ICP (8-11) Presentation 2 Report

Team details:

Sarath Chandra Kolisetty – sk83g@umsystem.edu
Github links:
Presenation: https://github.com/sarath98-lab/spring-2022/tree/main/Mobile/ICP-Presentation
Source code: https://github.com/sarath98-lab/spring-2022/tree/main/Mobile

Eeshwara Sai Tota – ettkv@umsystem.edu
Github links:
Presenation: https://github.com/SaiKicks/WebMobile-Spring2022/tree/main/mobile/ICP-presentation
Source code: https://github.com/SaiKicks/WebMobile-Spring2022/tree/main/mobile/

Presentation Video:
https://drive.google.com/drive/folders/1EKqrExzU8skzYSCCjtGwPH9X4yDplbl1?usp=sharing

Introduction:

In the ICPs 8-11, the following are the key tasks to be done for each mobile application:

Login application – Input validations, redirect from one activity intent to another.

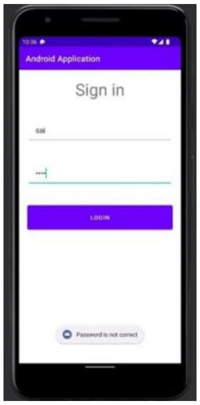
Pizza delivery application – Liner layout, ACTION_SEND intent to send email

Display data using public APIs – Grant user permission to access internet, retrofit to fetch data from APIs

Text to speech application – Converting text to audio format

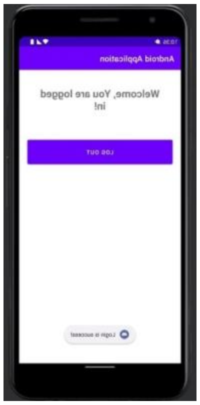
Features:

Login application:



sai

Password is not correct

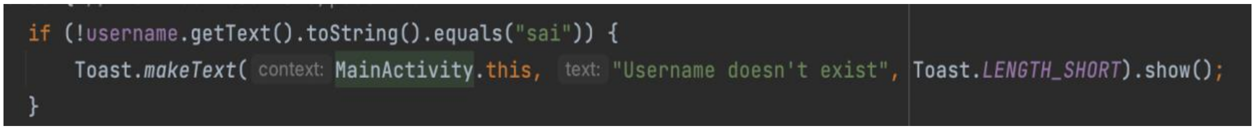


Welcome, You are logged in!

Login is success!

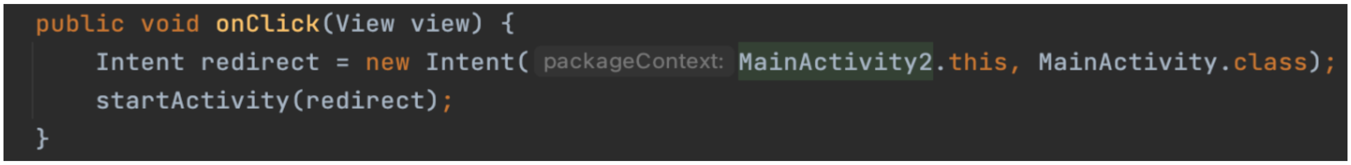
- The sign in screen has input validations if login is done using either blank fields or incorrect username or password toast message is shown.
- During a successful login, screen is redirect to a new intent screen where log out button is displayed.

Toast message:



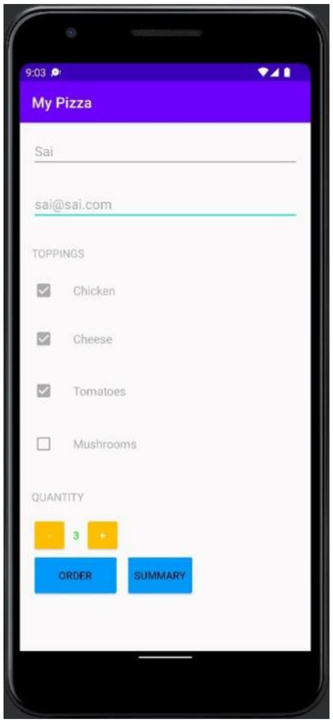
```
if (!username.getText().toString().equals("sai")) {
    Toast.makeText( context: MainActivity.this,  text: "Username doesn't exist", Toast.LENGTH_SHORT).show();
}
```

