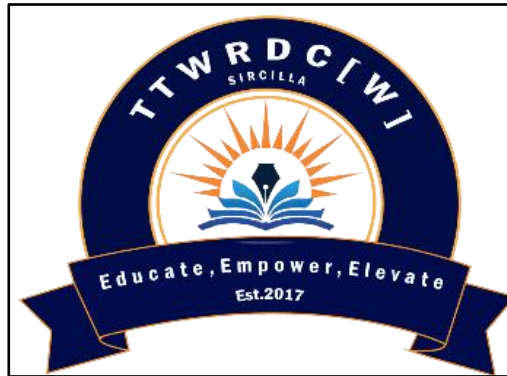A Study

on

"CONTACT APP ANDRIOD"

Submitted by

1. A. Supriya (17077104402001)

2. P. Maheshwari (17077104468014)

3. J. Swetha (17077104402005)

4. R. Swetha (17077104468016)

5. Sushmitha (17077104468003)

Under the guidance

Of

B. Himabindu, MCA, Lecturer

Department of Computer Science& Applications

Department of Computer Science & Applications

Telangana Tribal Welfare Residential Degree College(W), Rajanna Sircilla.

(Affiliated to Satavahana University)

(2019-2020)

DECLARATION

I hereby, declare that this project entitled **" CONTACT APP ANDRIOD "** have completed successfully towards the partial fulfillment for the award of the degree " **BACHELOR OF COMPUTER SCIENCE & APPLICATIONS** " from " **TELANGANA TRIBAL WELFARE RESIDENTIAL DEGREE COLLEGE FOR WOMEN, RAJANNA SIRICILLA** .This is the bonafide work undertaken by me which is not submitted to any other university or institution for the award of any degree / diploma.

DATE :

PLACE: RAJANNA SIRICILLA

Name of the Students

1. A. Supriya (17077104402001)
2. P. Maheshwari (17077104468014)
3. J. Swetha (17077104402005)
4. R. Swetha (17077104468016)
5. Sushmitha (17077104468003)

Telanagana Tribal Welfare Residential Degree College for Women, Thangallapally, Rajanna Siricilla
(Affiliated to Satavahana University)

CERTIFICATE

This is to certify that a study on project titled **" CONTACT APP ANDRIOD "**. This project is submitted by A. Supriya (17077104402001), P. Maheshwari (17077104468014), J. Swetha (17077104402005), R. Swetha (17077104468016) and Sushmitha (17077104468003) from **Telanagana Tribal Welfare Residential Degree College for Women, Siricilla** under the guidance of B.Himabindu, Lecturer, Department of Computer Science and Applications. This has not been submitted to any other institute or university for the award of any degree.

Himabindu

Signature of the guide

K. Rajani

Principal
Principal
TTWRDC(W)SIRCILLA
Dist: Rajanna Sircilla

Contact App Andriod

Objective

To build an Android application for managing contacts using Android Studio, Java and SQLite3.

Project Context

Java is one of the most popular programming languages in the world. Java classes run on Android Runtime (ART), a specialized virtual machine.

The Android Studio IDE is the best way you can build an android application with utmost ease. It is based on IntelliJ IDEA and also the official environment for Google's operating
system. Features like real-time profilers, intelligent code editor and fast emulator makes the work in Android Studio fun. **Android SDK** is the official development kit for Android App
development.

The main aim is to build a Contact Application which can be used by a user to make calls, store contact numbers in their respective local storage and also provide a simple way to delete them too. This will be kind of a basic phone book application. This project will also include brief usage of SQLite3 database for local storage of data.
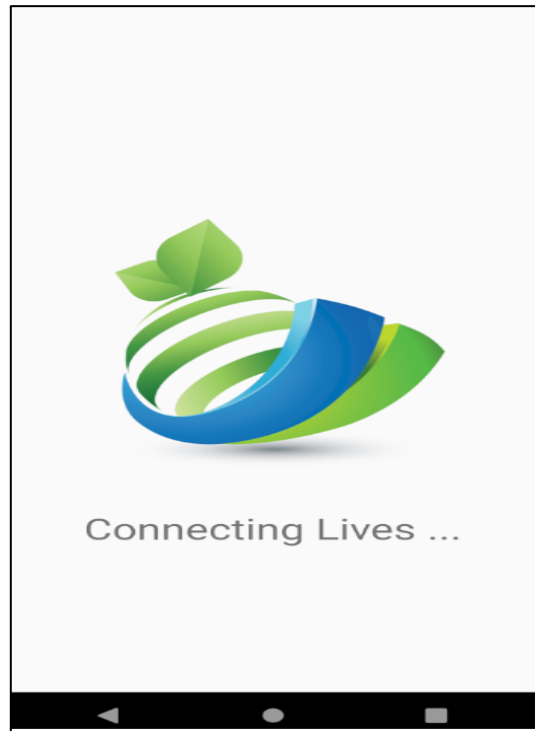
Project Stages

