Implicit Intent – WebLinkAndPhoneCallApp

Name : Snehal Yeole

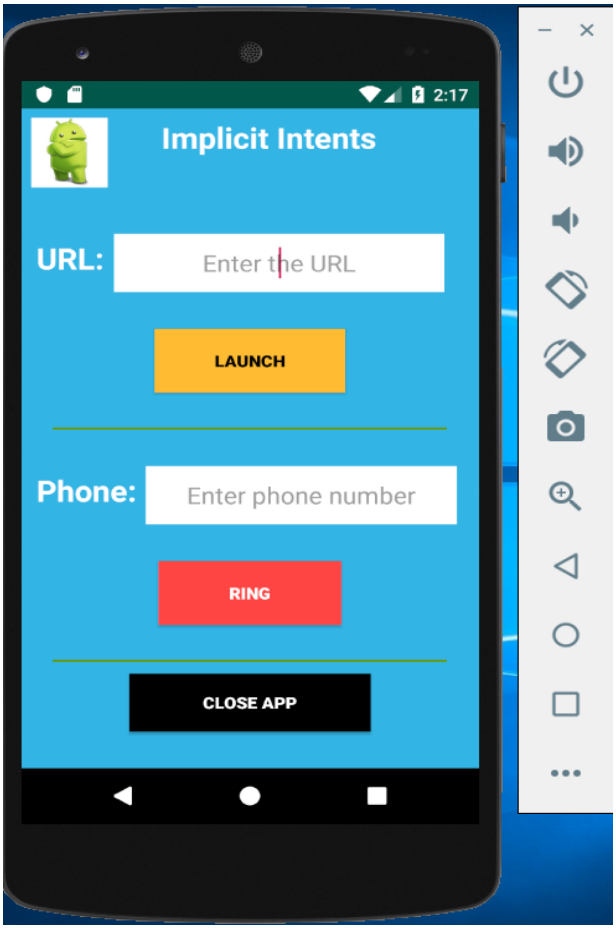
SJSU ID : 012548471

Implicit Intents:

Implicit intents are used to invoke other applications within the device. Intents are nothing but an action user wants to perform on other application. These intents does not specify the components explicitly, but instead declare the action to perform wherein a component from another application can handle it.

Home Screen:

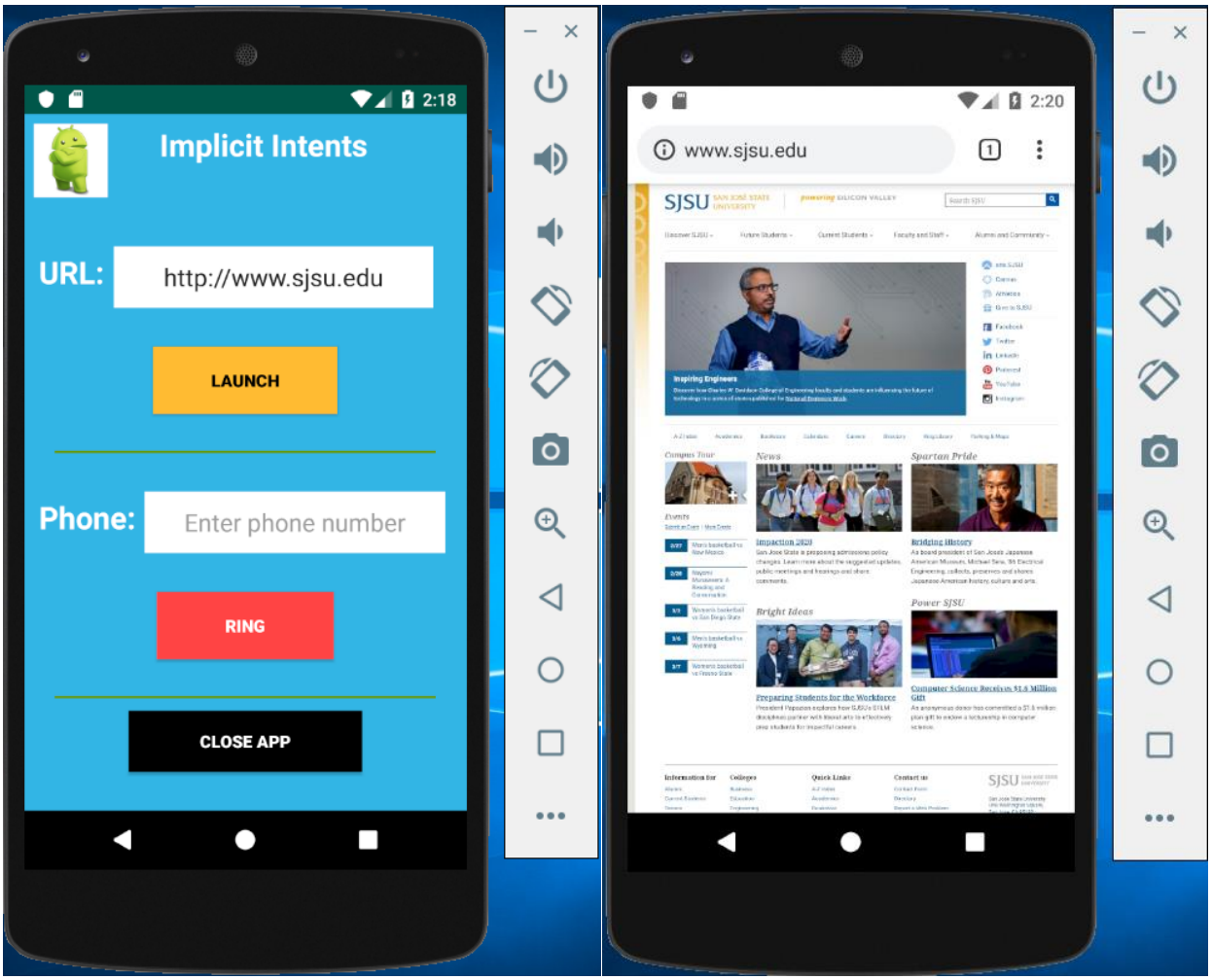
The home screen for implicit intents application