Redirect to another activity intent:



```
public void onClick(View view) {
    Intent redirect = new Intent( packageContext: MainActivity2.this, MainActivity.class);
    startActivity(redirect);
}
```

Pizza delivery application:



Sai

sai@sai.com

TOPPINGS

☑ Chicken

☑ Cheese

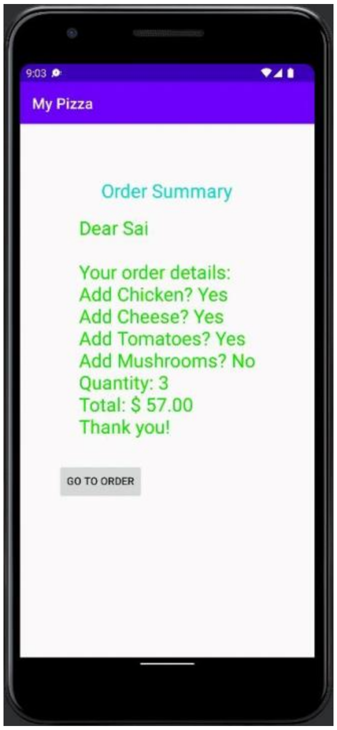
☑ Tomatoes

□ Mushrooms

QUANTITY

- 3 +

ORDER SUMMARY

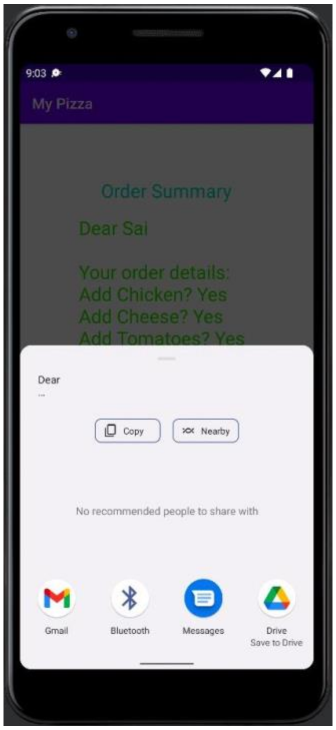


Order Summary

Dear Sai

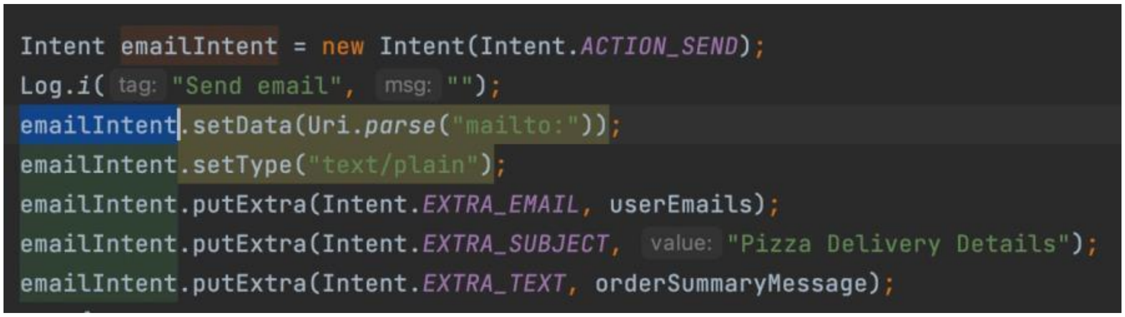
Your order details:
Add Chicken? Yes
Add Cheese? Yes
Add Tomatoes? Yes
Add Mushrooms? No
Quantity: 3
Total: $ 57.00
Thank you!