This project consists of the following stages: contact-android-app-sequence-diagram
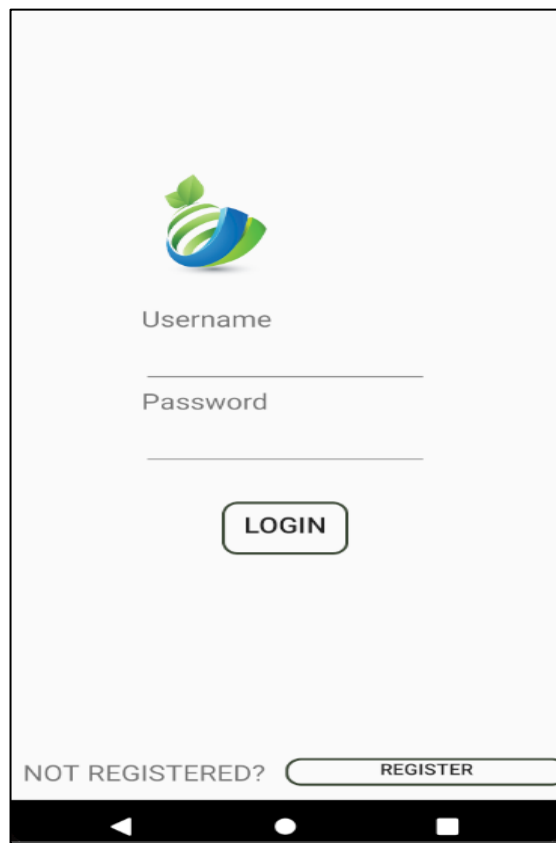
High-Level Approach

- Creating the loading screen with Splash time.
- Creating the registration and login activities.
- Connecting the database and storing the registration details in the database.
- In Login Activity, checking if the inserted data matches with database or not.
- Creating custom recycler view to show the data present in the database.
- Adding a Call function (for which the call access permission should be given.
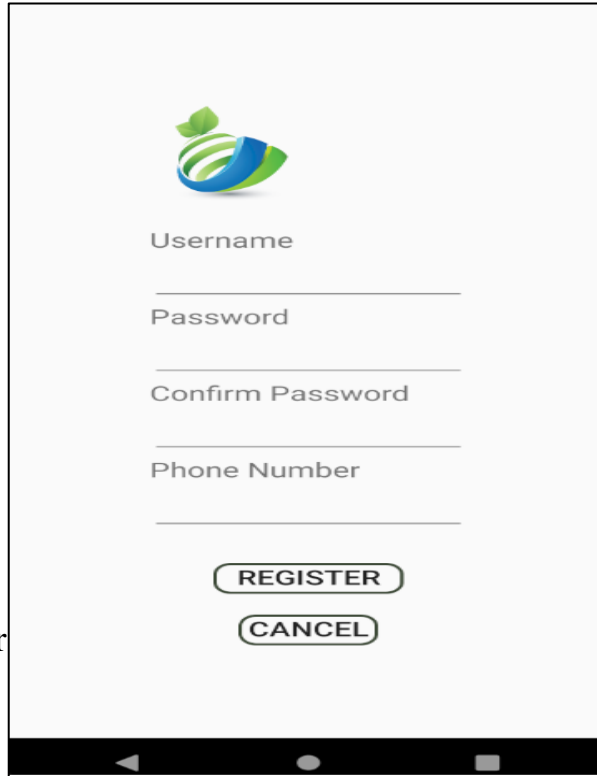
Sample Application Design

- Loading Page

- Login Page

- Registration Page

- Recycler

- Contact Details and Calling
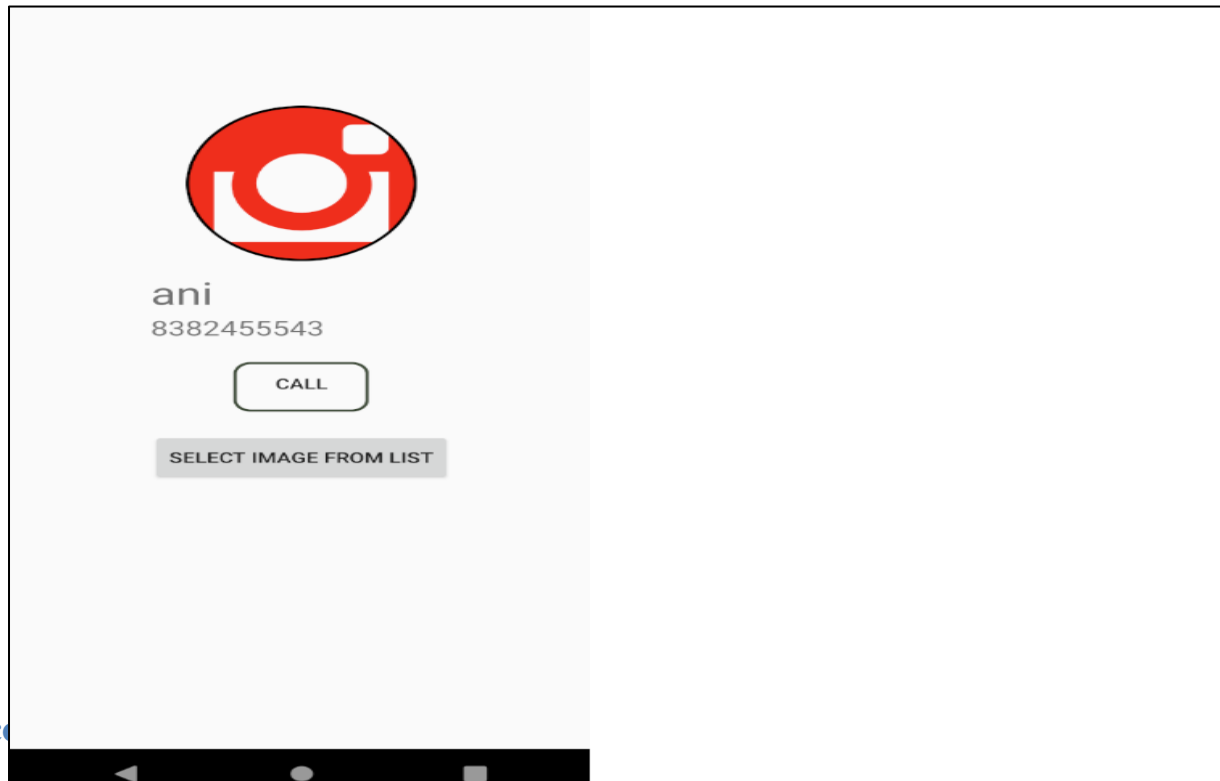
Task 1

Getting the c

This task is all about the concepts that will be needed in building our application. Checkout the **Requirements** given below for more details.

Requirements
- Understanding how activities are rendered in the application. Also, what do we mean by activities in Android?
- Understanding the concept of Activity Life Cycle.
- Learning about Intents in Android, which will be your best friend in Android application development. It will help you communicate within you activities.

Bring it On!
- Try creating the basic layout of your application, may be try to draw out the design of your app. Develop a basic design layout, or you can go as creative as you want. [**Note**: You can use the sample designs as well.]
- XML is always there to make your designs easier and easy to visualize too. Use it!

Expected Outcome

The main purpose of this task is for you to have a basic knowledge of the life cycle methods of an Android application and how to communicate and transfer data between your
activities.

Task 2

Setting up the main project

It's time to work on the actual project. We'll start off by creating the basic structure of the project. [**NOTE**: The file structure is of a typical Android project. You are free to refer to
official documentations for the file structure.]

Requirements

- Add all necessary dependencies for your project in the build.gradle file. Gradle is a build system commonly used in all java applications to automate building, testing and deployment. build.gradle contains scripts to automate several tasks like simply copying some file from one directory to other. It contains compiledSdkVersion, buildTypes and dependencies. The following Dependencies need to be added:
  - appcompat- maven dependency for using various frameworks
  - recyclerview- dependency for using the recycler view in our app
  - cardview- for using cardview class with the viewholder class
  - constraint-layout- for designing in constraint layout

```
dependencies {
    implementation 'com.android.support:appcompat-v7:28.0.0'
    implementation 'com.android.support:recyclerview-v7:28.0.0'
```

```
        implementation 'com.android.support:cardview-v7:28.0.0'
        implementation
'com.android.support.constraint:constraint-layout:2.0.4'
        implementation 'com.android.support:design:28.0.0-rc01'
        implementation 'de.hdodenhof:circleimageview:3.0.0'
}
```

- Now after setting the proper dependencies, we can see main activity being present by default in our application. This main activity class will be the first function to be executed when our application will be launched. We can create new activities for different pages of your app under `com.example.[appname]` by creating new classes under this directory.
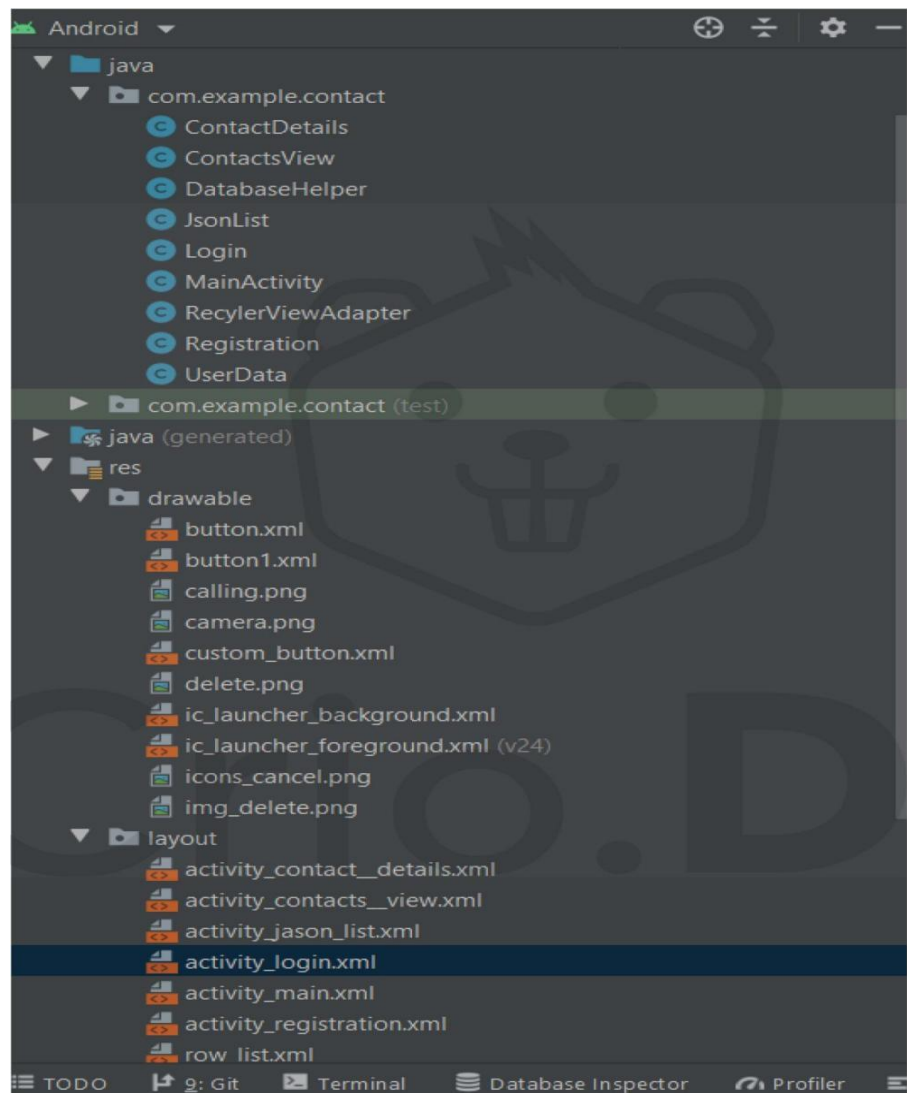
- Each activity can be designed using xml and the component .xml files can be found in the layout folder of the res directory. In the activity.xml file we can design our activity screen using XML and drag & drop methods.

