Sri Lanka Institute of Information Technology

Android Flower Information App Using RESTful Web Services

Android Project Report

SE4010 - CTSE

IT16017184 T.H.S.T.Chandrasena

IT16001244 S.Dharshika

May 2019

Contents

1. Introduction

This document is a detailed report of Android Flower Information application using REST-API.The Flora Bucket Android application helps users to discover flowers information with names, images and descriptions.User able to add new flower information with the name, description and image.Also user can add images from phone image gallery or able to allow and access the phone camera and capture instant pictures of flowers.Furthermore users able to select and update, delete specific flower information.In advanced by clicking ‘help’ option in right menu item, user able to redirect to the help UI and know more about the app guidance. Addition to that, user able to click an message box icon on top right of the UI and redirect to the gmail account link by accessing the internet.

2. Methodology

This Android application using flower-rest-api as REST-API backend build on Express JS, Node JS and MySQL as the database. Following are the other required models for our backend project.

- body-parser
- sequelize
- express-validator/check
- express-validator/filter
- multer
- path
- Crypto

MySQL server hosted in the localhost port 3000.

Android application basically uses two activities to run the flora bucket application. In there we have use

- Adapters - Flower Adapter, image Adapter
- Models to handle API responses and to retrieved data from the database (Flower)
- A separate class to specify the endpoint connecters.

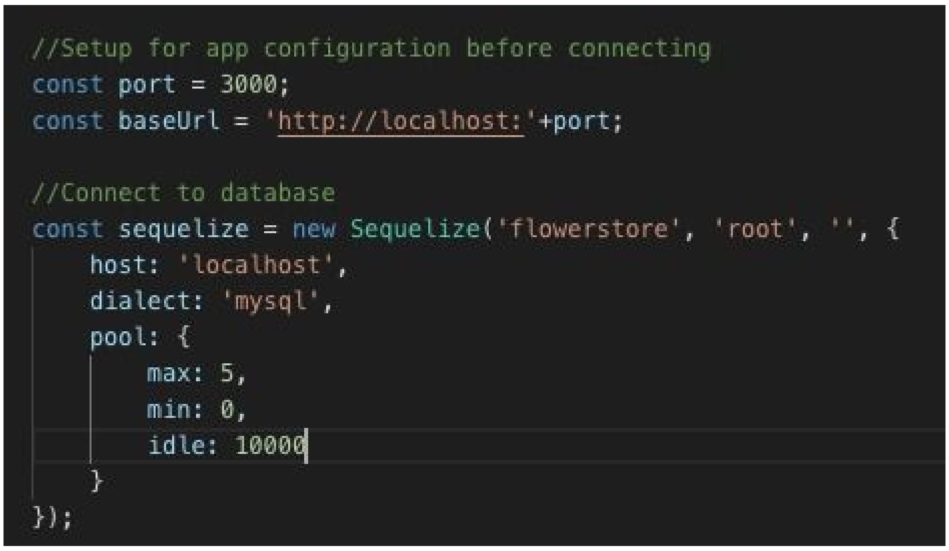
In the process of application development we have used

- Alert dialog boxes to provide pop up notifications
- Toolbars to handle multiple list views and handle item selecting events
- Camera and gallery access
- Internet permission handling
- Explicit and implicit intents
- Menu bars
- Image slide shows

All these android related options have been used to make the Flora bucket app more user friendly and useful.

3. Installation

1. Install Android Studio, NodeJs and MySQL server on local machine.
2. Follow the steps to run **flower-rest-api**(Backend REST-API)
    1. Create new database using MySQL database. Grab **flowerstore.sql** included in **flower-rest-api** folder and import into MySQL server.
    2. 'cd' to directory and run **npm install** using Terminal or Windows PowerShell
    3. Look into lines below in **index.js** file and configure the database and port



```
//Setup for app configuration before connecting
const port = 3000;
const baseUrl = 'http://localhost:'+port;

//Connect to database
const sequelize = new Sequelize('flowerstore', 'root', '', {
    host: 'localhost',
    dialect: 'mysql',
    pool: {
        max: 5,
        min: 0,
        idle: 10000
    }
});
```

Figure 3.1

    4. Run **node index.js** to start the server

Routes Used in API CRUD

GET /flower/ Get all flowers

GET /flower/<fno> Get flower by FNO(flower number)

POST /flower/add Add new flower into collection

POST /flower/:fno/update Update existing flower

POST /flower/<fbn> Delete flower by FNO(flower number)

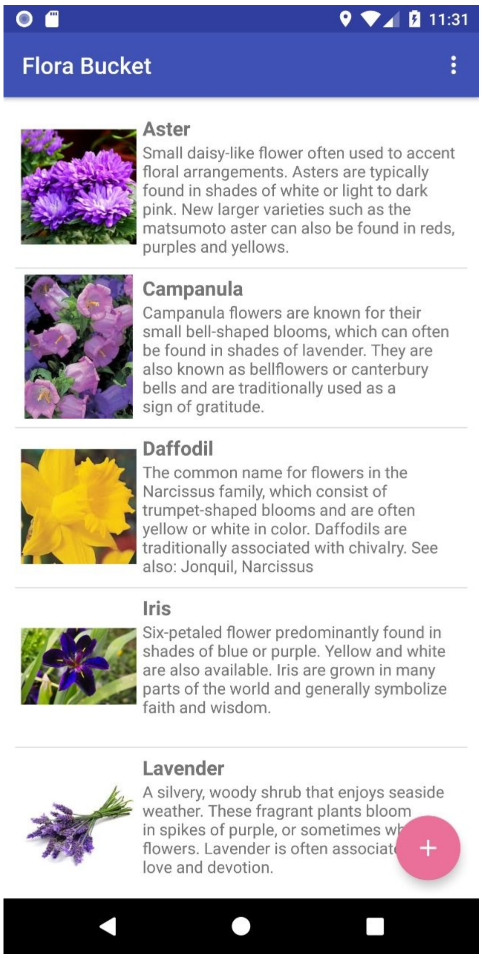
Test flower-rest-api

Can use any REST-API Clients to test endpoints

3. Follow the steps to run **simple-android-crud** App(FrontEnd Android Application)
    1. Open Android Studio and open **simple-android-crud** Android project.
    2. Connect to localhost using Android Emulator, use IP 10.0.2.2:3000 as BASE URL.
    3. Run the application using device or Android Emulator.

4. User Interfaces

When user opens the application, Main UI will be appeared as shown in Figure 4.1.User able to see a list of flowers with information in Main UI. In Figure 4.2 it shows there is an add button with plus icon on the bottom right of the Main UI.When user clicked the add button ,user will be redirected to the Add UI shown in figure 4.3. User can add a flower information in Add UI.By clicking 'ADD FLOWER' button user can include an image by choosing 'Gallery' or 'Camera' option in the Alert Dialog as shown in figure 4.4.User can capture an image of a flower or select an image from gallery. When chosen a picture by clicking 'SAVE' button client can add new flower information to the list.



Aster
Small daisy-like flower often used to accent floral arrangements. Asters are typically found in shades of white or light to dark pink. New larger varieties such as the matsumoto aster can also be found in reds, purples and yellows.