GO TO ORDER



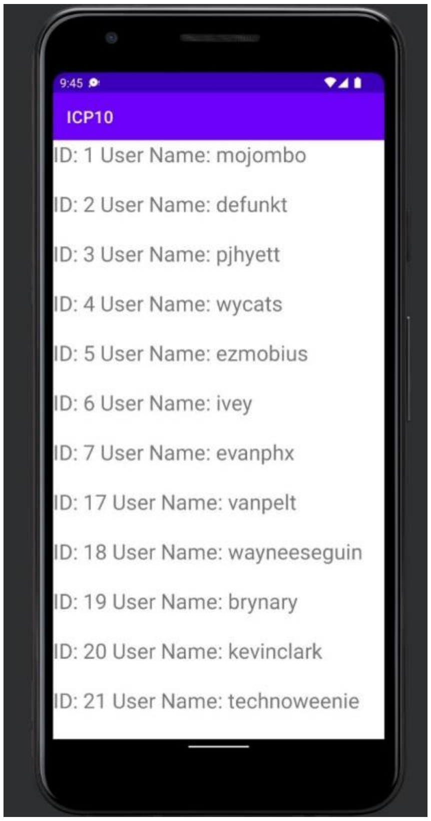
- Using the liner layout, we have designed form to order pizzas.
- To view the summary, we redirect to new intent where we calculate price associated with each item and display final bill.
- ACTION_SEND intent is used with extra information which include recipient email, subject, and content of email to send email.

Code snippet to send mail:



```
Intent emailIntent = new Intent(Intent.ACTION_SEND);
Log.i( tag: "Send email",  msg: "");
emailIntent.setData(Uri.parse("mailto:"));
emailIntent.setType("text/plain");
emailIntent.putExtra(Intent.EXTRA_EMAIL, userEmails);
emailIntent.putExtra(Intent.EXTRA_SUBJECT,  value: "Pizza Delivery Details");
emailIntent.putExtra(Intent.EXTRA_TEXT, orderSummaryMessage);
```

Display data from public APIs:



ID: 1 User Name: mojombo

ID: 2 User Name: defunkt

ID: 3 User Name: pjhyett

ID: 4 User Name: wycats

ID: 5 User Name: ezmobius

ID: 6 User Name: ivey

ID: 7 User Name: evanphx

ID: 17 User Name: vanpelt

ID: 18 User Name: wayneeseguin

ID: 19 User Name: brynary

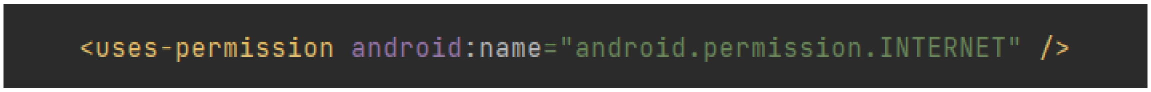
ID: 20 User Name: kevinclark

ID: 21 User Name: technoweenie

- In order to fetch data from internet, user permission is added in manifest file.
- Retrofit

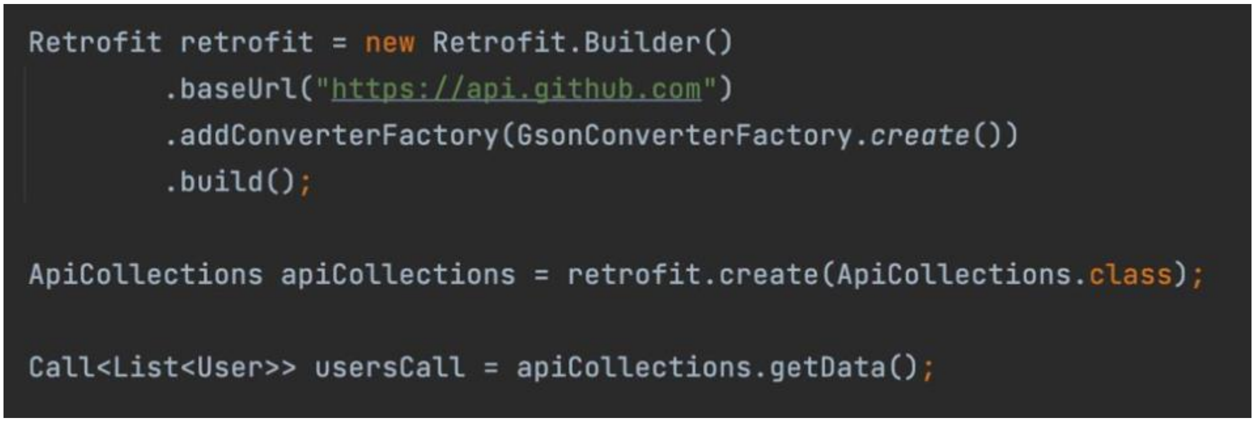
Using retrofit we construct the API endpoint and fetch data from GIT public API.