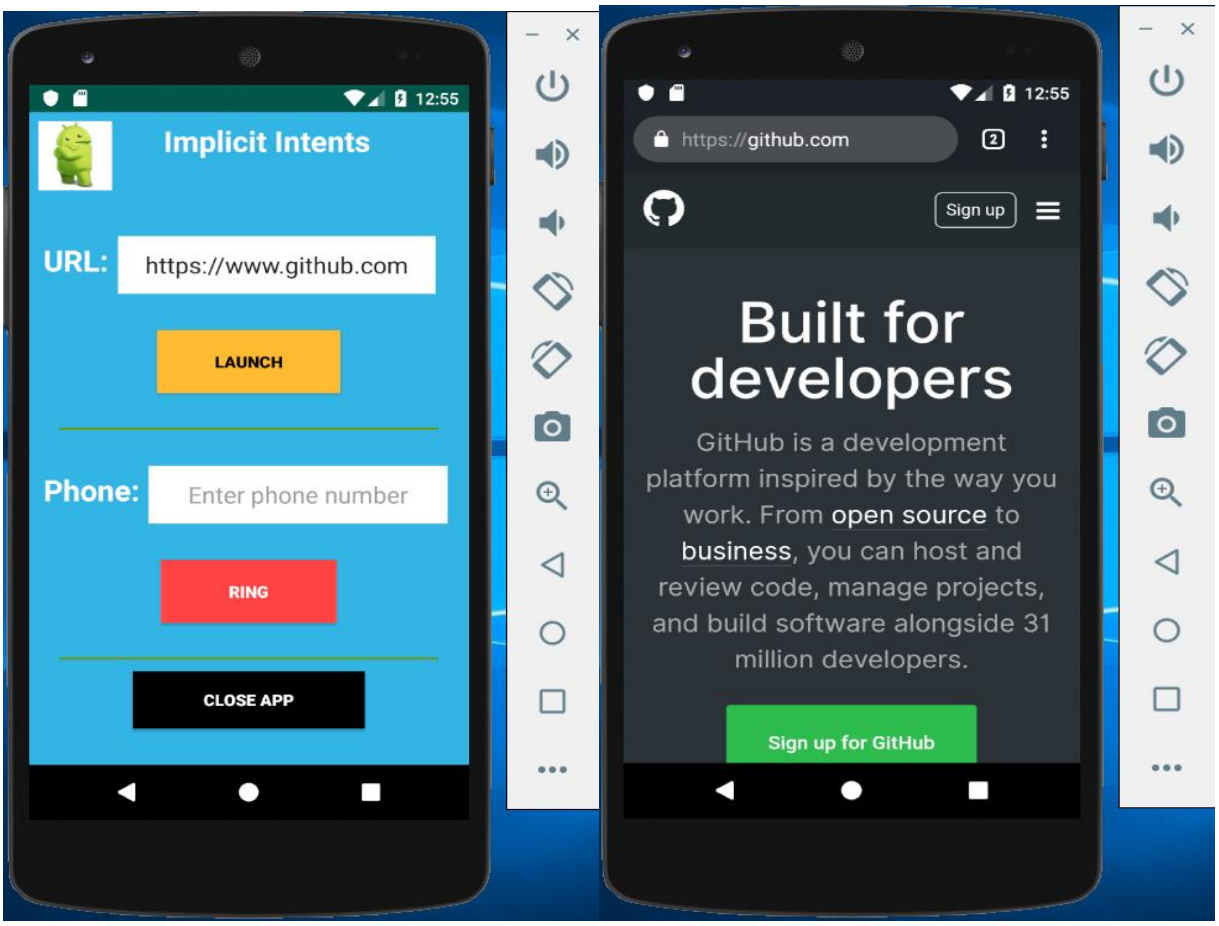
Implicit Intents to invoke a Web link:

The web link invocation with implicit intents can be achieved using “ACTION_VIEW” trigger. It is mainly used to display the data to the user. When used with URI or a web address, it opens the desired web page and displays all the information contained on that web page. On clicking the launch button, the webpage will be opened. I have attached the screenshots displaying ***http*** and ***https*** web pages

Web link : http://www.sjsu.edu



Web link : https://www.github.com



Code for invoking the web page with implicit intent:

ImplicitIntentActivity.java | activity_main.xml | AndroidManifest.xml

```
14
15  public class ImplicitIntentActivity extends Activity {
16
17      @Override
18      protected void onCreate(Bundle savedInstanceState) {
19          super.onCreate(savedInstanceState);
20          setContentView(R.layout.activity_main);
21          Button openURL = (Button)findViewById(R.id.openURL);
22          openURL.setOnClickListener(new View.OnClickListener() {
23              @Override
24              public void onClick(View v) {
25                  EditText websiteURL = (EditText)findViewById(R.id.websiteURL);
26                  String webPage = websiteURL.getText().toString();
27                  Intent webIntent = new Intent(Intent.ACTION_VIEW, Uri.parse(webPage));
28                  startActivity(webIntent);
29
30              }
31          });
```

Implicit Intents to invoke a Phone call:

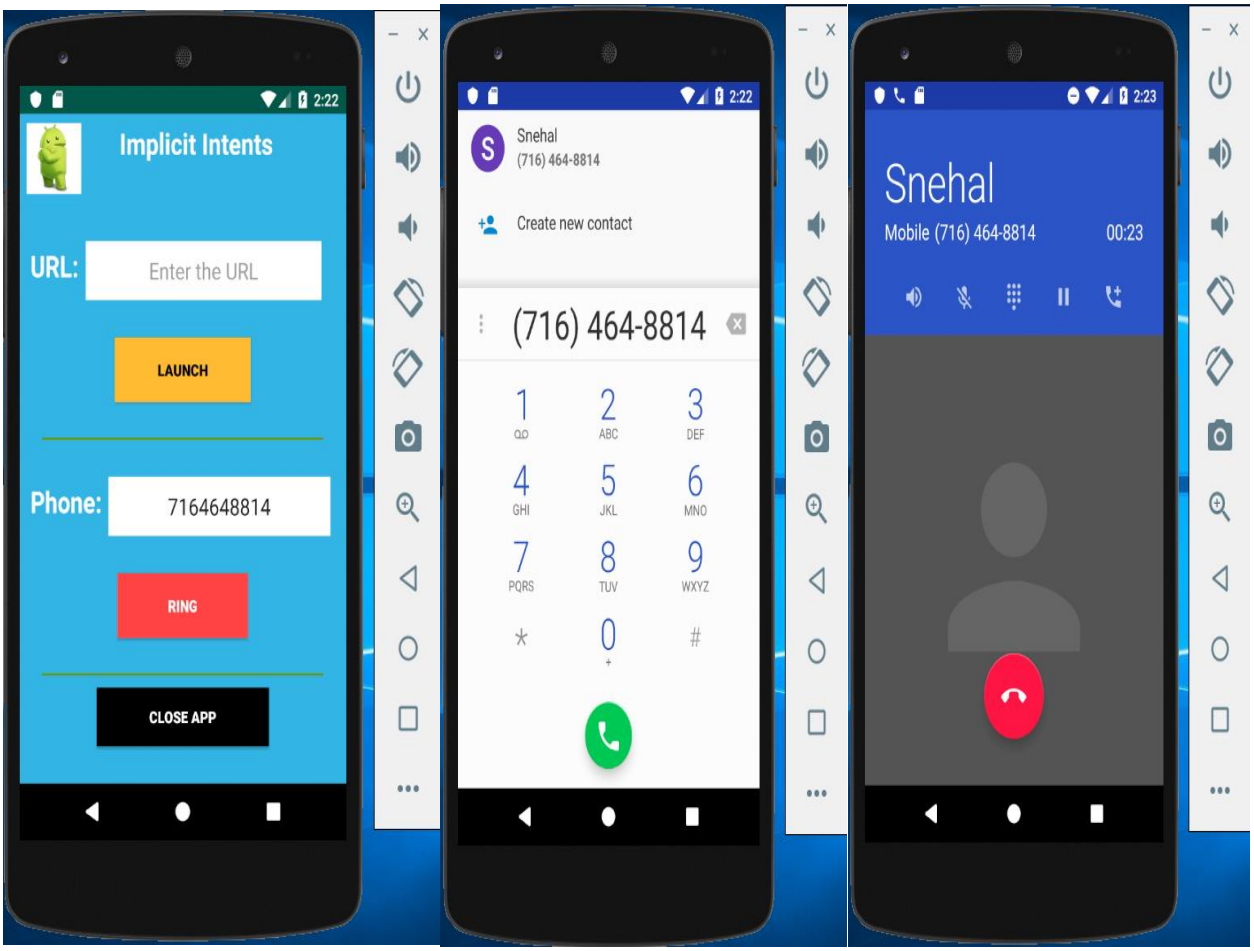
The phone calls can be invoked with implicit intents using “ACTION_DIAL” or “ACTION_CALL”.

ACTION_DIAL : It launches the phone application and displays the phone number in the dialer with no need to request permission from the user before calling.

ACTION_CALL: It makes the phone call directly from within your application and needs the permission from the user if its not already granted. The permission to set up the phone call is given in AndroidManifest.xml file.

