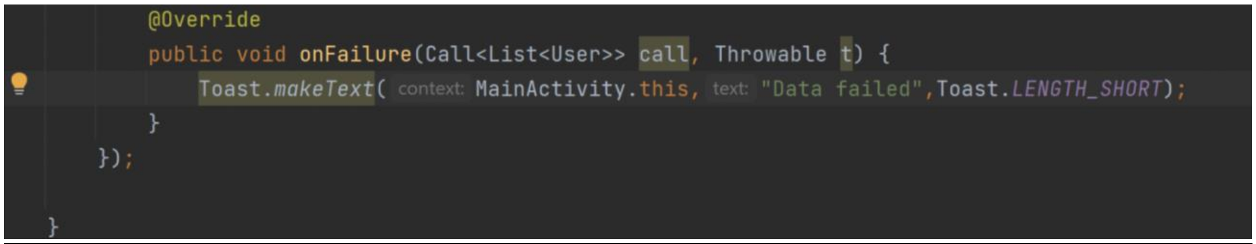
```
public class User {
    private int id;

    @SerializedName("login")
    private String username;

    public String getUsername() { return username; }

    public void setUsername(String username) { this.username = username; }

    public int getId() { return id; }

    public void setId(int id) { this.id = id; }
}
```

On successful rest-api response, converting the response to users list and displays o/p



```
        @Override
        public void onResponse(Call<List<User>> call, Response<List<User>> response) {

            if(response.isSuccessful()){
                List<User> users = response.body();
                for(User user: users){
                    String data ="";
                    data+="ID: "+user.getId()+" Username: "+ user.getUsername()+"\n";

                    textView.append(data);
                }
            }
        }
```

On Failure of getting the response from the rest-api the this function executes



```
        @Override
        public void onFailure(Call<List<User>> call, Throwable t) {
            Toast.makeText( context: MainActivity.this, text: "Data failed",Toast.LENGTH_SHORT);
        }
    });

}
```

**ICP-11: Text to Speech**

In this ICP, we have created an application which converts a text into a speech.

1. First we have added textview, edit text and a button to design the interface.

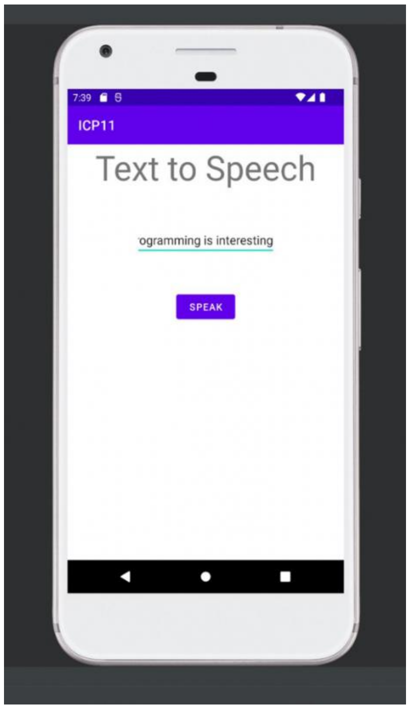
2. textview displays the tile of the icp. Edit text is a field which takes the input from the user which the application later converts it to a speech on clicking button named as speak.

3. The converstion process is taken care by text to speech api.

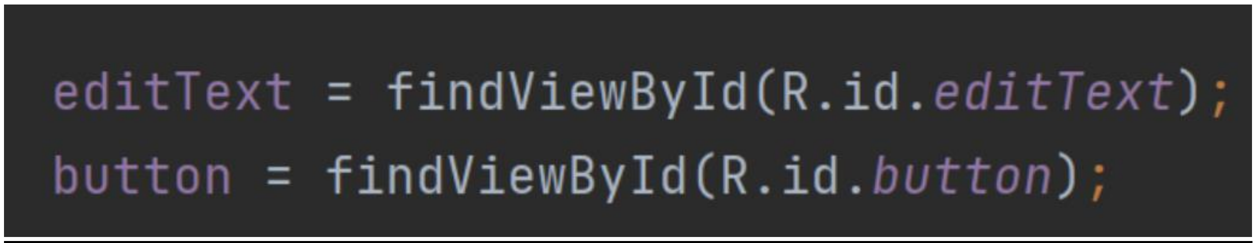
4. In this text to speech api, speak method converts the text to speech.

5. If the input entered by the user is unpronounceable or contains any error, then the error message “TextToSpeech cannot be translated” will be displayed.

**OUTPUT:**

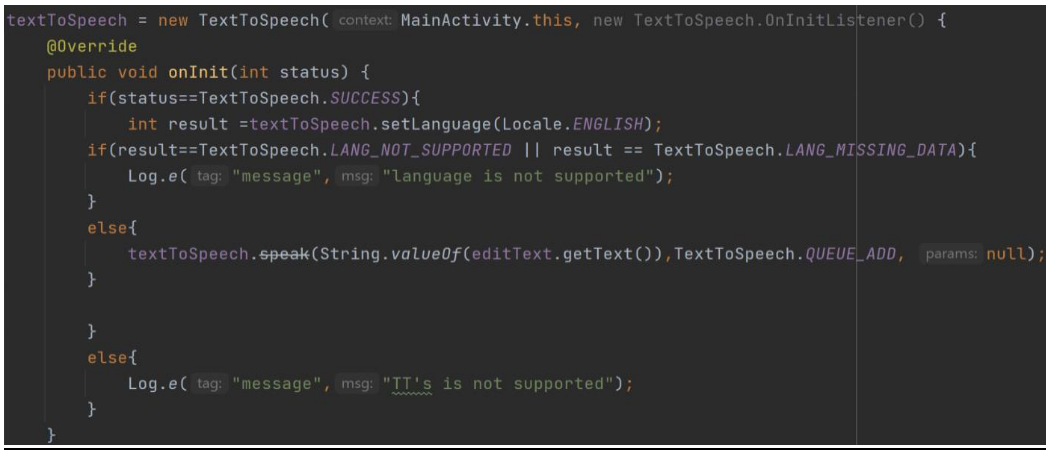


Fetching the data using Id's



```
editText = findViewById(R.id.editText);
button = findViewById(R.id.button);
```

Text to speech logic which handles error and success scenario's



```
textToSpeech = new TextToSpeech( context: MainActivity.this, new TextToSpeech.OnInitListener() {
    @Override
    public void onInit(int status) {
        if(status==TextToSpeech.SUCCESS){
            int result =textToSpeech.setLanguage(Locale.ENGLISH);
        if(result==TextToSpeech.LANG_NOT_SUPPORTED || result == TextToSpeech.LANG_MISSING_DATA){
            Log.e( tag: "message", msg: "language is not supported");
        }
        else{
            textToSpeech.~~speak~~(String.valueOf(editText.getText()),TextToSpeech.QUEUE_ADD,  params: null);
        }

        }
        else{
            Log.e( tag: "message", msg: "TT's is not supported");
        }
    }
```