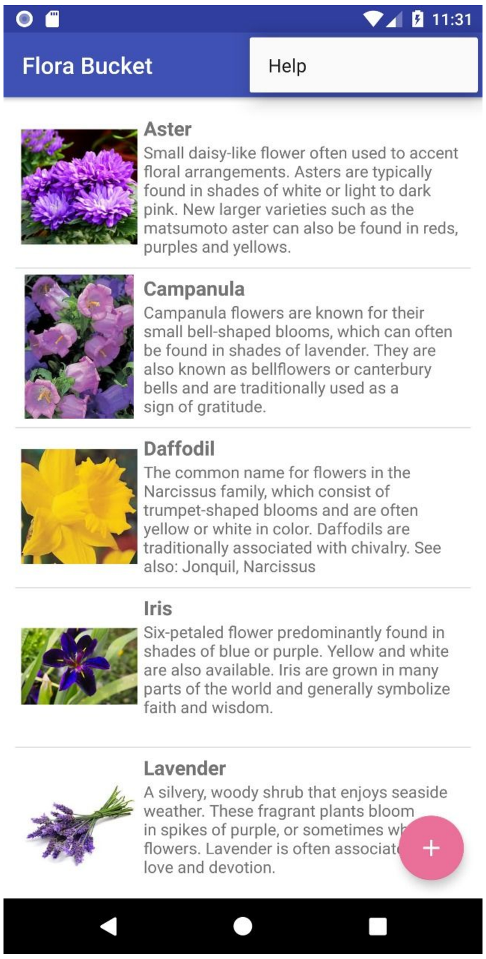
Campanula
Campanula flowers are known for their small bell-shaped blooms, which can often be found in shades of lavender. They are also known as bellflowers or canterbury bells and are traditionally used as a sign of gratitude.

Daffodil
The common name for flowers in the Narcissus family, which consist of trumpet-shaped blooms and are often yellow or white in color. Daffodils are traditionally associated with chivalry. See also: Jonquil, Narcissus

Iris
Six-petaled flower predominantly found in shades of blue or purple. Yellow and white are also available. Iris are grown in many parts of the world and generally symbolize faith and wisdom.

Lavender
A silvery, woody shrub that enjoys seaside weather. These fragrant plants bloom in spikes of purple, or sometimes wh flowers. Lavender is often associat love and devotion.

Figure 4.1 Main UI



Aster
Small daisy-like flower often used to accent floral arrangements. Asters are typically found in shades of white or light to dark pink. New larger varieties such as the matsumoto aster can also be found in reds, purples and yellows.

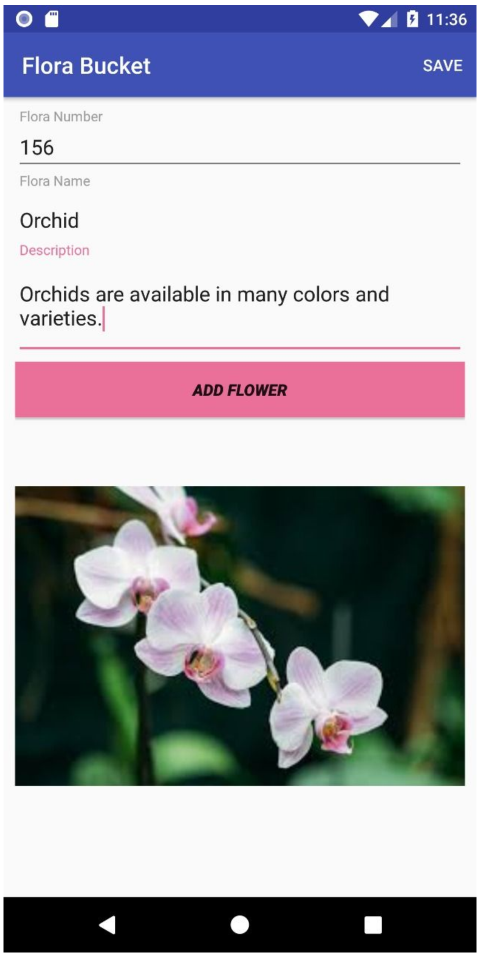
Campanula
Campanula flowers are known for their small bell-shaped blooms, which can often be found in shades of lavender. They are also known as bellflowers or canterbury bells and are traditionally used as a sign of gratitude.

Daffodil
The common name for flowers in the Narcissus family, which consist of trumpet-shaped blooms and are often yellow or white in color. Daffodils are traditionally associated with chivalry. See also: Jonquil, Narcissus

Iris
Six-petaled flower predominantly found in shades of blue or purple. Yellow and white are also available. Iris are grown in many parts of the world and generally symbolize faith and wisdom.

Lavender
A silvery, woody shrub that enjoys seaside weather. These fragrant plants bloom in spikes of purple, or sometimes wh flowers. Lavender is often associat love and devotion.

Figure 4.2 Main UI



Flora Number
156
Flora Name
Orchid
Description
Orchids are available in many colors and varieties.

Figure 4.3 Add UI

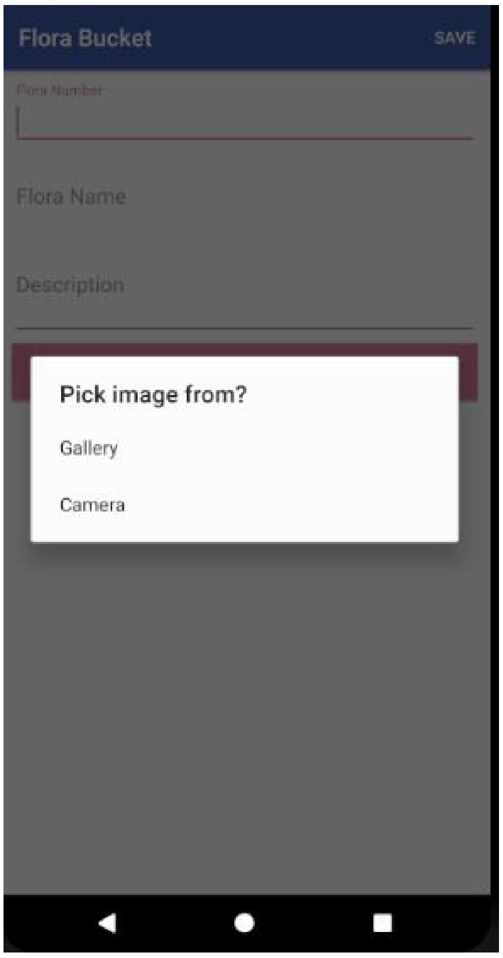
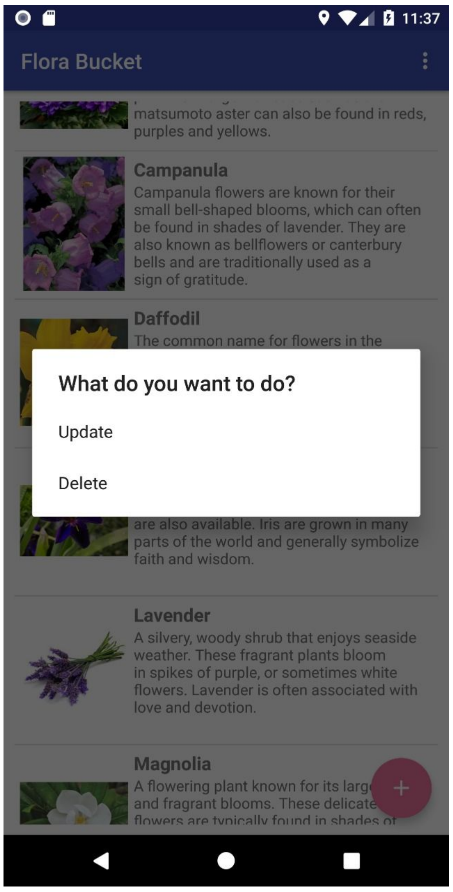


Figure 4.4 Image Alert Dialog Box

Figure 4.5 shows that user able to choose a particular flower from the list and select 'Update' or 'Delete' option from Alert Dialog box.By selecting ‘Update’ option user able to update the selected specific flower information and ‘Delete’ option to delete selected flower from the list.By clicking ‘Help’ option from the top side menu button in main UI user will be redirected to the Help UI shown in figure 4.6.In Help UI user able to see more images by swiping the image slide.User can click message icon button on the top right of the Help UI to redirect to the email link using internet.Figure 4.7 shows redirected email page.