Permission to access data from internet:



```
<uses-permission android:name="android.permission.INTERNET" />
```

Fetch data from GIT API

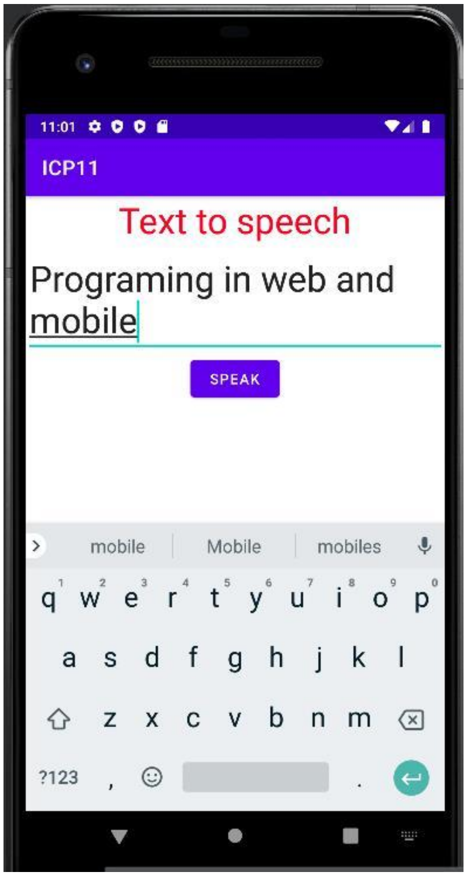


```
Retrofit retrofit = new Retrofit.Builder()
        .baseUrl("https://api.github.com")
        .addConverterFactory(GsonConverterFactory.create())
        .build();

ApiCollections apiCollections = retrofit.create(ApiCollections.class);

Call<List<User>> usersCall = apiCollections.getData();
```

Text to speech:

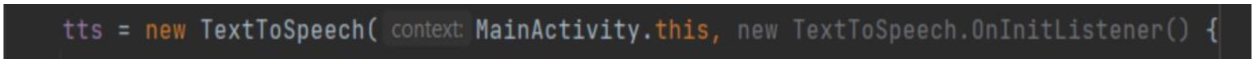


Text to speech

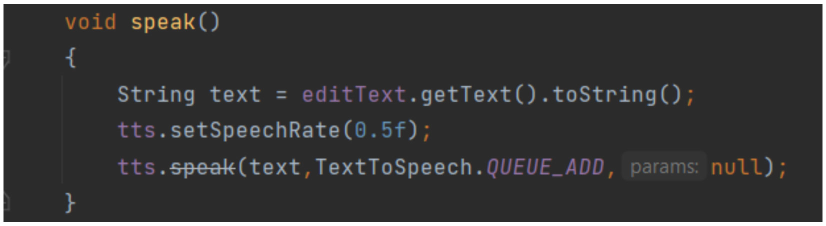
Programing in web and mobile

- User input is converted to text when clicked on speak button
- Using TextToSpeech listener is successfully initialized we set the desired speech rate
- and add the text to Queue.
- The input text entered is heard in audio format.

Text to speech code snippet:



```
tts = new TextToSpeech( context: MainActivity.this, new TextToSpeech.OnInitListener() {
```



```
void speak()
{
    String text = editText.getText().toString();
    tts.setSpeechRate(0.5f);
    tts.~~speak~~(text,TextToSpeech.QUEUE_ADD, params: null);
}
```

Conclusion:

By completing the above ICPs we have learnt:

- Develop an android application using android studio.
- The project structure followed i.e., activity_main.xml for design, MainAcitity.java for functionality and manifest file for meta data.
- Redirect from one activity intent to another.
- Access APIs using retrofit.
- To grant user permission to access hardware components like mic and speakers and interact with them.