APPENDICES

APPENDIX-I

SOURCE CODE

A-I.1Working with the Java and PHP Codes

MainActivity.java

```
package net.simplifiedcoding.firebasecloudmessaging;

public class MainActivity extends AppCompatActivity implements View.OnClickListener {


    //defining views
    private Button buttonSendPush;
    private Button buttonRegister;
    private EditText editTextEmail;
    private ProgressDialog progressDialog;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        //getting views from xml
        editTextEmail = (EditText) findViewById(R.id.editTextEmail);
        buttonRegister = (Button) findViewById(R.id.buttonRegister);
        buttonSendPush = (Button) findViewById(R.id.buttonSendNotification);
        //adding listener to view
        buttonRegister.setOnClickListener(dis);
        buttonSendPush.setOnClickListener(dis);
    }
```

```
//storing token to mysql server
private void sendTokenToServer() {
    progressDialog = new ProgressDialog(dis);
    progressDialog.setMessage("Registering Device...");
    progressDialog.show();

    final String token = SharedPrefManager.getInstance(dis).getDeviceToken();
    final String email = editTextEmail.getText().toString();
     if (token == null) {
        progressDialog.dismiss();
        Toast.makeText(dis, "Token not generated", Toast.LENGTH_LONG).show();
        return;
    }
    StringRequest stringRequest = new StringRequest(Request.Method.POST, EndPoints.URL_REGISTER_DEVICE,
            new Response.Listener<String>() {
                @Override
                public void onResponse(String response) {
                    progressDialog.dismiss();
                    try {
                        JSONObject obj = new JSONObject(response);
                        Toast.makeText(MainActivity.dis, obj.getString("message"), Toast.LENGTH_LONG).show();
                    } catch (JSONException e) {
                        e.printStackTrace();
                    }                    }                    },
            new Response.ErrorListener() {
                @Override
                public void onErrorResponse(VolleyError error) {
                    progressDialog.dismiss();
                    Toast.makeText(MainActivity.dis, error.getMessage(), Toast.LENGTH_LONG).show();
                }
            }) {
```

```
        @Override
        protected Map<String, String> getParams() throws AuthFailureError {
            Map<String, String> params = new HashMap<>();
            params.put("email", email);
            params.put("token", token);
            return params;
        }
    };
    RequestQueue requestQueue = Volley.newRequestQueue(this);
    requestQueue.add(stringRequest);
}

@Override
public void onClick(View view) {
    if (view == buttonRegister) {
        sendTokenToServer();
    }
    //starting send notification activity
    if(view == buttonSendPush){
        startActivity(new Intent(this, ActivitySendPushNotification.class));
    }
}
}
```