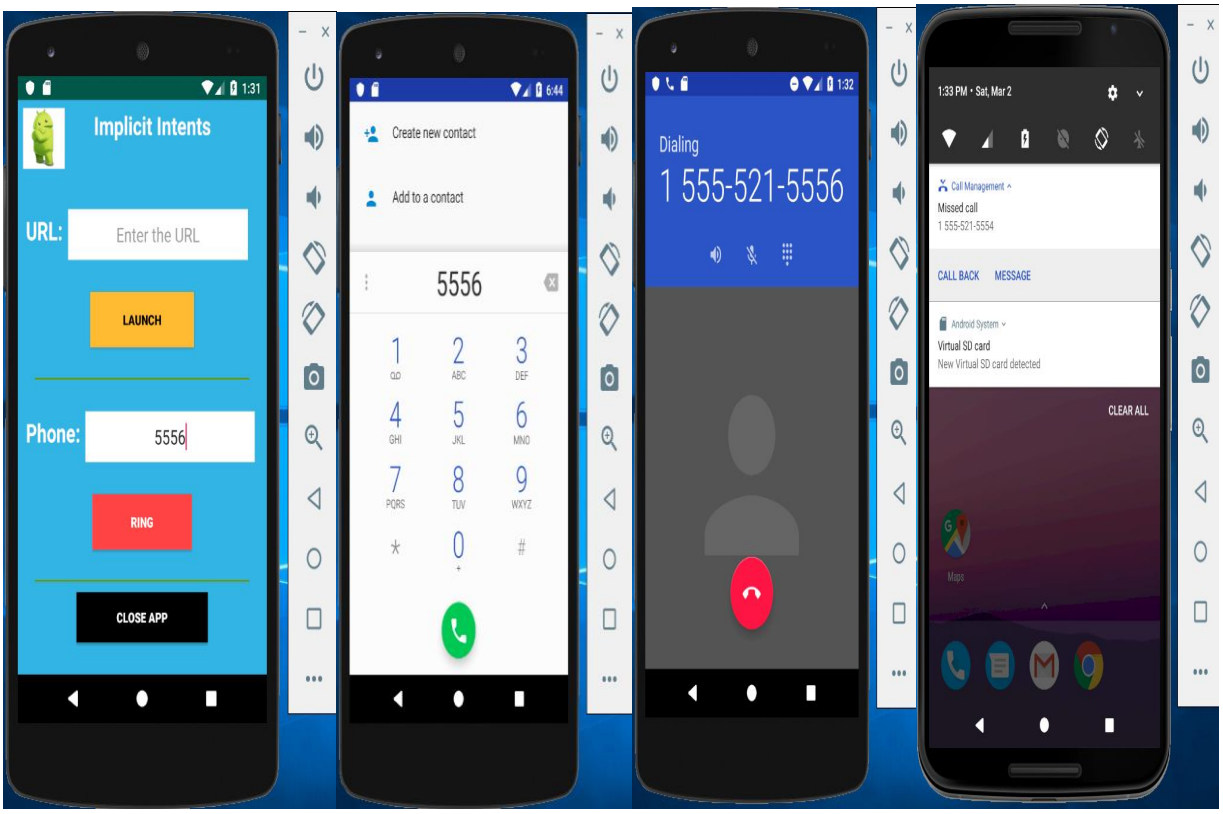
Note: I have used ACTION_DIAL to invoke the phone calls. When the “Ring” button is clicked, the dial pad is opened and user can manually place a phone call or press the back button and come back to application.

Using single emulator to test phone call/dial functionality:

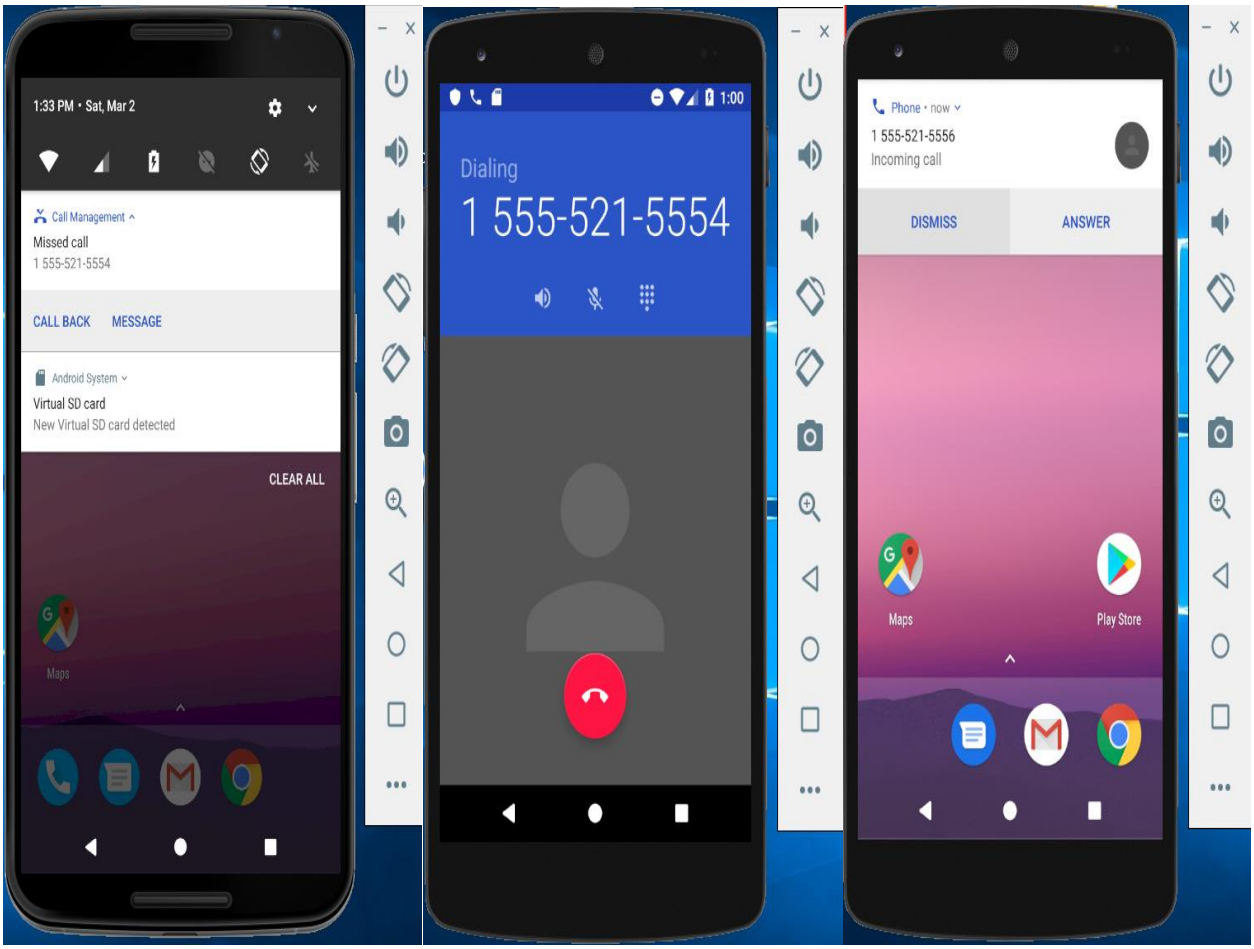


Using two emulators to test the phone call/dial functionality:

1. Launch an emulator directly from the AVD manager Tools -> Android -> AVD Manager.
2. Note down the port number of this emulator instance.
3. Run the implicit-intent application in android studio and choose the other emulator instance.
4. Enter the port number of first emulator instance in the phone calling text box instead of a real phone number and click on the ring button. The dial pad is opened and if user calls the number, the emulator shows that the phone call is started and the other emulator should receive the call.
5. We can click answer or dismiss to receive or reject the call.



Call Back from the first emulator:



Code to invoke phone calls using ACTION_DIAL:

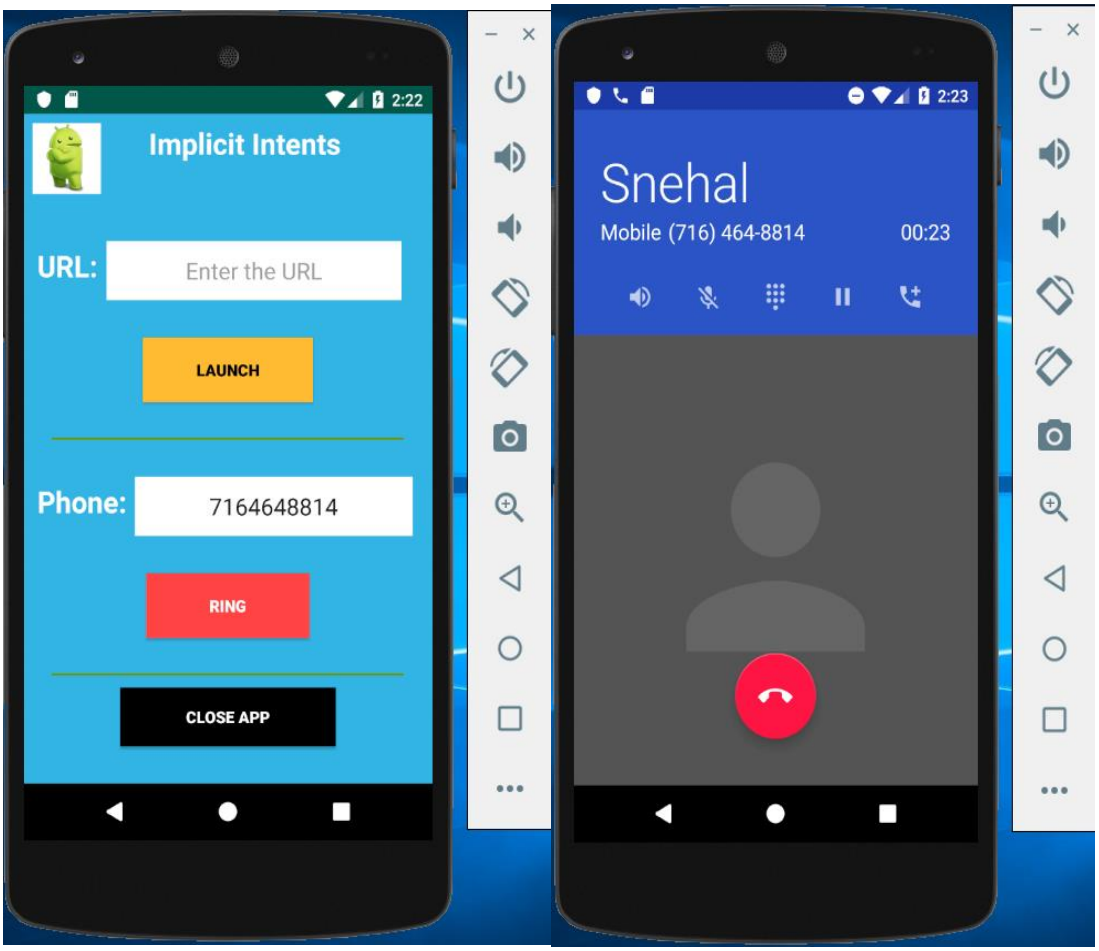
ImplicitIntentActivity.java | activity_main.xml | AndroidManifest.xml

```
32
33          Button dialNumber = (Button)findViewById(R.id.dialNumber);
34          dialNumber.setOnClickListener(new View.OnClickListener(){
35              @Override
36              public void onClick(View v) {
37                  EditText phoneNumber = (EditText)findViewById(R.id.phoneNumber);
38                  Uri number = Uri.parse("tel:"+phoneNumber.getText().toString());