- Always remember to map the elements in the .xml file to your class activity files.

- The drawable folder under res (resource) directory will contain all your images, designs etc. files needed for the frontend.

- Now go on creating as many activities you will be using in your app. Always make sure that the main activity will regulate the flow of the application i.e., which activity will be followed next. Intents will be come handy here.

References
- Dependency in build.gradle
- Creating new Activities
- Layouts & Views in Android
- Intents and uses

Expected Outcome

You should be able to create a basic layout of your app with few activities. [**NOTE**: For this task we are just expecting empty files for the classes. The content of these files are discussed in the next sections.]

Task 3

Registering Users & Authenticating Login

The basic idea of our application is that multiple users can register on the application and store details of various contacts and access the same on login.
In this milestone, we'll be
accepting user data on registration which needs to be fed to the database and then while
login the user will be authenticated when the user enters a valid input in the placeholders
and clicks on the Login button.

Requirements

- Creating placeholders for the name, username, mobile number, password for the user input in the Registration activity. The Login activity will require only username and password. Intents can be used here to take the values of these placeholders and transfer it across our list of activity classes.

- Create a UserData class which will contain the way in which data will be stored in our database. Idea is to have objects of users in which each object shall contain username and mobile number.

- Once we are done with this, we have to connect our database to our application where user data will be stored. Create a DatabaseHelper class which will be used to connect SQLite3. SQLiteOpenHelper is used to create an object to create tables, open or manage the database. Here also define the schema of the database by executing a create table query in the onCreate() method in DatabaseHelper class.

  - Now we need the Registration activity class to call our DatabaseHelper class using an object of SQLiteHelper to send the data inserted in the placeholders to DatabaseHelper class where these will be fed to the database. This method shall
  contain the username, mobile number, password as the arguments.

  - Create an insert method in DatabaseHelper class which will receive the arguments from Registration activity and feed it to the database using ContentValues. You
  can have the return type of this insert method as Boolean to check whether the data has been successfully inserted or not. ContentValues is a map like class that matches string key and inserts the data given into it as arguments into the table

- Now in the Login activity class we have fields for username and password. We need to check that if this user is present in our database or not and then if there is a user
  present we need to authenticate that the user has entered correct password or not.

  - Create a check_username method in DatabaseHelper class which receives the username as argument to check if it is present in the database or not.

- Create a match method in our DatabaseHelper class which receives username and password as arguments and checks if the password matches with the password present in the database.

  - Here in the match method we have to use cursor for iterating through the table stored in the database. Cursor basically contains the result of a query made against a database in android. Use the getCount() method of Cursor which returns the

number of matches, if it returns zero hence no matches found and user is not present in the database.

- Also check out these commands for creating and updating the tables of your database.

  - Executing create table - onCreate() This will contain the database query executing command for creating a table.

  - Updating table - onUpgrade() This will contain the command for updating data in our database.

References
- SQLite helper
- SQLite Basic Commands
- Accessing Database using cursor
- ContentValues

Bring it On!
- How to verify whether the classes coded in this milestone will work as expected or not? For testing them, we can write unit test cases using **JUnit**. Try to do so.

Expected Outcome

You should be able to write the necessary content for the Registration and Login activities. The data coming in from the Registration activity should be passed to the DatabaseHelper which feeds it to the database. The Login activity shall call an authentication method of DatabaseHelper class to authenticate the user.

Task 4

Building the recycler view adapter and populating the list

In this task the aim is to build a custom recycler view for our list in which the data present in the database will be shown.