ActivitySendPushNotification.java

```
package net.simplifiedcoding.firebasecloudmessaging;

public class ActivitySendPushNotification extends AppCompatActivity implements
RadioGroup.OnCheckedChangeListener, View.OnClickListener {

    private Button buttonSendPush;
    private RadioGroup radioGroup;
```

```
private Spinner spinner;
private ProgressDialog progressDialog;
private EditText editTextTitle, editTextMessage, editTextImage;
private boolean isSendAllChecked;
private List<String> devices;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_send_push_notification);
    radioGroup = (RadioGroup) findViewById(R.id.radioGroup);
    spinner = (Spinner) findViewById(R.id.spinnerDevices);
    buttonSendPush = (Button) findViewById(R.id.buttonSendPush);
    editTextTitle = (EditText) findViewById(R.id.editTextTitle);
    editTextMessage = (EditText) findViewById(R.id.editTextMessage);
    editTextImage = (EditText) findViewById(R.id.editTextImageUrl);
    devices = new ArrayList<>();
    radioGroup.setOnCheckedChangeListener(this);
    buttonSendPush.setOnClickListener(this);
    loadRegisteredDevices();
}

//method to load all the devices from database
private void loadRegisteredDevices() {
    progressDialog = new ProgressDialog(this);
    progressDialog.setMessage("Fetching Devices...");
    progressDialog.show();
    StringRequest stringRequest = new StringRequest(Request.Method.GET, EndPoints.URL_FETCH_DEVICES,
            new Response.Listener<String>() {
                @Override
                public void onResponse(String response) {
```

```
                progressDialog.dismiss();
                JSONObject obj = null;
                try {
                    obj = new JSONObject(response);
                    if (!obj.getBoolean("error")) {
                        JSONArray jsonDevices = obj.getJSONArray("devices");
                        for (int i = 0; i < jsonDevices.length(); i++) {
                            JSONObject d = jsonDevices.getJSONObject(i);
                            devices.add(d.getString("email"));
                        }
                        ArrayAdapter<String> arrayAdapter = new ArrayAdapter<String>(
                                ActivitySendPushNotification.this,
                                android.R.layout.simple_spinner_dropdown_item,
                                devices);
                        spinner.setAdapter(arrayAdapter);
                    }
                } catch (JSONException e) {
                    e.printStackTrace();
                }
            }
        },
        new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
            }
        }) {
    };
    MyVolley.getInstance(this).addToRequestQueue(stringRequest);
}

  //this method will send the push
```

```
    //from here we will call sendMultiple() or sendSingle() push method



    //depending on the selection
    private void sendPush() {
        if (isSendAllChecked) {
            sendMultiplePush();
        } else {
            sendSinglePush();
        }
    }


    private void sendMultiplePush() {
        final String title = editTextTitle.getText().toString();
        final String message = editTextMessage.getText().toString();
        final String image = editTextImage.getText().toString();


        progressDialog.setMessage("Sending Push");
        progressDialog.show();


        StringRequest stringRequest = new StringRequest(Request.Method.POST,
EndPoints.URL_SEND_MULTIPLE_PUSH,
                new Response.Listener<String>() {
                    @Override
                    public void onResponse(String response) {
                        progressDialog.dismiss();


                        Toast.makeText(ActivitySendPushNotification.this, response, Toast.LENGTH_LONG).show();
                    }
                },
                new Response.ErrorListener() {
```

```
            @Override
            public void onErrorResponse(VolleyError error) {
            }
        }) {
        @Override
        protected Map<String, String> getParams() throws AuthFailureError {
            Map<String, String> params = new HashMap<>();
            params.put("title", title);
            params.put("message", message);
            if (!TextUtils.isEmpty(image))
                params.put("image", image);
            return params;
        }
    };
    MyVolley.getInstance(this).addToRequestQueue(stringRequest);
}

private void sendSinglePush() {
    final String title = editTextTitle.getText().toString();
    final String message = editTextMessage.getText().toString();
    final String image = editTextImage.getText().toString();
    final String email = spinner.getSelectedItem().toString();

    progressDialog.setMessage("Sending Push");
    progressDialog.show();

    StringRequest stringRequest = new StringRequest(Request.Method.POST,
EndPoints.URL_SEND_SINGLE_PUSH,
            new Response.Listener<String>() {
                @Override
                public void onResponse(String response) {
```

```
                progressDialog.dismiss();


                Toast.makeText(ActivitySendPushNotification.this, response, Toast.LENGTH_LONG).show();
            }
        },
        new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
            }
        }) {
    @Override
    protected Map<String, String> getParams() throws AuthFailureError {
        Map<String, String> params = new HashMap<>();
        params.put("title", title);
        params.put("message", message);


        if (!TextUtils.isEmpty(image))
            params.put("image", image);
        params.put("email", email);
        return params;
    }
};


MyVolley.getInstance(this).addToRequestQueue(stringRequest);
}
@Override
public void onCheckedChanged(RadioGroup radioGroup, int i) {
    switch (radioGroup.getCheckedRadioButtonId()) {
        case R.id.radioButtonSendAll:
            isSendAllChecked = true;
            spinner.setEnabled(false);
```

```
            break;
        case R.id.radioButtonSendOne:
            isSendAllChecked = false;
            spinner.setEnabled(true);
            break;
    }
  }

  @Override
  public void onClick(View view) {
    //calling the method send push on button click
    sendPush();
  }
}
```

MyNotificationManager.java

```
package net.simplifiedcoding.firebasecloudmessaging;
public class MyNotificationManager {
  public static final int ID_BIG_NOTIFICATION = 234;
  public static final int ID_SMALL_NOTIFICATION = 235;
  private Context mCtx;
  public MyNotificationManager(Context mCtx) {
    this.mCtx = mCtx;
  }

  //the method will show a big notification with an image
  //parameters are title for message title, message for message text, url of the big image and an intent that will open
  //when you will tap on the notification
  public void showBigNotification(String title, String message, String url, Intent intent) {
```

```
    PendingIntent resultPendingIntent =
            PendingIntent.getActivity(
                    mCtx,
                    ID_BIG_NOTIFICATION,
                    intent,
                    PendingIntent.FLAG_UPDATE_CURRENT
            );
    NotificationCompat.BigPictureStyle bigPictureStyle = new NotificationCompat.BigPictureStyle();
    bigPictureStyle.setBigContentTitle(title);

    bigPictureStyle.setSummaryText(Html.fromHtml(message).toString());
    bigPictureStyle.bigPicture(getBitmapFromURL(url));
    NotificationCompat.Builder mBuilder = new NotificationCompat.Builder(mCtx);
    Notification notification;
    notification = mBuilder.setSmallIcon(R.mipmap.ic_launcher).setTicker(title).setWhen(0)
            .setAutoCancel(true)
            .setContentIntent(resultPendingIntent)
            .setContentTitle(title)
            .setStyle(bigPictureStyle)
            .setSmallIcon(R.mipmap.ic_launcher)
            .setLargeIcon(BitmapFactory.decodeResource(mCtx.getResources(), R.mipmap.ic_launcher))
            .setContentText(message)
            .build();

    notification.flags |= Notification.FLAG_AUTO_CANCEL;

    NotificationManager notificationManager = (NotificationManager)
mCtx.getSystemService(Context.NOTIFICATION_SERVICE);
    notificationManager.notify(ID_BIG_NOTIFICATION, notification);
  }
```

```
    //the method will show a small notification
    //parameters are title for message title, message for message text and an intent that will open
    //when you will tap on the notification
    public void showSmallNotification(String title, String message, Intent intent) {
        PendingIntent resultPendingIntent =
                PendingIntent.getActivity(
                        mCtx,
                        ID_SMALL_NOTIFICATION,
                        intent,
                        PendingIntent.FLAG_UPDATE_CURRENT                    );

        NotificationCompat.Builder mBuilder = new NotificationCompat.Builder(mCtx);
        Notification notification;
        notification = mBuilder.setSmallIcon(R.mipmap.ic_launcher).setTicker(title).setWhen(0)
                .setAutoCancel(true)
                .setContentIntent(resultPendingIntent)
                .setContentTitle(title)
                .setSmallIcon(R.mipmap.ic_launcher)
                .setLargeIcon(BitmapFactory.decodeResource(mCtx.getResources(), R.mipmap.ic_launcher))
                .setContentText(message)
                .build();
        notification.flags |= Notification.FLAG_AUTO_CANCEL;
        NotificationManager notificationManager = (NotificationManager)
mCtx.getSystemService(Context.NOTIFICATION_SERVICE);
        notificationManager.notify(ID_SMALL_NOTIFICATION, notification);
    }

    //The method will return Bitmap from an image URL
    private Bitmap getBitmapFromURL(String strURL) {
        try {
            URL url = new URL(strURL);
```

```
        HttpURLConnection connection = (HttpURLConnection) url.openConnection();
        connection.setDoInput(true);
        connection.connect();
        InputStream input = connection.getInputStream();
        Bitmap myBitmap = BitmapFactory.decodeStream(input);
        return myBitmap;
      } catch (IOException e) {
        e.printStackTrace();
        return null;
      }
  }  }
```

MyFirebasseMessagingService.java

```
package net.simplifiedcoding.firebasecloudmessaging;
public class MyFirebaseMessagingService extends FirebaseMessagingService {
   private static final String TAG = "MyFirebaseMsgService";

   @Override
   public void onMessageReceived(RemoteMessage remoteMessage) {
      if (remoteMessage.getData().size() > 0) {
        Log.e(TAG, "Data Payload: " + remoteMessage.getData().toString());
        try {
          JSONObject json = new JSONObject(remoteMessage.getData().toString());
          sendPushNotification(json);
        } catch (Exception e) {
          Log.e(TAG, "Exception: " + e.getMessage());
        }
      }
   }
```

```
//this method will display the notification
//We are passing the JSONObject that is received from
//firebase cloud messaging
private void sendPushNotification(JSONObject json) {
    //optionally we can display the json into log
    Log.e(TAG, "Notification JSON " + json.toString());
    try {
        //getting the json data
        JSONObject data = json.getJSONObject("data");
        //parsing json data
        String title = data.getString("title");
        String message = data.getString("message");
        String imageUrl = data.getString("image");
        //creating MyNotificationManager object
        MyNotificationManager mNotificationManager = new MyNotificationManager(getApplicationContext());

        //creating an intent for the notification
        Intent intent = new Intent(getApplicationContext(), MainActivity.class);
        //if there is no image
        if(imageUrl.equals("null")){
            //displaying small notification
            mNotificationManager.showSmallNotification(title, message, intent);
        }else{
            //if there is an image
            //displaying a big notification
            mNotificationManager.showBigNotification(title, message, imageUrl, intent);
        }
    } catch (JSONException e) {
        Log.e(TAG, "Json Exception: " + e.getMessage());
    } catch (Exception e) {
        Log.e(TAG, "Exception: " + e.getMessage());
```

```
        }
    }
}
```

MyFirebaseInstanceIDService.java

```
package net.simplifiedcoding.firebasecloudmessaging;
public class MyFirebaseInstanceIDService extends FirebaseInstanceIdService {
    private static final String TAG = "MyFirebaseIIDService";
    @Override
    public void onTokenRefresh() {
        String refreshedToken = FirebaseInstanceId.getInstance().getToken();
        Log.d(TAG, "Refreshed token: " + refreshedToken);
        storeToken(refreshedToken);
    }

    private void storeToken(String token) {
        //saving the token on shared preferences
        SharedPrefManager.getInstance(getApplicationContext()).saveDeviceToken(token);
    }
}
```

SharedPrefManager.java

```
package net.simplifiedcoding.firebasecloudmessaging;
public class SharedPrefManager {
    private static final String SHARED_PREF_NAME = "FCMSharedPref";
    private static final String TAG_TOKEN = "tagtoken";
    private static SharedPrefManager mInstance;
```

```
    private static Context mCtx;

    private SharedPrefManager(Context context) {
        mCtx = context;
    }


    public static synchronized SharedPrefManager getInstance(Context context) {
        if (mInstance == null) {
            mInstance = new SharedPrefManager(context);
        }
        return mInstance;
    }




    //this method will save the device token to shared preferences
    public boolean saveDeviceToken(String token){
        SharedPreferences sharedPreferences = mCtx.getSharedPreferences(SHARED_PREF_NAME,
                                        Context.MODE_PRIVATE);
        SharedPreferences.Editor editor = sharedPreferences.edit();
        editor.putString(TAG_TOKEN, token);
        editor.apply();
        return true;
    }


    //this method will fetch the device token from shared preferences
    public String getDeviceToken(){
        SharedPreferences sharedPreferences = mCtx.getSharedPreferences(SHARED_PREF_NAME,
Context.MODE_PRIVATE);
        return  sharedPreferences.getString(TAG_TOKEN, null);
    }
}
```

EndPoints.java

```
package net.simplifiedcoding.firebasecloudmessaging;

public class EndPoints {

    public static final String URL_REGISTER_DEVICE = "http://192.168.1.101/FcmExample/RegisterDevice.php";

    public static final String URL_SEND_SINGLE_PUSH = "http://192.168.1.101/FcmExample/sendSinglePush.php";

    public static final String URL_SEND_MULTIPLE_PUSH =
"http://192.168.1.101/FcmExample/sendMultiplePush.php";

    public static final String URL_FETCH_DEVICES = "http://192.168.1.101/FcmExample/GetRegisteredDevices.php";

}
```

MyVolley.java

```
package net.simplifiedcoding.firebasecloudmessaging;

public class MyVolley {


    private static MyVolley mInstance;

    private RequestQueue mRequestQueue;

    private static Context mCtx;


    private MyVolley(Context context) {

        mCtx = context;

        mRequestQueue = getRequestQueue();

    }


    public static synchronized MyVolley getInstance(Context context) {

        if (mInstance == null) {

            mInstance = new MyVolley(context);

        }

        return mInstance;

    }
```

```
    public RequestQueue getRequestQueue() {
        if (mRequestQueue == null) {
            // getApplicationContext() is key, it keeps you from leaking the
            // Activity or BroadcastReceiver if someone passes one in.
            mRequestQueue = Volley.newRequestQueue(mCtx.getApplicationContext());
        }
        return mRequestQueue;
    }

    public <T> void addToRequestQueue(Request<T> req) {
        getRequestQueue().add(req);
    }
}
```