11:37

Flora Bucket

matsumoto aster can also be found in reds, purples and yellows.

Campanula
Campanula flowers are known for their small bell-shaped blooms, which can often be found in shades of lavender. They are also known as bellflowers or canterbury bells and are traditionally used as a sign of gratitude.

Daffodil
The common name for flowers in the

What do you want to do?

Update

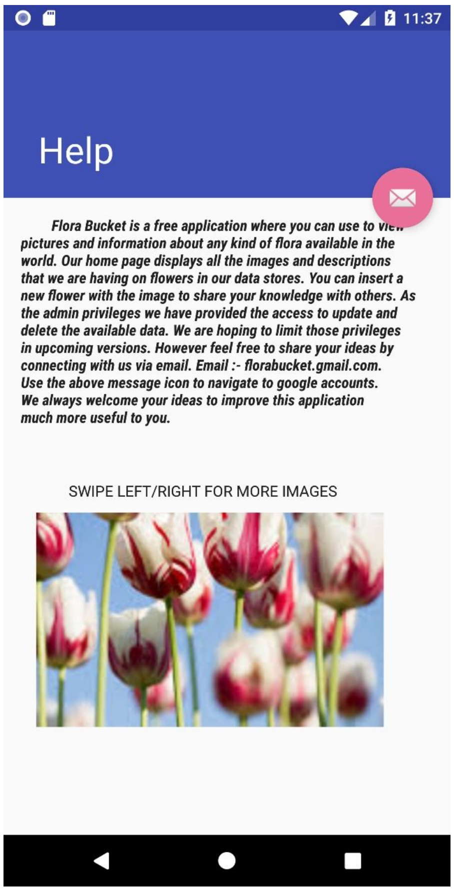
Delete

are also available. Iris are grown in many parts of the world and generally symbolize faith and wisdom.

Lavender
A silvery, woody shrub that enjoys seaside weather. These fragrant plants bloom in spikes of purple, or sometimes white flowers. Lavender is often associated with love and devotion.

Magnolia
A flowering plant known for its large and fragrant blooms. These delicate flowers are typically found in shades of

Figure 4.5 Alert Dialog Box

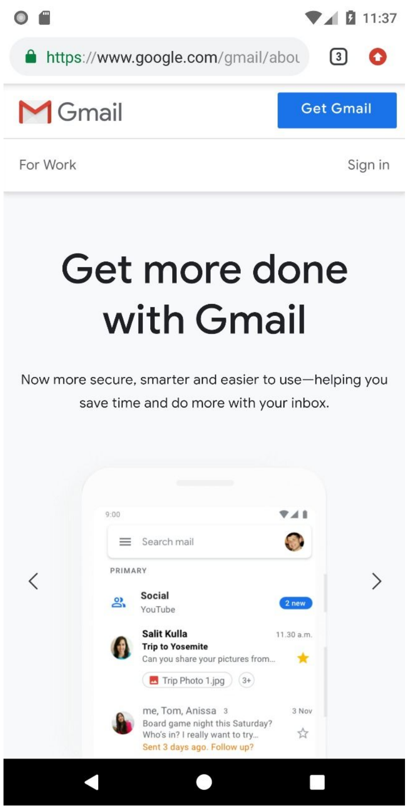

11:37

Help

Flora Bucket is a free application where you can use to view pictures and information about any kind of flora available in the world. Our home page displays all the images and descriptions that we are having on flowers in our data stores. You can insert a new flower with the image to share your knowledge with others. As the admin privileges we have provided the access to update and delete the available data. We are hoping to limit those privileges in upcoming versions. However feel free to share your ideas by connecting with us via email. Email :- florabucket.gmail.com. Use the above message icon to navigate to google accounts. We always welcome your ideas to improve this application much more useful to you.

SWIPE LEFT/RIGHT FOR MORE IMAGES

Figure 4.6 Help UI


11:37

https://www.google.com/gmail/abou

Gmail

Get Gmail

For Work

Sign in

Get more done with Gmail

Now more secure, smarter and easier to use—helping you save time and do more with your inbox.

Search mail
PRIMARY
Social
YouTube
2 new
Salit Kulla
Trip to Yosemite
Can you share your pictures from...
11:30 a.m.
Trip Photo 1.jpg
me, Tom, Anissa 3
Board game night this Saturday?
Who's in? I really want to try...
Sent 3 days ago. Follow up?
3 Nov

Figure 4.7 Gmail

4.References

[1].[online]. https://github.com/budasuyasa/simple-android-crud

[2].[online].
https://camposha.info/course/android-fragment/lesson/android-data-passing-fragment-to-activity-via-intent/