Requirements
- We have to create a custom recycler view class. RecyclerView is a class under ViewGroup which helps in building and displaying large sets of data efficiently. In our case, we will use it to build a list of items which shall display the username, phone
  number and a delete button for each element in the list. ViewHolder class provides all the functionality for each list items. To implement a custom

RecyclerView we would need a Custom Adapter also. Adapter associates the data with the ViewHolder class.

- Before moving ahead we need to create a method getAllContacts() in the DatabaseHelper class which returns all the data present in the database by iterating through it using Cursor. This is first needed to get the data in an ArrayList. Then we move onto creating our recycler view.

- Now create a ContactsView class which is basically the RecyclerView wrapper class. This class calls on the RecyclerViewAdapter class with arguments as the ArrayList of UserData. This also defines the layout of the list i.e. whether you want a vertical list or a horizontal list. It contains two methods -
  - onCreate()- calls the RecyclerViewAdapter class and sends the ArrayList of UserData as argument. It also manages the layout of the list view. This method will call the next method loadContactList().

  - loadContactList() - This method should have an instance of the database which is used to call using the DatabaseHelper object to get the list of data stored in the
database. It also calls the getAllContacts() to populate the adapter list in the next step.

- Create a custom RecyclerViewAdapter class which will denote each row of the database. In this class we should have an ArrayList of UserData to receive the same from ContactsView. Each of the views in the list will contain username, phone number and a delete button. Each of the list view item is represented by RecyclerViewAdapter.

- This class has various methods like
  - onCreateViewHolder()- used to populate the list receiving data from UserData.

  - getItemCount()- returns the number of rows in the list.

  - ViewHolder()- contains a database instance and deleting method. The delete button is used with a onClickListener event to delete the current list item when this
button is clicked.

  - Create a deleteContact() method in the DatabaseHelper class that receives a UserData to be deleted and deletes that data from the database. ##

References

- Recycler View
- Custom Views
- ViewHolder
- Custom Adapter

Bring it On!

- As discussed above we can change the layout of the recycler view which is vertical by default. You can try to make a horizontal scrollable list. Try to do so.

Expected Outcome

By the end of this milestone, you should have a working RecyclerView list populated with the data in the database with each item having username, mobile number & delete button.

Task 5

Adding the Call & Image capture method

By now we have almost come to the end of our project. The only part left is to create a details page from which we can make a call and also store image in the contact details. We will also be using a circular card view for some customization to our details page.

Displaying an image is done using ImageView but it is difficult to create circular view with it. Hence we will use card view library to make our task simple. The activity_contact__details file corresponding to the ContactDetails class can also be used to modify the changes.

Requirements

- Create a ContactDetails activity class along with call button, mobile number and profile photo.

- Add the permission for using the camera and calling option in the details activity. The camera and call permissions have to be allowed manually from the settings of the android device.

- Create two different onClickListener methods, one for camera access and clicking the image, while the other is for making a calling method which is implemented by using Intent methods for performing the call function.

```
public static final int CAMERA_REQUEST=9999;

// mention these in the onclick listener functions
Intent callintent = new Intent(Intent.ACTION_CALL); // for call
Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE); // for camera
```

- Now you also have to store this in the profile image, for which we have to convert our image into bitmap and then store. Before inserting into database, you need to convert your Bitmap image into byte array first then apply it using database query.
- This bitmap image has to be stored in the database using an instance of it. It should call a method InsertImage() in DatabaseHelper class which takes in the bitmap image and converts it into a byte array. The InsertImage() method inserts the image into the

  database using the two arguments byte array and username. Query commands are

  executed here. It uses the Cursor again to search for the corresponding username and then with the help of ContentValues inserts into the database.

References
- Card view
- Storing image in database
- Calling method
- Camera Method

Bring it On!
- You can try showing the Image in each item of the recycler view list in the ContactView

  activity. [**NOTE:** The Adapterhas to be modified to store an image view.]
- By now the application should be ready. Build an APK and take the app for a test run.
- Customize the application in your own flavour, suitable enough to be published on the

  Google Play Store.

Expected Outcome

You should be able to develop the details page along with the basic functionalities of calling function and image capture function.