39                  Intent callIntent = new Intent(Intent.ACTION_DIAL, number);
40                  startActivity(callIntent);
41              }
42          });
43      }
```

Invoke Phone calls using ACTION_CALL:

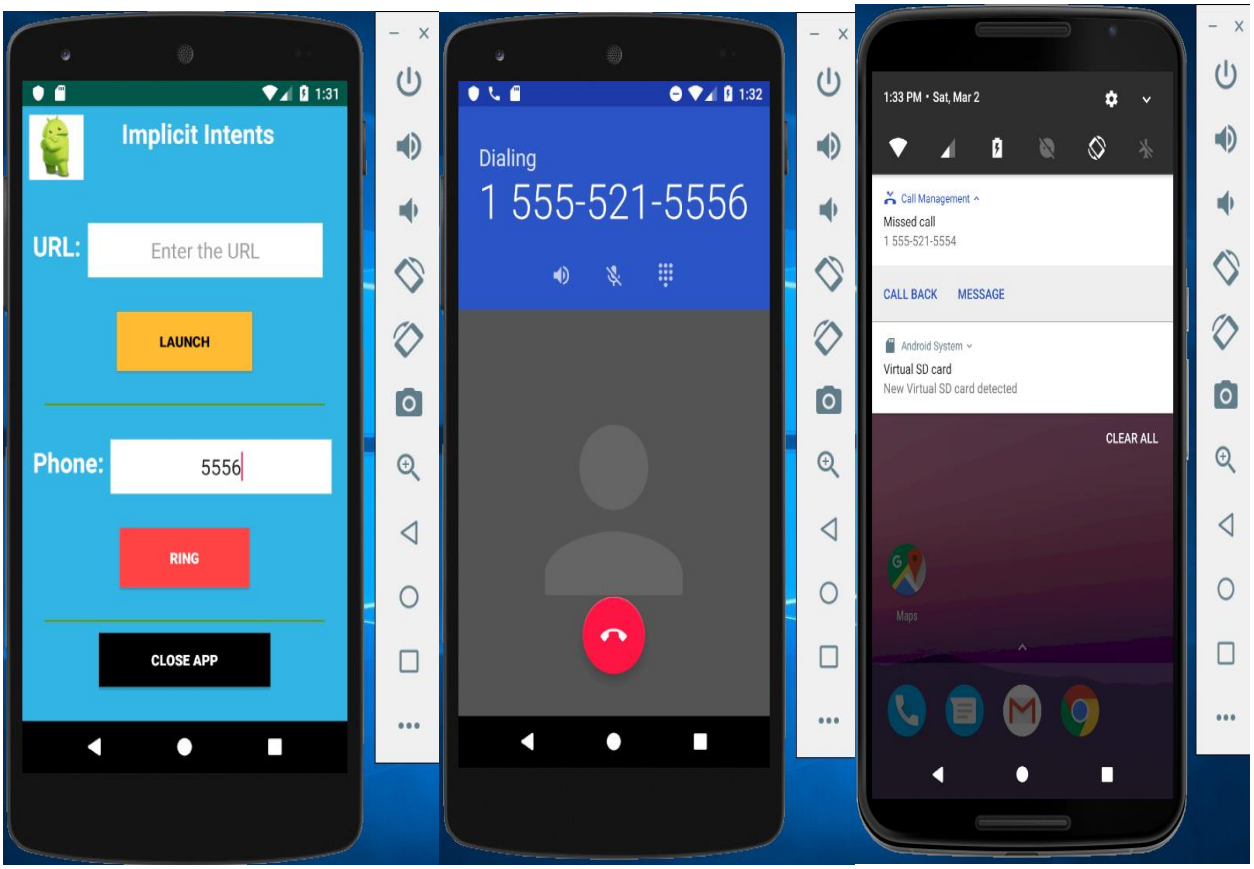
I have also implemented the phone call invocation using ACTION_CALL. In this case, when user enters the number in text box and click on “Ring” button, it automatically calls the specified number without opening the dial pad as shown below.