[3][online]. https://github.com/budasuyasa/express-rest-api

5.Appendix

back End

```
//import required module
const express = require('express');
const app = express();
const bodyParser = require('body-parser'); //post body handler
const Sequelize = require('sequelize'); //Database ORM
const { check, validationResult } =
require('express-validator/check'); //form validation
const { matchedData, sanitize } =
require('express-validator/filter'); //sanitize form params
const multer  = require('multer'); //multipar form-data
const path = require('path');
const crypto = require('crypto');

//Set body parser for HTTP post operation
app.use(bodyParser.json()); // support json encoded bodies
app.use(bodyParser.urlencoded({ extended: true })); // support
encoded bodies

//set static assets to public directory
app.use(express.static('public'));
const uploadDir = '/img/';
const storage = multer.diskStorage({
   destination: "./public"+uploadDir,
   filename: function (req, file, cb) {
     crypto.pseudoRandomBytes(16, function (err, raw) {
       if (err) return cb(err)

       cb(null, raw.toString('hex') +
path.extname(file.originalname))
     })
   }
})

const upload = multer({storage: storage, dest: uploadDir });

//Setup for app configuration before connecting
const port = 3000;
const baseUrl = 'http://localhost:'+port;

//Connect to database
const sequelize = new Sequelize('flowerstore', 'root', '', {
   host: 'localhost',
   dialect: 'mysql',
   pool: {
       max: 5,
       min: 0,
       idle: 10000
   }
});

//Define model
const flower = sequelize.define('flower', {
   'id': {
       type: Sequelize.INTEGER,
       primaryKey: true,
       autoIncrement: true
   },
   'fno': Sequelize.STRING,
   'name': Sequelize.STRING,
   'description': Sequelize.TEXT,
   'image': {
       type: Sequelize.STRING,
       //Set custom getter for flower image using URL
       get(){
           const image = this.getDataValue('image');
           return uploadDir+image;
       }
   },
   'createdAt': {
       type: Sequelize.DATE,
       defaultValue: Sequelize.NOW
   },
   'updatedAt': {
       type: Sequelize.DATE,
       defaultValue: Sequelize.NOW
   },



}, {
   //prevent sequelize transform table name into plural
   freezeTableName: true,
});

//get all flowers
app.get('/flower/', (req, res) => {
   flower.findAll().then(flower => {
       res.json(flower)
   })
})

//get flower by fno
app.get('/flower/:fno', (req, res) => {
   flower.findOne({where: {fno: req.params.fno}}).then(flower
=> {
       res.json(flower)
   })
})

//Insert operation
app.post('/flower/add', [
   //File upload
   upload.single('image'),

   //Set form validation rule
   check('fno')
       .isLength({ min: 2 })
       .isNumeric()
       .custom(value => {
           return flower.findOne({where: {fno: value}}).then(b
=> {
               if(b){
                   throw new Error('Flower Number already in
use');
               }
           })
       }
   ),
   check('name')
       .isLength({min: 2}),
   check('description')
    .isLength({min: 10})

],(req, res) => {
   const errors = validationResult(req);
   if (!errors.isEmpty()) {
       return res.status(200).json({ status: 'error',
message:"Form error", data:null, errors: errors.mapped() });
   }

   flower.create({
       name: req.body.name,
       fno:req.body.fno,
       description: req.body.description,
       image: req.file === undefined ? "" : req.file.filename
   }).then(newFlower => {
       res.json({
           "status":"success",
           "message":"Flower added",
           "data": newFlower
       })
   })
})



//Update operation
app.post('/flower/:fno/update', [
   //File upload
   upload.single('image'),
   //Set form validation rule
   check('fno')
       .isLength({ min: 2 })
       .isNumeric()
       .custom(value => {
           return flower.findOne({where: {fno: value}}).then(b
=> {
               if(!b){
                   throw new Error('Flower Number not found');
               }
           })
       }
   ),
   check('name')
       .isLength({min: 2}),
   check('description')
    .isLength({min: 10})

],(req, res) => {
   const errors = validationResult(req);
   if (!errors.isEmpty()) {
       return res.status(200).json({ status: 'error',
message:"Form error", data:null, errors: errors.mapped() });
   }

   let prevImage = null;
   flower.findOne({where: {fno: req.body.fno}}).then(b => {

       let newThumbnail = '';
       console.log("Req.file", req.file);
```

```
        if(req.file === undefined || req.file.filename ===""){
            let parts = b.image.split('/');
            newThumbnail = parts[parts.length - 1];
        }else{
            newThumbnail = req.file.filename;
        }

        console.log("newThumbnail", newThumbnail);

        const update = {
            name: req.body.name,
            fno: req.body.fno,
            description: req.body.description,
            image: newThumbnail
        }

        flower.update(update,{where: {fno: req.body.fno}})
            .then(affectedRow => {
                return flower.findOne({where: {fno:
req.body.fno}})
            })
            .then(b => {
                res.json({
                    "status": "success",
                    "message": "Flower updated",
                    "data": b
                })
            })
        })
})
```

```
app.post('/flower/:fno/delete',[
   //Set form validation rule
   check('fno')
       .isNumeric()
       .custom(value => {
           return flower.findOne({where: {fno: value}}).then(b
=> {
               if(!b){
                   throw new Error('Flower Number not found');
               }
           })
       }
   ),
], (req, res) => {
   flower.destroy({where: {fno: req.params.fno}})
       .then(affectedRow => {
           if(affectedRow){
               return {
                   "status":"success",
                   "message": "Flower deleted",
                   "data": null
               }
           }

           return {
               "status":"error",
               "message": "Failed",
               "data": null
           }

       })
```

```
        .then(r => {
            res.json(r)
        })
})

app.listen(port, () => console.log("flower-rest-api run on
"+baseUrl ))
```

Front End

01.FlowerAdapter.java

```
package budasuyasa.android.simplecrud.Adapter;

import android.content.Context;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;
import com.squareup.picasso.Picasso;

import java.util.List;

import budasuyasa.android.simplecrud.Config.ApiEndpoint;
import budasuyasa.android.simplecrud.Models.Flower;
import budasuyasa.android.simplecrud.R;


public class FlowerAdapter extends RecyclerView.Adapter<FlowerAdapter.ViewHolder> {
  /**
   * Create ViewHolder class to bind list item view
   */
  public class ViewHolder extends RecyclerView.ViewHolder{

      public TextView title;
      public TextView description;
      public ImageView thumbnail;

      public ViewHolder(View itemView) {
          super(itemView);

          title = (TextView) itemView.findViewById(R.id.tvTitle);
          description = (TextView) itemView.findViewById(R.id.tvDescription);
          thumbnail = (ImageView) itemView.findViewById(R.id.imageView);
      }
  }

  private List<Flower> mListData;
  private Context mContext;

  public FlowerAdapter(Context context, List<Flower> listData){
      mListData = listData;
      mContext = context;
  }

  private Context getmContext(){return mContext;}

  @Override
  public ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
      Context context = parent.getContext();
      LayoutInflater inflater = LayoutInflater.from(context);
```

```java
        // Inflate the custom layout
        View view = inflater.inflate(R.layout.flower_list_item, parent, false);

        // Return a new holder instance
        ViewHolder viewHolder = new ViewHolder(view);
        return viewHolder;
    }

    @Override
    public void onBindViewHolder(ViewHolder holder, int position) {
        Flower m = mListData.get(position);
        holder.title.setText(m.getName());
        holder.description.setText(m.getDescription());
        Picasso.get().load(ApiEndpoint.BASE + m.getImage()).into(holder.thumbnail);
    }

    @Override
    public int getItemCount() {
        return mListData.size();
    }


}
```

02. RecycleItemClickListner.java

```java
package budasuyasa.android.simplecrud.Adapter;

import android.content.Context;
import android.support.v7.widget.RecyclerView;
import android.view.GestureDetector;
import android.view.MotionEvent;
import android.view.View;

/**
* Class for handling click events on RecycleView.
*/
public class RecyclerItemClickListener implements RecyclerView.OnItemTouchListener {
  public interface OnItemClickListener {
      void onItemClick(View view, int position);

      void onItemLongClick(View view, int position);
  }

  private OnItemClickListener mListener;
  private GestureDetector mGestureDetector;

  public RecyclerItemClickListener(Context context, final RecyclerView recyclerView,
OnItemClickListener listener) {
      mListener = listener;

      mGestureDetector = new GestureDetector(context, new
GestureDetector.SimpleOnGestureListener() {
          @Override
          public boolean onSingleTapUp(MotionEvent e) {
              return true;
          }

          @Override
          public void onLongPress(MotionEvent e) {
              View childView = recyclerView.findChildViewUnder(e.getX(), e.getY());
```

```
                if (childView != null && mListener != null) {
                    mListener.onItemLongClick(childView,
recyclerView.getChildAdapterPosition(childView));
                }
            }
        });
    }

    @Override
    public boolean onInterceptTouchEvent(RecyclerView view, MotionEvent e) {
        View childView = view.findChildViewUnder(e.getX(), e.getY());

        if (childView != null && mListener != null && mGestureDetector.onTouchEvent(e)) {
            mListener.onItemClick(childView, view.getChildAdapterPosition(childView));
        }

        return false;
    }

    @Override
    public void onTouchEvent(RecyclerView view, MotionEvent motionEvent) {
    }

    @Override
    public void onRequestDisallowInterceptTouchEvent(boolean disallowIntercept) {
    }
}
```

03. ApiEndpoint.java

```
package budasuyasa.android.simplecrud.Config;
```

```
public class ApiEndpoint {
  public static String BASE = "http://10.0.2.2:3000/";
  public static String FLOWERS = BASE + "flower";
  public static String ADD_FLOWER = BASE + "flower/add";
}
```

04. APIResponse.java

```
package budasuyasa.android.simplecrud.Models;

/**
* Class for mapping API responses
*/
public class APIResponse {
  String status;
  String message;

  public String getStatus() {
      return status;
  }
  public void setStatus(String status) {
      this.status = status;
  }
  public String getMessage() {
      return message;
  }

  public void setMessage(String message) {
      this.message = message;
   }
}
```

05. Flower.java

```
package budasuyasa.android.simplecrud.Models;
import android.os.Parcel;
import android.os.Parcelable;

/**
* Class to mapping book object implement Parcelable
* because it will be used for the Flower passing object between activities
*/

public class Flower implements Parcelable{

   String id;
   String name;
   String fno;
   String image;
   String description;
   public String getId() {
      return id;
   }
   public void setId(String id) {
      this.id = id;
   }
   public String getName() {
      return name;
   }
```

```
  public void setName(String name) {
      this.name = name;
  }
  public String getFno() {
      return fno;
  }
  public void setFno(String isbn) {
      this.fno = fno;
  }
  public String getImage() {
      return image;
  }
  public void setImage(String image) {
      this.image = image;
  }
  public String getDescription() {
      return description;
  }
  public void setDescription(String description) {
      this.description = description;
  }
  protected Flower(Parcel in) {
      id = in.readString();
      name = in.readString();
      fno = in.readString();

      image = in.readString();
      description = in.readString();
  }
  @Override
  public int describeContents() {
      return 0;
```

```
   }
   @Override
   public void writeToParcel(Parcel dest, int flags) {
       dest.writeString(id);
       dest.writeString(name);
       dest.writeString(fno);
       dest.writeString(image);
       dest.writeString(description);
   }
   @SuppressWarnings("unused")
   public static final Parcelable.Creator<Flower> CREATOR = new
Parcelable.Creator<Flower>() {
       @Override
       public Flower createFromParcel(Parcel in) {
           return new Flower(in);
       }
       @Override
       public Flower[] newArray(int size) {
           R eturn new Flower[size];
       }
   };
}
```

06. AddFlower.java

```
package budasuyasa.android.simplecrud;

import android.Manifest;
import android.content.DialogInterface;
import android.content.Intent;
import android.support.design.widget.TextInputEditText;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Toast;

import com.facebook.stetho.okhttp3.StethoInterceptor;
import com.google.gson.Gson;
import com.google.gson.JsonSyntaxException;
import com.karumi.dexter.Dexter;
import com.karumi.dexter.MultiplePermissionsReport;
import com.karumi.dexter.PermissionToken;
import com.karumi.dexter.listener.PermissionRequest;
import com.karumi.dexter.listener.multi.MultiplePermissionsListener;
import com.kbeanie.multipicker.api.CameraImagePicker;
import com.kbeanie.multipicker.api.ImagePicker;
import com.kbeanie.multipicker.api.Picker;
import com.kbeanie.multipicker.api.callbacks.ImagePickerCallback;
import com.kbeanie.multipicker.api.entity.ChosenImage;
import com.squareup.picasso.Picasso;

import org.apache.commons.lang3.StringUtils;

import java.io.File;
import java.io.IOException;
import java.util.List;
import budasuyasa.android.simplecrud.Config.ApiEndpoint;
import budasuyasa.android.simplecrud.Models.APIResponse;
import budasuyasa.android.simplecrud.Models.Flower;
import butterknife.BindView;
import butterknife.ButterKnife;
import okhttp3.Call;
import okhttp3.Callback;
import okhttp3.MediaType;
import okhttp3.MultipartBody;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.RequestBody;
import okhttp3.Response;

/**
* Activity untuk menambahkan dan mengupdate data buku
*/
public class AddFlower extends AppCompatActivity {



  String imagePath;
  String imageFileName;

  private static int EDIT_MODE = 0;
  private static int ADD_MODE = 1;
  int MODE = 1;

  Flower editFlower;



  ImagePicker imagePicker;
  CameraImagePicker cameraImagePicker;

  String TAG = getClass().getName().toString();
  private static final MediaType MEDIA_TYPE_PNG = MediaType.parse("image/jpeg");

  @BindView(R.id.button) Button btnAddFlower;
  @BindView(R.id.imageView2) ImageView imageView;
  @BindView(R.id.etFNO)
  TextInputEditText etFno;
  @BindView(R.id.etName)
  TextInputEditText etName;

  @BindView(R.id.etDescription)
  TextInputEditText etDescription;

  OkHttpClient client = new OkHttpClient.Builder()
          .addNetworkInterceptor(new StethoInterceptor())
          .build();

  Gson gson = new Gson(); //handling json object formt

  //Image picker callback, via gallery/camera
  ImagePickerCallback callback = new ImagePickerCallback(){
      @Override
      public void onImagesChosen(List<ChosenImage> images) {
          // get image path
          if(images.size() > 0){
              imagePath = images.get(0).getOriginalPath();
              imageFileName = images.get(0).getDisplayName();
              Picasso.get().load(new File(imagePath)).into(imageView);
          }
      }
```

```
        @Override
        public void onError(String message) {
            // Do error handling
        }
    };

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_add_flower);
        ButterKnife.bind(this);

        //camera ImagePicker
        imagePicker = new ImagePicker(AddFlower.this);
        cameraImagePicker = new CameraImagePicker(AddFlower.this);

        //set callback handler
        imagePicker.setImagePickerCallback(callback);
        cameraImagePicker.setImagePickerCallback(callback);



        btnAddFlower.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                pickImageChoice();
            }
        });



        if(getIntent().getParcelableExtra("flower") != null){
            editFlower = (Flower) getIntent().getParcelableExtra("flower");
            MODE = EDIT_MODE;
            etFno.setText(editFlower.getFno());
            etName.setText(editFlower.getName());
            etDescription.setText(editFlower.getDescription());
            Picasso.get().load(ApiEndpoint.BASE + editFlower.getImage()).into(imageView);
            Log.d(TAG, "onCreate: "+ApiEndpoint.BASE + editFlower.getImage());

        }else{
            MODE = ADD_MODE;

            //Set default isi form
            etFno.setText("");
            etName.setText("");
            etDescription.setText("");
        }
    }

    /**
     * Method untuk mengambil gambar ketika tombol Thumbnail di click
     */
    private void pickImageChoice(){

        Dexter.withActivity(this)
            .withPermissions(Manifest.permission.CAMERA,
Manifest.permission.WRITE_EXTERNAL_STORAGE)
            .withListener(new MultiplePermissionsListener() {
                @Override
                public void onPermissionsChecked(MultiplePermissionsReport report) {

                    // setup the alert builder
                    AlertDialog.Builder builder = new AlertDialog.Builder(AddFlower.this);
                    builder.setTitle("Pick image from?");
                    // add a list
                    String[] menus = {"Gallery", "Camera"};
                    builder.setItems(menus, new DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface dialog, int which) {
                            switch (which) {
                                case 0: // dari gallery
                                    imagePicker.pickImage();
                                    break;
                                case 1: // dari camera
                                    imagePath = cameraImagePicker.pickImage();
                                    break;
                            }
                        }
                    });
                    // create and show the alert dialog
                    AlertDialog dialog = builder.create();
                    dialog.show();
                }

                @Override
                public void onPermissionRationaleShouldBeShown(List<PermissionRequest>
permissions, PermissionToken token) {

                }
            }).check();
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        if(resultCode == RESULT_OK) {
```

```
        if(requestCode == Picker.PICK_IMAGE_DEVICE) {
            if(imagePicker == null) {
                imagePicker = new ImagePicker(AddFlower.this);
                imagePicker.setImagePickerCallback(callback);
            }
            imagePicker.submit(data);
        }
        if(requestCode == Picker.PICK_IMAGE_CAMERA) {
            if(cameraImagePicker == null) {
                cameraImagePicker = new CameraImagePicker(AddFlower.this);
                cameraImagePicker.reinitialize(imagePath);
                // OR in one statement
                // imagePicker = new CameraImagePicker(Activity.this, outputPath);
                cameraImagePicker.setImagePickerCallback(callback);
            }
            cameraImagePicker.submit(data);
        }
    }
}

@Override
protected void onSaveInstanceState(Bundle outState) {
    // You have to save path in case your activity is killed.
    // In such a scenario, you will need to re-initialize the CameraImagePicker
    outState.putString("picker_path", imagePath);
    super.onSaveInstanceState(outState);
}

@Override
protected void onRestoreInstanceState(Bundle savedInstanceState) {
    // After Activity recreate, you need to re-initialize these
    // two values to be able to re-initialize CameraImagePicker
```

```
        if (savedInstanceState != null) {
            if (savedInstanceState.containsKey("picker_path")) {
                imagePath = savedInstanceState.getString("picker_path");
            }
        }
        super.onRestoreInstanceState(savedInstanceState);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.menu_add, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle item selection
        switch (item.getItemId()) {
            case R.id.save:
                saveFlower();
                return true;
            default:
                return super.onOptionsItemSelected(item);
        }
    }

    private void saveFlower(){
        //Get  edit text, assign to variabel
        String fno = etFno.getText().toString();
        String name = etName.getText().toString();
        String description = etDescription.getText().toString();
    if(StringUtils.isEmpty(fno)) return;
    if(StringUtils.isEmpty(name)) return;
    if(StringUtils.isEmpty(description)) return;

    RequestBody requestBody = null;
    String URL = "";

    if(MODE == ADD_MODE){

        if(StringUtils.isEmpty(imagePath)) return;

        requestBody = new MultipartBody.Builder()
                .setType(MultipartBody.FORM)
                .addFormDataPart("fno", fno)
                .addFormDataPart("name", name)
                .addFormDataPart("description", description)
                .addFormDataPart("image", imageFileName,
                        RequestBody.create(MEDIA_TYPE_PNG, new File(imagePath)))
                .build();
        URL = ApiEndpoint.ADD_FLOWER;

    }else if(MODE == EDIT_MODE) {
        if(StringUtils.isBlank(imageFileName)){
            requestBody = new MultipartBody.Builder()
                    .setType(MultipartBody.FORM)
                    .addFormDataPart("fno", fno)
                    .addFormDataPart("name", name)
                    .addFormDataPart("description", description)
                    .build();
        }else{
            requestBody = new MultipartBody.Builder()
```

```
                    .setType(MultipartBody.FORM)
                    .addFormDataPart("fno", fno)
                    .addFormDataPart("name", name)
                    .addFormDataPart("description", description)
                    .addFormDataPart("image", imageFileName,
                            RequestBody.create(MEDIA_TYPE_PNG, new File(imagePath)))
                    .build();
        }
        URL = ApiEndpoint.FLOWERS+"/"+fno+"/update";
    }

    Request request = new Request.Builder()
            .url(URL)
            .post(requestBody)
            .build();

    client.newCall(request).enqueue(new Callback() {
        @Override
        public void onFailure(Call call, final IOException e) {
            AddFlower.this.runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    Log.d("Main Activity", e.getMessage());
                    Toast.makeText(AddFlower.this, e.getMessage(),
Toast.LENGTH_SHORT).show();
                }
            });
        }

        @Override
        public void onResponse(Call call, final Response response) throws IOException {
            if (response.isSuccessful()) {
                try {
                    //Finish activity
                    AddFlower.this.runOnUiThread(new Runnable() {
                        @Override
                        public void run() {
                            APIResponse res =  gson.fromJson(response.body().charStream(),
APIResponse.class);
                            //response success, finish activity
                            if(StringUtils.equals(res.getStatus(), "success")){
                                Toast.makeText(AddFlower.this, "Flower saved!",
Toast.LENGTH_SHORT).show();
                                finish();
                            }else{

                                Toast.makeText(AddFlower.this, "Error: "+res.getMessage(),
Toast.LENGTH_SHORT).show();
                            }
                        }
                    });
                } catch (JsonSyntaxException e) {
                    Log.e("MainActivity", "JSON Errors:"+e.getMessage());
                } finally {
                    response.body().close();
                }

            } else {
                AddFlower.this.runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        Toast.makeText(AddFlower.this, "Server error",
Toast.LENGTH_SHORT).show();
                    }
                });
            }
        }
    });
  }
}
```

07. CrudApplication.java

```
package budasuyasa.android.simplecrud;

import android.app.Application;

import com.facebook.stetho.Stetho;

/**
* Main application file
*/
public class CrudApplication extends Application {
  public void onCreate() {
      super.onCreate();
      Stetho.initializeWithDefaults(this);
  }
}
```

08. HelpActivity.java

```
package budasuyasa.android.simplecrud;

import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.support.design.widget.FloatingActionButton;
import android.support.design.widget.Snackbar;
import android.support.v4.view.ViewPager;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.View;

public class HelpActivity extends AppCompatActivity {

  @Override
  protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_help);
      Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
      setSupportActionBar(toolbar);

      FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab);
      fab.setOnClickListener(new View.OnClickListener() {
          @Override
          public void onClick(View view) {
            /* Snackbar.make(view, "Replace with your own action",
Snackbar.LENGTH_LONG)
                    .setAction("Action", null).show();*/
             Intent browerIntent = new
Intent(Intent.ACTION_VIEW,Uri.parse("https://www.google.com/gmail/"));
             startActivity(browerIntent);
          }
      });

      ViewPager viewPager = findViewById(R.id.view_pager);
      ImageAdapter adapter = new ImageAdapter(this);
      viewPager.setAdapter(adapter);
  }
}
```

09. ImageAdapter.java

```
package budasuyasa.android.simplecrud;

import android.content.Context;
import android.support.v4.view.PagerAdapter;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;

public class ImageAdapter extends PagerAdapter {
  private Context mContext;
  private int[] mImageIds = new
int[]{R.drawable.unknown,R.drawable.images2,R.drawable.images3};

  ImageAdapter (Context context)
  {
      mContext = context;
  }
  @Override
  public int getCount() {
      return mImageIds.length;
  }

  @Override
  public boolean isViewFromObject(View view, Object object) {
      return view == object;
  }

  @Override
  public Object instantiateItem(ViewGroup container, int position) {
      ImageView imageView= new ImageView(mContext);
      imageView.setScaleType(ImageView.ScaleType.CENTER_CROP);
      imageView.setImageResource(mImageIds[position]);
      container.addView(imageView,0);
      return imageView;
  }

  @Override
  public void destroyItem(ViewGroup container, int position, Object object) {
     container.removeView((ImageView)object);
  }
}
```

10. MainActivity.java

```
package budasuyasa.android.simplecrud;

import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.support.design.widget.FloatingActionButton;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.DividerItemDecoration;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.support.v7.widget.Toolbar;
```

```java
import android.util.Log;
import android.view.View;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Toast;

import com.facebook.stetho.okhttp3.StethoInterceptor;
import com.google.gson.Gson;
import com.google.gson.JsonSyntaxException;
import com.google.gson.reflect.TypeToken;

import org.apache.commons.lang3.StringUtils;

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

import budasuyasa.android.simplecrud.Adapter.FlowerAdapter;
import budasuyasa.android.simplecrud.Adapter.RecyclerItemClickListener;
import budasuyasa.android.simplecrud.Config.ApiEndpoint;
import budasuyasa.android.simplecrud.Models.APIResponse;
import budasuyasa.android.simplecrud.Models.Flower;
import okhttp3.Call;
import okhttp3.Callback;
import okhttp3.FormBody;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.RequestBody;
import okhttp3.Response;

import static android.support.v7.widget.RecyclerView.VERTICAL;
```

```
public class MainActivity extends AppCompatActivity {

  RecyclerView recyclerView;
  FlowerAdapter recycleAdapter;
  OkHttpClient client = new OkHttpClient.Builder()
        .addNetworkInterceptor(new StethoInterceptor())
        .build();
  private List<Flower> flowerList = new ArrayList<Flower>();
  Gson gson = new Gson();

  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main);
     Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
     setSupportActionBar(toolbar);

     //Set floating button action (add new flower)
     FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab);
     fab.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
           Intent i = new Intent(MainActivity.this, AddFlower.class);
           startActivity(i);
        }
     });

     //Prepare RecycleView adapter
     recyclerView= (RecyclerView) findViewById(R.id.listView);
     DividerItemDecoration decoration = new
DividerItemDecoration(getApplicationContext(), VERTICAL);
     recyclerView.addItemDecoration(decoration);
```

```
    recycleAdapter = new FlowerAdapter(this, flowerList);
    recyclerView.setAdapter(recycleAdapter);
    recyclerView.setLayoutManager(new LinearLayoutManager(this));

    recyclerView.addOnItemTouchListener(new
RecyclerItemClickListener(MainActivity.this, recyclerView, new
RecyclerItemClickListener.OnItemClickListener() {
        @Override
        public void onItemClick(View view, int position) {
            final Flower flower = flowerList.get(position);
            Log.d("MainActivity", "onItemClick: ");

            // setup the alert builder
            AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);
            builder.setTitle("What do you want to do?");

            // add a list
            String[] menus = {"Update", "Delete"};
            builder.setItems(menus, new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    switch (which) {
                        case 0: // dari gallery
                            update(flower);
                            break;
                        case 1: // dari camera
                            delete(flower.getFno());
                            break;

                    }
                }
            });
```

```
                // create and show the alert dialog
                AlertDialog dialog = builder.create();
                dialog.show();
            }

            @Override
            public void onItemLongClick(View view, int position) {

            }
        }));

        getFlowers();
    }

    private void update(Flower flower){
        Intent intent = new Intent(this, AddFlower.class);
        intent.putExtra("flower", flower);
        startActivity(intent);
    }

    private void delete(String fno) {
        // request body mulipart
        RequestBody formBody = new FormBody.Builder()
                .add("fno", fno)
                .build();

        Request request = new Request.Builder()
                .url(ApiEndpoint.FLOWERS+"/"+fno+"/delete")
                .post(formBody)
                .build();

        //Handle response request
        client.newCall(request).enqueue(new Callback() {
            @Override
            public void onFailure(Call call, final IOException e) {
                MainActivity.this.runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        Log.d("Main Activity", e.getMessage());
                        Toast.makeText(MainActivity.this, e.getMessage(),
    Toast.LENGTH_SHORT).show();
                    }
                });
            }

            @Override
            public void onResponse(Call call, final Response response) throws IOException {
                if (response.isSuccessful()) {
                    try {
                        //Finish activity
                        MainActivity.this.runOnUiThread(new Runnable() {
                            @Override
                            public void run() {
                                APIResponse res =  gson.fromJson(response.body().charStream(),
    APIResponse.class);
                                if(StringUtils.equals(res.getStatus(), "success")){
                                    //Refresh flowers list
                                    getFlowers();
                                }
                            }
                        });
                    } catch (JsonSyntaxException e) {
                        Log.e("MainActivity", "JSON Errors:"+e.getMessage());
                    } finally {
                       // response.body().close();
                        Toast.makeText(MainActivity.this,"Successfully
    Deleted!",Toast.LENGTH_LONG).show();
                    }

                } else {
                    MainActivity.this.runOnUiThread(new Runnable() {
                        @Override
                        public void run() {
                            Toast.makeText(MainActivity.this, "Server error",
    Toast.LENGTH_SHORT).show();
                        }
                    });
                }
            }
        });
     }

     /**
      * Get Flower from REST-API and populate into RecycleView
      */
     private void getFlowers(){
        Request request = new Request.Builder()
                .url(ApiEndpoint.FLOWERS)
                .build();

        //Handle response request
        client.newCall(request).enqueue(new Callback() {
            @Override
            public void onFailure(Call call, final IOException e) {
                MainActivity.this.runOnUiThread(new Runnable() {
                    @Override
```

```
                public void run() {
                    Log.d("Main Activity", e.getMessage());
                    Toast.makeText(MainActivity.this, e.getMessage(),
Toast.LENGTH_SHORT).show();
                }
            });
        }

        @Override
        public void onResponse(Call call, Response response) throws IOException {
            if (response.isSuccessful()) {
                try {
                    final ArrayList<Flower> res = gson.fromJson(response.body().string(), new
TypeToken<ArrayList<Flower>>(){}.getType());
                    MainActivity.this.runOnUiThread(new Runnable() {
                        @Override
                        public void run() {
                            flowerList.clear();
                            flowerList.addAll(res);
                            recycleAdapter.notifyDataSetChanged();
                        }
                    });
                } catch (JsonSyntaxException e) {
                    Log.e("MainActivity", "JSON Errors:"+e.getMessage());
                } finally {
                    response.body().close();
                }

            } else {
                MainActivity.this.runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        Toast.makeText(MainActivity.this, "Server error",
Toast.LENGTH_SHORT).show();
                    }
                });
            }
        }
    });
  }

  @Override
  public boolean onCreateOptionsMenu(Menu menu) {
      // Inflate the menu; this adds items to the action bar if it is present.
      getMenuInflater().inflate(R.menu.menu_main, menu);
      return true;
  }

  @Override
  public boolean onOptionsItemSelected(MenuItem item) {
      // Handle action bar item clicks here. The action bar will
      // automatically handle clicks on the Home/Up button, so long
      // as you specify a parent activity in AndroidManifest.xml.
      int id = item.getItemId();



      if (id == R.id.action_help) {

          Intent myIntent = new Intent(MainActivity.this,HelpActivity.class);
          startActivity(myIntent);
          return true;
      }

      return super.onOptionsItemSelected(item);
  }

  @Override
  protected void onResume() {
      super.onResume();
      getFlowers();
  }

}
```

11. Activity_add_flower.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".AddFlower"
    android:padding="10dp"
    >
    <android.support.design.widget.TextInputLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:id="@+id/ilfno"
        >

        <android.support.design.widget.TextInputEditText
        android:id="@+id/etFNO"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Flora Number " />
  </android.support.design.widget.TextInputLayout>

  <android.support.design.widget.TextInputLayout
      android:layout_width="match_parent"
      android:layout_height="60dp"
      android:layout_alignParentStart="true"
      android:layout_below="@+id/ilfno"
      android:layout_alignParentLeft="true"
      android:id="@+id/ilname"
      >

      <android.support.design.widget.TextInputEditText
        android:id="@+id/etName"
        android:layout_width="match_parent"
        android:layout_height="60dp"
        android:hint="Flora Name" />
  </android.support.design.widget.TextInputLayout>

  <android.support.design.widget.TextInputLayout
      android:layout_width="match_parent"
      android:layout_height="100dp"
      android:layout_alignParentStart="true"
      android:layout_below="@+id/ilname"
      android:layout_alignParentLeft="true"
      android:id="@+id/ildescription"
      >

      <android.support.design.widget.TextInputEditText
```

```
            android:id="@+id/etDescription"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:hint="Description" />
    </android.support.design.widget.TextInputLayout>

    <Button
        android:id="@+id/button"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/ildescription"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="4dp"
        android:background="@color/colorAccent"
        android:text="Add Flower"
        android:textStyle="bold|italic" />
    <ImageView
        android:id="@+id/imageView2"
        android:layout_width="match_parent"
        android:layout_marginTop="40dp"
        android:layout_height="300dp"
        android:layout_below="@+id/button"
        />
</RelativeLayout>
```

12. Activity_help.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.CoordinatorLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
```

```
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:fitsSystemWindows="true"
  tools:context=".HelpActivity">


  <android.support.design.widget.AppBarLayout
      android:id="@+id/app_bar"
      android:layout_width="match_parent"
      android:layout_height="@dimen/app_bar_height"
      android:fitsSystemWindows="true"
      android:theme="@style/AppTheme.AppBarOverlay">

      <android.support.design.widget.CollapsingToolbarLayout
          android:id="@+id/toolbar_layout"
          android:layout_width="match_parent"
          android:layout_height="match_parent"
          android:fitsSystemWindows="true"
          app:contentScrim="?attr/colorPrimary"
          app:layout_scrollFlags="scroll|exitUntilCollapsed"
          app:toolbarId="@+id/toolbar">

          <android.support.v7.widget.Toolbar
              android:id="@+id/toolbar"
              android:layout_width="match_parent"
              android:layout_height="?attr/actionBarSize"
              app:layout_collapseMode="pin"
              app:popupTheme="@style/AppTheme.PopupOverlay" />

      </android.support.design.widget.CollapsingToolbarLayout>
  </android.support.design.widget.AppBarLayout>
  <include layout="@layout/content_help" />

  <android.support.design.widget.FloatingActionButton
      android:id="@+id/fab"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_margin="@dimen/fab_margin"
      app:layout_anchor="@id/app_bar"
      app:layout_anchorGravity="bottom|end"
      app:srcCompat="@android:drawable/ic_dialog_email" />

  <!--<ImageView
      android:id="@+id/imageView5"
      android:layout_width="100dp"
      android:layout_marginTop="400dp"
      android:layout_marginLeft="150dp"
      android:layout_height="100dp"
      app:srcCompat="@drawable/logo" />-->

  <android.support.v4.view.ViewPager
      android:id="@+id/view_pager"
      android:layout_width="320dp"
      android:layout_marginTop="450dp"
      android:layout_marginLeft="30dp"
      android:layout_height="200dp"/>

  <TextView
      android:layout_width="250dp"
      android:layout_height="20dp"
      android:layout_marginLeft="60dp"
      android:layout_marginTop="420dp"
```

```
       android:text="Swipe left/right for more images"
       android:textAllCaps="true"
       android:textAppearance="@style/TextAppearance.AppCompat.Body1" />



</android.support.design.widget.CoordinatorLayout>
```

13. Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.CoordinatorLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context="budasuyasa.android.simplecrud.MainActivity">

   <android.support.design.widget.AppBarLayout
       android:layout_width="match_parent"
       android:layout_height="wrap_content"
       android:theme="@style/AppTheme.AppBarOverlay">

       <android.support.v7.widget.Toolbar
           android:id="@+id/toolbar"
           android:layout_width="match_parent"
           android:layout_height="?attr/actionBarSize"
           android:background="?attr/colorPrimary"
           app:popupTheme="@style/AppTheme.PopupOverlay" />
```

```
  </android.support.design.widget.AppBarLayout>

  <include layout="@layout/content_main" />

  <android.support.design.widget.FloatingActionButton
      android:id="@+id/fab"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_gravity="bottom|end"
      android:layout_margin="@dimen/fab_margin"
      app:srcCompat="@drawable/ic_add_white_24dp" />

</android.support.design.widget.CoordinatorLayout>
```

14. Activity_update_flower.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".UpdateFlower">

</android.support.constraint.ConstraintLayout>
```

15. Content_help.xml

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<android.support.v4.widget.NestedScrollView
xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  app:layout_behavior="@string/appbar_scrolling_view_behavior"
  tools:context=".HelpActivity"
  tools:showIn="@layout/activity_help">

  <TextView
      android:id="@+id/my_text"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_margin="@dimen/text_margin"
      android:text="
      Flora Bucket is a free application where you can use to view pictures and information
about any kind of flora available in the world. Our home page displays all the images and
descriptions that we are having on flowers in our data stores. You can insert a new flower
with the image to share your knowledge with others. As the admin privileges we have
provided the access to update and delete the available data. We are hoping to limit those
privileges in upcoming versions. However feel free to share your ideas by connecting with us
via email. Email :- florabucket.gmail.com. Use the above message icon to navigate to google
accounts. We always welcome your ideas to improve this application much more useful to
you.
"
      android:textAppearance="@style/TextAppearance.AppCompat"
      android:textStyle="bold|italic"
      android:typeface="sans"
      app:fontFamily="sans-serif-condensed" />
```

```
</android.support.v4.widget.NestedScrollView>
```

16. Content_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   app:layout_behavior="@string/appbar_scrolling_view_behavior"
   tools:context="budasuyasa.android.simplecrud.MainActivity"
   tools:showIn="@layout/activity_main"
   android:background="#ffffff"
   >

   <android.support.v7.widget.RecyclerView
       android:id="@+id/listView"
       android:padding="10dp"
       android:layout_width="match_parent"
       android:layout_height="match_parent"
       tools:layout_editor_absoluteX="8dp"
       tools:layout_editor_absoluteY="8dp"

       />
</RelativeLayout>
```

17. Flower_list_item.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
```

```
  xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:padding="5dp"
  android:layout_marginBottom="2dp"
  >

  <ImageView
      android:id="@+id/imageView"
      android:layout_width="100dp"
      android:layout_height="125dp"
      android:layout_alignParentLeft="true"
      android:layout_alignParentStart="true"
      android:layout_alignParentTop="true"
      app:srcCompat="@android:drawable/ic_menu_gallery"
      android:layout_marginRight="5dp"
      />

  <TextView
      android:id="@+id/tvTitle"
      android:textSize="17sp"
      android:textStyle="bold"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_alignParentTop="true"
      android:layout_toEndOf="@+id/imageView"
      android:layout_toRightOf="@+id/imageView"
      android:text="TextView" />

  <TextView
      android:id="@+id/tvDescription"
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/tvTitle"
    android:layout_toEndOf="@+id/imageView"
    android:layout_toRightOf="@+id/imageView"
    android:text="TextView" />

</RelativeLayout>
```

18. Menu_add.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:android="http://schemas.android.com/apk/res/android">
  <item android:id="@+id/save"
    android:title="Save"
    app:showAsAction="ifRoom" />

</menu>
```

19. Menu_help.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  tools:context="budasuyasa.android.simplecrud.HelpActivity">
  <item
    android:id="@+id/action_settings"
    android:orderInCategory="100"
    android:title="@string/action_settings"
    app:showAsAction="never" />
```

```
</menu>
```

20. Menu_main.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  tools:context="budasuyasa.android.simplecrud.MainActivity">
  <item
      android:id="@+id/action_help"
      android:orderInCategory="100"
      android:title="@string/action_help"
      app:showAsAction="never" />
</menu>
```

21. Styles.xml

```
<resources>

  <!-- Base application theme. -->
  <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
      <!-- Customize your theme here. -->
      <item name="colorPrimary">@color/colorPrimary</item>
      <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
      <item name="colorAccent">@color/colorAccent</item>
  </style>

  <style name="AppTheme.NoActionBar">
      <item name="windowActionBar">false</item>
      <item name="windowNoTitle">true</item>
  </style>
  <style name="AppTheme.AppBarOverlay"
parent="ThemeOverlay.AppCompat.Dark.ActionBar" />

  <style name="AppTheme.PopupOverlay" parent="ThemeOverlay.AppCompat.Light" />

</resources>
```

22. Strings.xml

```
<resources>
  <string name="app_name">Flora Bucket</string>
  <string name="action_help">Help</string>
  <string name="title_activity_help">Help</string>
  <string name="action_settings">Settings</string>
</resources>
```