Using single emulator:



Using two emulators to test the phone call functionality:

When the number is entered, on clicking the Ring button, the call gets automatically placed and the other emulator receives the incoming call.



Code to invoke phone calls using ACTION_CALL:

```
        Button dialNumber = (Button)findViewById(R.id.dialNumber);
        dialNumber.setOnClickListener(new View.OnClickListener(){
            @Override
            public void onClick(View v) {
                if(isPermissionGranted()){
                    call_action();
                }
            }
        });
    }

    public void call_action(){
        EditText phoneNumber = (EditText)findViewById(R.id.phoneNumber);
        Uri number = Uri.parse("tel:"+phoneNumber.getText().toString());
        Intent callIntent = new Intent(Intent.ACTION_CALL, number);
        startActivity(callIntent);
    }
```

```
public boolean isPermissionGranted(){
    if (Build.VERSION.SDK_INT >= 23) {
        if (checkSelfPermission(android.Manifest.permission.CALL_PHONE)
                == PackageManager.PERMISSION_GRANTED) {
            Log.v( tag: "TAG", msg: "Permission for calling is granted");
            return true;
        } else {

            Log.v( tag: "TAG", msg: "Permission for calling is revoked");
            ActivityCompat.requestPermissions( activity: this, new String[]{Manifest.permission.CALL_PHONE}, requestCode:
            return false;
        }
    }
    else { //permission is automatically granted on sdk<23 upon installation
        Log.v( tag: "TAG", msg: "Permission for calling is granted");
        return true;
    }
}
```

```
@Override
public void onRequestPermissionsResult(int requestCode, String permissions[], int[] grantResults) {
    switch (requestCode) {

        case 1: {

            if (grantResults.length > 0
                    && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
                Toast.makeText(getApplicationContext(), text: "Permission granted", Toast.LENGTH_SHORT).show();
                call_action();
            } else {
                Toast.makeText(getApplicationContext(), text: "Permission denied", Toast.LENGTH_SHORT).show();
            }
            return;
        }
    }
}
```

Added the phone call permission in AndroidManifest.xml file as shown below:

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3       package="edu.sjsu.android.implicit_intents">
4       <uses-permission android:name="android.permission.CALL_PHONE" />
5
6       <application
```

Note: I have commented the code for ACTION_CALL in the zip file for submission but implemented both ACTION_DIAL and ACTION_CALL to invoke the phone calls for this application.

Close Application Button:

On clicking the close application button, the application will be closed and the control will be navigated to the home screen of android as shown below: