



TUGAS AKHIR PEMROGRAMAN BERGERAK

“JEJAK MASJID”

Disusun oleh:

Fathur Rahman Haikal (4617010028)

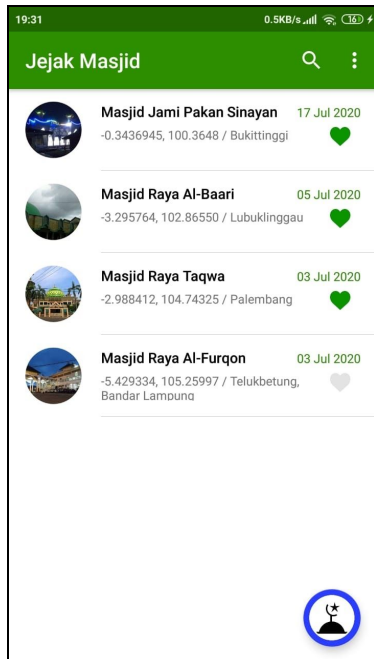
TI 6A

PROGRAM STUDI TEKNIK INFORMATIKA

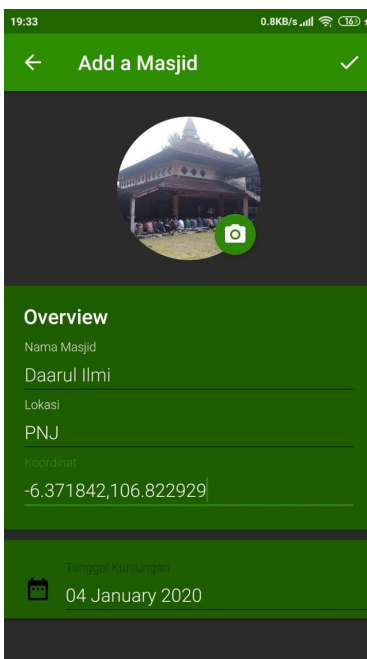
JURUSAN TEKNIK INFORMATIKA DAN KOMPUTER

POLITEKNIK NEGERI JAKARTA

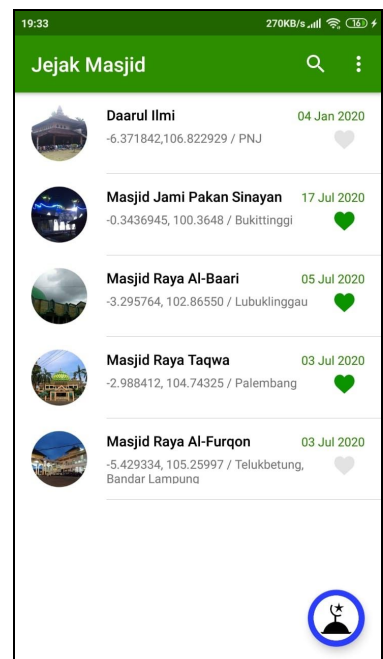
Cara Penggunaan



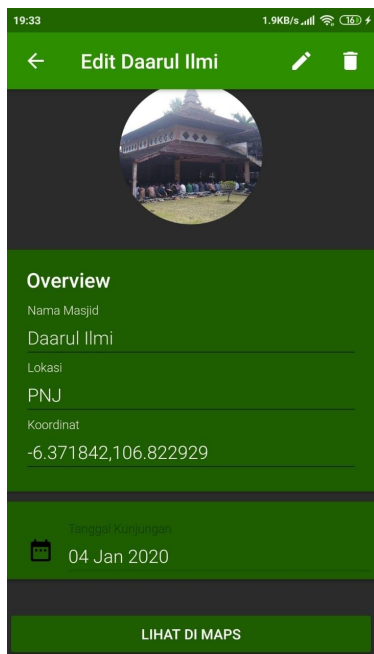
Main Activity



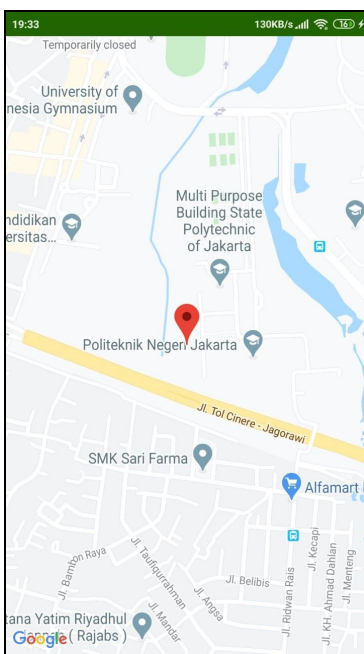
Add Masjid



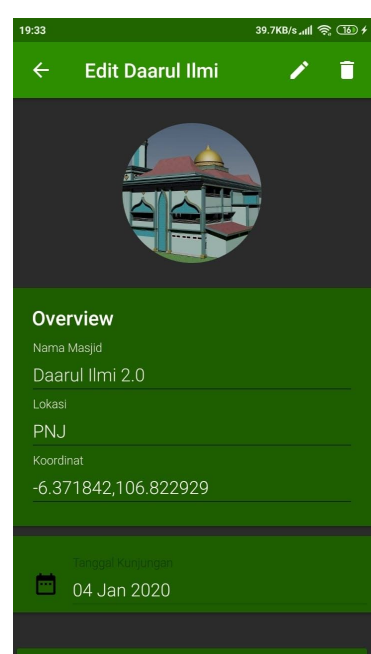
Masjid bertambah



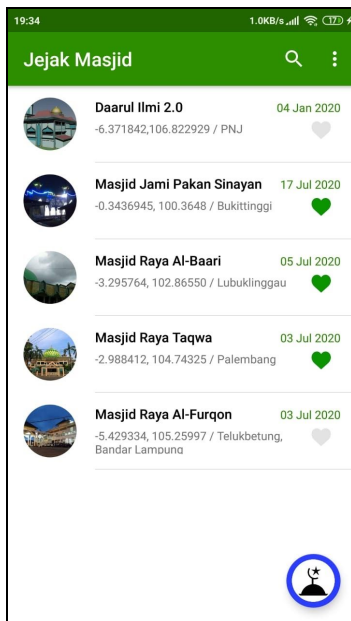
Detail masjid



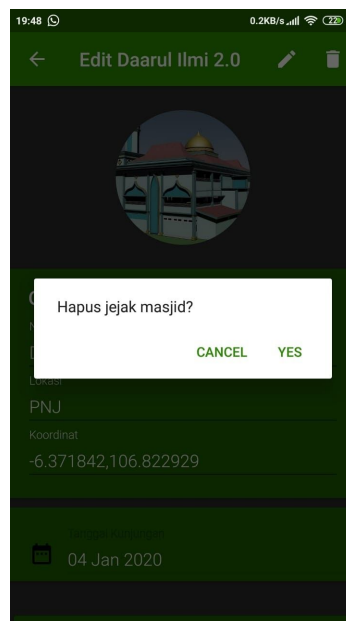
Maps



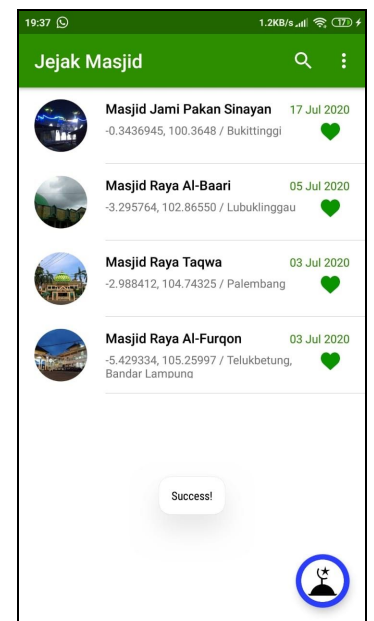
Edit masjid



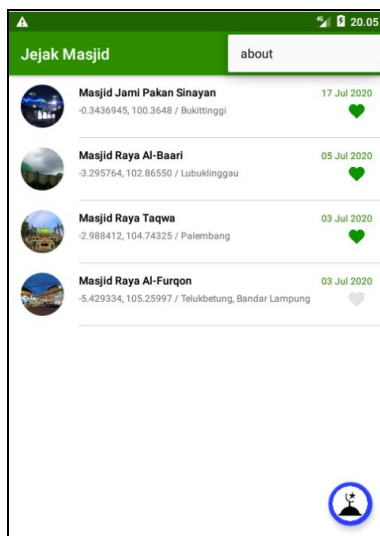
Data masjid berubah



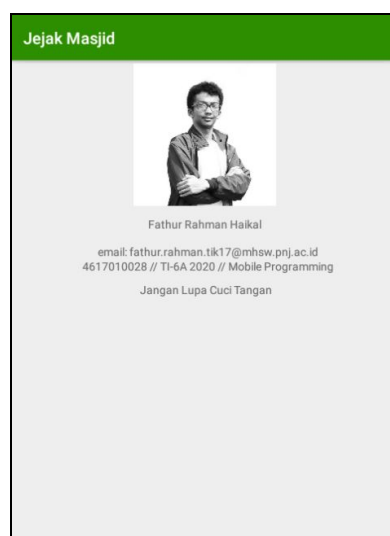
hapus masjid



masjid terhapus



Item about



About

Proses Pengerjaan

API Interface

<pre>public interface ApilInterface { @POST("get_masjid.php") Call<List<Masjid>> getMasjid(); }</pre>	Inisialisasi Interface API dan fungsi get untuk semua datanya
<pre>@FormUrlEncoded @POST("add_masjid.php") Call<Masjid> insertMasjid(@Field("key") String key, @Field("name") String name, @Field("lokasi") String lokasi, @Field("koordinat") String koordinat, @Field("tanggal") String tanggal, @Field("picture") String picture);</pre>	Fungsi insert
<pre>@FormUrlEncoded @POST("update_masjid.php") Call<Masjid> updateMasjid(@Field("key") String key, @Field("id") int id, @Field("name") String name, @Field("lokasi") String lokasi, @Field("koordinat") String koordinat, @Field("tanggal") String tanggal, @Field("picture") String picture);</pre>	Fungsi Edit
<pre>@FormUrlEncoded @POST("delete_masjid.php") Call<Masjid> deleteMasjid(@Field("key") String key, @Field("id") int id, @Field("picture") String picture); }</pre>	Fungsi Hapus

API Client

```
class ApiClient {  
  
    private static final String BASE_URL =  
    "http://192.168.1.2/masjid_server/";  
    private static Retrofit retrofit;  
  
    static Retrofit getClient(){  
  
        if (retrofit==null){  
            retrofit = new Retrofit.Builder()  
                .baseUrl(BASE_URL)  
  
                .addConverterFactory(GsonConverterFactory.create())  
                .build();  
  
        }  
        return retrofit;  
    }  
}
```

API Untuk koneksi ke URL server

Model

```
public class Masjid {  
    @SerializedName("id")  
    private int id;  
    @SerializedName("name")  
    private String name;  
    @SerializedName("lokasi")  
    private String lokasi;  
    @SerializedName("koordinat")  
    private String koordinat;  
    @SerializedName("tanggal")  
    private String tanggal;  
    @SerializedName("picture")  
    private String picture;  
    @SerializedName("love")  
    private Boolean love;  
    @SerializedName("value")  
    private String value;  
    @SerializedName("message")  
    private String message;  
}
```

Model untuk dipergunakan dalam Aplikasi

```
public int getId() { return id; }  
public void setId(int id) { this.id = id; }  
public String getName() { return name; }  
public void setName(String name) { this.name = name; }  
public String getLokasi() { return lokasi; }  
public void setLokasi(String lokasi) { this.lokasi = lokasi; }  
public String getKoordinat() { return koordinat; }  
public void setKoordinat(String koordinat) { this.koordinat = koordinat; }  
public String getTanggal() { return tanggal; }  
public void setTanggal(String tanggal) { this.tanggal = tanggal; }  
public String getPicture() { return picture; }  
public void setPicture(String picture) { this.picture = picture; }  
public Boolean getLove() { return love; }  
public void setLove(Boolean love) { this.love = love; }  
public String getValue() { return value; }  
public void setValue(String value) { this.value = value; }  
public String getMessage() { return message; }  
public void setMessage(String message) { this.message = message; }
```

Getter dan Setter dari model

Web Server

<pre><?php define('host', 'localhost'); define('user', 'root'); define('pass', ''); define('db', 'masjid_db'); \$conn = mysqli_connect(host, user, pass, db) or die('Unable to Connect'); ?></pre>	Connect
<pre><?php header("Content-Type: application/json; charset=UTF-8"); require_once 'connect.php'; \$query = "SELECT * FROM masjid ORDER BY id DESC "; \$result = mysqli_query(\$conn, \$query); \$response = array(); \$server_name = \$_SERVER['SERVER_ADDR']; while(\$row = mysqli_fetch_assoc(\$result)){ array_push(\$response, array('id' =>\$row['id'], 'name' =>\$row['name'], 'lokasi' =>\$row['lokasi'], 'koordinat' =>\$row['koordinat'], 'tanggal' =>date('d M Y', strtotime(\$row['tanggal'])), 'picture' =>"http://\$server_name" . \$row['picture'], 'love' =>\$row['love'])); } echo json_encode(\$response); mysqli_close(\$conn); ?></pre>	Get
<pre><?php require_once 'connect.php'; \$key = \$_POST['key']; \$name = \$_POST['name']; \$lokasi = \$_POST['lokasi']; \$koordinat = \$_POST['koordinat']; \$tanggal = \$_POST['tanggal']; \$picture = \$_POST['picture']; if (\$key == "insert"){ \$tanggal_newformat = date('Y-m-d', strtotime(\$tanggal)); \$query = "INSERT INTO masjid (name,lokasi,koordinat,tanggal) VALUES ('\$name', '\$lokasi', '\$koordinat', '\$tanggal_newformat') "; if (mysqli_query(\$conn, \$query)){ if (\$picture == null) { \$finalPath = "/masjid_server/logo.png"; \$result["value"] = "1"; \$result["message"] = "Success"; echo json_encode(\$result); } } }</pre>	Add

```

        mysqli_close($conn);

    } else {

        $id = mysqli_insert_id($conn);
        $path = "masjid_picture/$id.jpeg";
        $finalPath = "/masjid_server/".$path;

        $insert_picture = "UPDATE masjid SET picture='$finalPath'
WHERE id='$id' ";

        if (mysqli_query($conn, $insert_picture)) {

            if ( file_put_contents( $path, base64_decode($picture) )
) {

                $result["value"] = "1";
                $result["message"] = "Success!";

                echo json_encode($result);
                mysqli_close($conn);

            } else {

                $response["value"] = "0";
                $response["message"] = "Error! ".mysqli_error($conn);
                echo json_encode($response);

                mysqli_close($conn);

            }

        }

    }

}

else {
    $response["value"] = "0";
    $response["message"] = "Error! ".mysqli_error($conn);
    echo json_encode($response);

    mysqli_close($conn);
}

}

?>

```

```
<?php
```

Delete

```

header("Content-Type: application/json; charset=UTF-8");

require_once 'connect.php';

$key = $_POST['key'];
$id   = $_POST['id'];
$picture = $_POST['picture'];

if ( $key == "delete" ){

    $query = "DELETE FROM masjid WHERE id='$id' ";

    if ( mysqli_query($conn, $query) ){

        $iparr = explode("/", $picture);
        $picture_split = $iparr[5];

        if ( unlink("masjid_picture/".$picture_split) ){

            $result["value"] = "1";
            $result["message"] = "Success!";

            echo json_encode($result);
            mysqli_close($conn);

        } else {

            $response["value"] = "0";
            $response["message"] = "Error to delete a image!";
            $.mysqli_error($conn);
            echo json_encode($response);

            mysqli_close($conn);

        }

    }

} else {

    $response["value"] = "0";
    $response["message"] = "Error! " . $.mysqli_error($conn);
    echo json_encode($response);

    mysqli_close($conn);

}

?>

```



```
<?php

header("Content-Type: application/json; charset=UTF-8");

require_once 'connect.php';

$key = $_POST['key'];

if ( $key == "update" ){

    $id          = $_POST['id'];
    $name        = $_POST['name'];
    $lokasi      = $_POST['lokasi'];
    $koordinat   = $_POST['koordinat'];
    $tanggal     = $_POST['tanggal'];
    $picture     = $_POST['picture'];

    $tanggal = date('Y-m-d', strtotime($tanggal));

    $query = "UPDATE masjid SET
name='$name',
lokasi='$lokasi',
koordinat='$koordinat',
tanggal='$tanggal'
WHERE id='$id' ";

    if ( mysqli_query($conn, $query) ){

        if ($picture == null) {

            $result["value"] = "1";
            $result["message"] = "Success";

            echo json_encode($result);
            mysqli_close($conn);

        } else {

            $path = "masjid_picture/$id.jpeg";
            $finalPath = "/masjid_server/".$path;

            $insert_picture = "UPDATE masjid SET picture='$finalPath'
WHERE id='$id' ";

            if (mysqli_query($conn, $insert_picture)) {

                if ( file_put_contents( $path, base64_decode($picture) )
) {

                    $result["value"] = "1";
                    $result["message"] = "Success!";

                    echo json_encode($result);
                    mysqli_close($conn);
```

```
        } else {

            $response["value"] = "0";
            $response["message"] = "Error! ".mysqli_error($conn);
            echo json_encode($response);

            mysqli_close($conn);

        }

    }

}

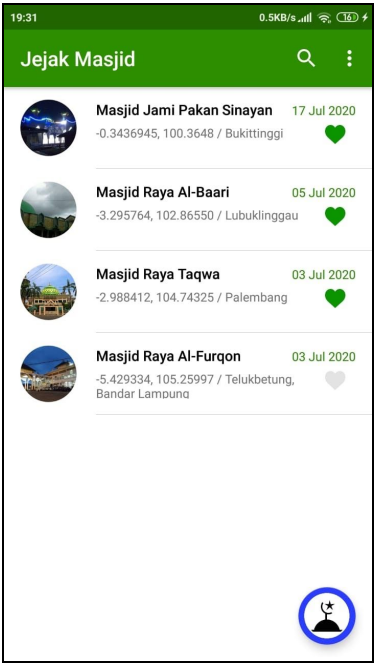
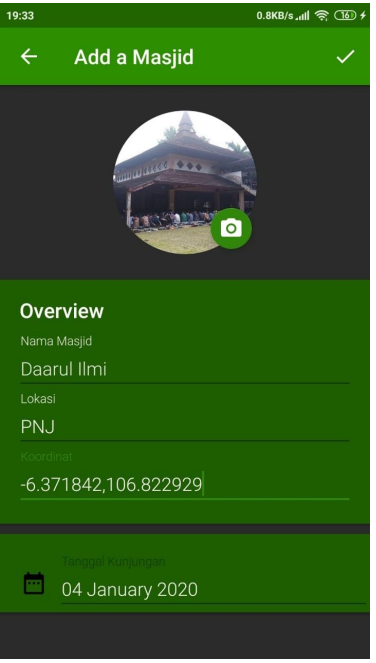
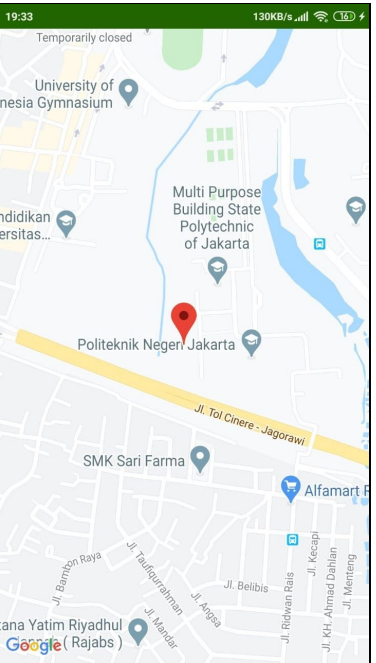
else {
    $response["value"] = "0";
    $response["message"] = "Error! ".mysqli_error($conn);
    echo json_encode($response);


    mysqli_close($conn);
}

}

?>
```

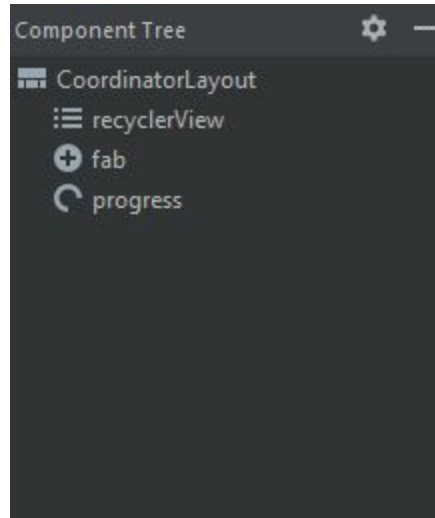
Activities

 <p>Main Activity</p>	 <p>Edit Activity</p>	 <p>Maps Activity</p>
<p>Memiliki 2 menu serta Item List.</p>	<p>Berisi form yang digunakan untuk edit dan tambah, juga ada menu hapus.</p>	<p>Berisi Maps yang memiliki pin tepat di koordinat yang diberikan.</p>

 <p>Item list dari Main Activity seperti berikut</p>	 <p>Menu pada activity edit yang berubah. Pensil dan Trashbin ketika detail, centang ketika edit.</p>
-----------------------------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------



Layout dari activity main, hanya terdapat list untuk recyclerview serta floating action button



```
public class MainActivity extends AppCompatActivity{

    private RecyclerView recyclerView;
    private RecyclerView.LayoutManager layoutManager;
    private Adapter adapter;
    private List<Masjid> masjidList;
    ApiInterface apiInterface;
    Adapter.RecyclerViewClickListener listener;
    ProgressBar progressBar;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        apiInterface = ApiClient.getClient().create(ApiInterface.class);

        progressBar = findViewById(R.id.progress);
        recyclerView = findViewById(R.id.recyclerview);

        layoutManager = new LinearLayoutManager(this);
        recyclerView.setLayoutManager(layoutManager);
    }
}
```

MainActivity, saat Create akan langsung membuat koneksi menggunakan APIClient dan ApiInterface, memuat seluruh item yang ada pada View

```
listener = new Adapter.RecyclerViewClickListener() {
    @Override
    public void onClick(View view, final int position) {
```

```
        Intent intent = new Intent(MainActivity.this, EditorActivity.class);
        intent.putExtra("id", masjidList.get(position).getId());
        intent.putExtra("name", masjidList.get(position).getName());
        intent.putExtra("lokasi", masjidList.get(position).getLokasi());
        intent.putExtra("koordinat", masjidList.get(position).getKoordinat());
        intent.putExtra("picture", masjidList.get(position).getPicture());
        intent.putExtra("tanggal", masjidList.get(position).getTanggal());
        startActivity(intent);
```

```
    }

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

```
        final int id = masjidList.get(position).getId();
        final Boolean love = masjidList.get(position).getLove();
        final ImageView mLove = view.findViewById(R.id.love);
```

```
        if (love){
            mLove.setImageResource(R.drawable.likeof);
            masjidList.get(position).setLove(false);
            updateLove("update_love", id, false);
            adapter.notifyDataSetChanged();
        } else {
            mLove.setImageResource(R.drawable.likeon);
            masjidList.get(position).setLove(true);
            updateLove("update_love", id, true);
            adapter.notifyDataSetChanged();
        }
    }
}
```

```
FloatingActionButton fab = findViewById(R.id.fab);
fab.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        startActivity(new Intent(MainActivity.this, EditorActivity.class));
    }
});
```

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
```

```
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.menu_main, menu);
```

```
    SearchManager searchManager = (SearchManager)
```

Juga terdapat
onClickListener dari
Adapter untuk membaca
click pada row, sehingga
akan dapat berpindah
activity ke detail.

Juga terdapat
onLoveClick untuk
menandai row dengan
Love

Menampilkan menu di
activity main

```

getSystemService(Context.SEARCH_SERVICE);
final SearchView searchView = (SearchView)
menu.findItem(R.id.action_search).getActionView();
MenuItem searchMenuItem = menu.findItem(R.id.action_search);

searchView.setSearchableInfo(
    searchManager.getSearchableInfo(getComponentName())
);
searchView.setQueryHint("Search Masjid...");
searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
    @Override
    public boolean onQueryTextSubmit(final String query) {

        adapter.getFilter().filter(query);
        return false;
    }

    @Override
    public boolean onQueryTextChange(String newText) {

        adapter.getFilter().filter(newText);
        return false;
    }
});

searchMenuItem.setIcon().setVisible(false, false);

return true;
}

```

```

@Override
public boolean onOptionsItemSelected(MenuItem item) {

    int id = item.getItemId();

    if (id == R.id.about) {
        Intent intent = new Intent(this, About.class);
        startActivity(intent);
    }
    return super.onOptionsItemSelected(item);
}

```

```

public void getMasjid(){

    Call<List<Masjid>> call = apiInterface.getMasjid();
    call.enqueue(new Callback<List<Masjid>>() {
        @Override
        public void onResponse(Call<List<Masjid>> call, Response<List<Masjid>>
response) {
            progressBar.setVisibility(View.GONE);
            masjidList = response.body();
            Log.i(MainActivity.class.getSimpleName(), response.body().toString());
        }
    });
}

```

Memilih item pada menu, ada about.

Menampilkan seluruh masjid

<pre> adapter = new Adapter(masjidList, MainActivity.this, listener); recyclerView.setAdapter(adapter); adapter.notifyDataSetChanged(); } @Override public void onFailure(Call<List<Masjid>> call, Throwable t) { Toast.makeText(MainActivity.this, "rp :"+ t.getMessage().toString(), Toast.LENGTH_SHORT).show(); } }; } </pre>	
<pre> public void updateLove(final String key, final int id, final Boolean love){ Call<Masjid> call = apiInterface.updateLove(key, id, love); call.enqueue(new Callback<Masjid>() { @Override public void onResponse(Call<Masjid> call, Response<Masjid> response) { Log.i(MainActivity.class.getSimpleName(), "Response "+response.toString()); String value = response.body().getValue(); String message = response.body().getMessage(); if (value.equals("1")){ Toast.makeText(MainActivity.this, message, Toast.LENGTH_SHORT).show(); } else { Toast.makeText(MainActivity.this, message, Toast.LENGTH_SHORT).show(); } } }); @Override public void onFailure(Call<Masjid> call, Throwable t) { Toast.makeText(MainActivity.this, t.getMessage().toString(), Toast.LENGTH_SHORT).show(); } }; } </pre>	<p>Update Love, fungsi yang tadi dipanggil oleh onLoveClick, untuk memperbarui data love di database</p>
<pre> @Override protected void onResume() { super.onResume(); getMasjid(); } </pre>	<p>Resume, memanggil fungsi getMasjid pada Interface</p>
<pre> public class EditorActivity extends AppCompatActivity { private EditText mName, mlokasi, mKoordinat, </pre>	<p>EDITOR ACTIVITY</p>

```

mTanggal, mMap;
private CircleImageView mPicture;
private FloatingActionButton mFabChoosePic;
private Button button;
Calendar myCalendar = Calendar.getInstance();
private String name, lokasi, koordinat, picture, tanggal;
private int id;
private Menu action;
private Bitmap bitmap;
private ApiInterface apiInterface;

```

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_editor);

    ActionBar actionBar = getSupportActionBar();
    if (actionBar != null) {
        actionBar.setDisplayHomeAsUpEnabled(true);
    }

    mName = findViewById(R.id.name);
    mlokasi = findViewById(R.id.lokasi);
    mKoordinat = findViewById(R.id.koordinat);
    mTanggal = findViewById(R.id.tanggal);
    mPicture = findViewById(R.id.picture);
    mFabChoosePic = findViewById(R.id.fabChoosePic);
    button = findViewById(R.id.button);

    button.setVisibility(View.INVISIBLE);

    mTanggal = findViewById(R.id.tanggal);

    mTanggal.setFocusableInTouchMode(false);
    mTanggal.setFocusable(false);
    mTanggal.setOnClickListener(new
View.OnClickListener() {
    @Override
    public void onClick(View v) {
        new DatePickerDialog(EditorActivity.this, date,
myCalendar
        .get(Calendar.YEAR),
myCalendar.get(Calendar.MONTH),

myCalendar.get(Calendar.DAY_OF_MONTH)).show();
    }
});

    mFabChoosePic.setOnClickListener(new
View.OnClickListener() {
    @Override
    public void onClick(View v) {
        chooseFile();
    }
});

```

Membuat data


```

Intent intent = getIntent();
id = intent.getIntExtra("id", 0);
name = intent.getStringExtra("name");
lokasi = intent.getStringExtra("lokasi");
koordinat = intent.getStringExtra("koordinat");
tanggal = intent.getStringExtra("tanggal");
picture = intent.getStringExtra("picture");

setDataFromIntentExtra();

}
private void setDataFromIntentExtra() {

    if (id != 0) {

        readMode();
        getSupportActionBar().setTitle("Edit " +
name.toString());

        mName.setText(name);
        mlokasi.setText(lokasi);
        mKoordinat.setText(koordinat);
        mTanggal.setText(tanggal);

        RequestOptions requestOptions = new
RequestOptions();
        requestOptions.skipMemoryCache(true);

requestOptions.diskCacheStrategy(DiskCacheStrategy.N
ONE);
        requestOptions.placeholder(R.drawable.logo);
        requestOptions.error(R.drawable.logo);

        Glide.with(EditorActivity.this)
            .load(picture)
            .apply(requestOptions)
            .into(mPicture);

    } else {
        getSupportActionBar().setTitle("Add a Masjid");
    }
}

```

```

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.menu_editor, menu);
    action = menu;
    action.findItem(R.id.menu_save).setVisible(false);

    if (id == 0) {

```

Membuat data

```

        action.findItem(R.id.menu_edit).setVisible(false);
        action.findItem(R.id.menu_delete).setVisible(false);
        action.findItem(R.id.menu_save).setVisible(true);

```

```

    }

```

```

    return true;
}

```

```

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case android.R.id.home:

```

```

        this.finish();

```

```

        return true;
        case R.id.menu_edit:
            //Edit

```

```

            editMode();

```

```

            InputMethodManager imm =
            (InputMethodManager)
            getSystemService(Context.INPUT_METHOD_SERVICE);

```

```

            imm.showSoftInput(mName,
            InputMethodManager.SHOW_IMPLICIT);

```

```

            action.findItem(R.id.menu_edit).setVisible(false);

```

```

            action.findItem(R.id.menu_delete).setVisible(false);
            action.findItem(R.id.menu_save).setVisible(true);

```

```

            return true;
            case R.id.menu_save:
                //Save

```

```

            if (id == 0) {

```

```

                if
                (TextUtils.isEmpty(mName.getText().toString())) ||

```

```

                TextUtils.isEmpty(mlokasi.getText().toString()) ||

```

```

                TextUtils.isEmpty(mKoordinat.getText().toString()) ||

```

```

                TextUtils.isEmpty(mTanggal.getText().toString())) {
                    AlertDialog.Builder alertDialog = new
                    AlertDialog.Builder(this);
                    alertDialog.setMessage("Please complete
                    the field!");
                    alertDialog.setPositiveButton("Ok", new

```

Menghapus item terpilih

```

DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog,
int which) {
        dialog.dismiss();
    }
});
alertDialog.show();
} else {

    postData("insert");

    action.findItem(R.id.menu_edit).setVisible(true);

    action.findItem(R.id.menu_save).setVisible(false);

    action.findItem(R.id.menu_delete).setVisible(true);

    readMode();

}

} else {

    updateData("update", id);
    action.findItem(R.id.menu_edit).setVisible(true);

    action.findItem(R.id.menu_save).setVisible(false);

    action.findItem(R.id.menu_delete).setVisible(true);

    readMode();

}

return true;
case R.id.menu_delete:

    AlertDialog.Builder dialog = new
AlertDialog.Builder(EditorActivity.this);
    dialog.setMessage("Hapus jejak masjid?");
    dialog.setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int
which) {
            dialog.dismiss();
            pulang();
            deleteData("delete", id, picture);

        }
    });
    dialog.setNegativeButton("Cancel", new
DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int

```

<pre> which) { dialog.dismiss(); } }); dialog.show(); return true; default: return super.onOptionsItemSelected(item); } } </pre>	
<pre> private void pulang() { Intent intent = new Intent(this, MainActivity.class); startActivity(intent); } </pre>	Untuk kembali ke main activity setelah hapus
<pre> DatePickerDialog.OnDateSetListener date = new DatePickerDialog.OnDateSetListener() { @Override public void onDateSet(DatePicker view, int year, int monthOfYear, int dayOfMonth) { // TODO Auto-generated method stub myCalendar.set(Calendar.YEAR, year); myCalendar.set(Calendar.MONTH, monthOfYear); myCalendar.set(Calendar.DAY_OF_MONTH, dayOfMonth); setTanggal(); } } </pre>	Listener tanggal
<pre> private void setTanggal() { String myFormat = "dd MMMM yyyy"; //In which you need put here SimpleDateFormat sdf = new SimpleDateFormat(myFormat, Locale.US); mTanggal.setText(sdf.format(myCalendar.getTime())); } </pre>	Set tanggal untuk layout tanggal
<pre> public String getStringImage(Bitmap bmp) { ByteArrayOutputStream baos = new ByteArrayOutputStream(); bmp.compress(Bitmap.CompressFormat.JPEG, 100, baos); byte[] imageBytes = baos.toByteArray(); } </pre>	Mengambil nama file

```

String encodedImage =
Base64.encodeToString(imageBytes,
Base64.DEFAULT);
return encodedImage;
}
private void chooseFile() {
Intent intent = new Intent();
intent.setType("image/*");
intent.setAction(Intent.ACTION_GET_CONTENT);
startActivityForResult(Intent.createChooser(intent,
"Select Picture"), 1);
}

```

```

@Override
protected void onActivityResult(int requestCode, int
resultCode, Intent data) {
super.onActivityResult(requestCode, resultCode, data);
if (requestCode == 1 && resultCode == RESULT_OK
&& data != null && data.getData() != null) {
Uri filePath = data.getData();
try {

bitmap =
MediaStore.Images.Media.getBitmap(getContentResolve
r(), filePath);

mPicture.setImageBitmap(bitmap);

} catch (IOException e) {
e.printStackTrace();
}

}
}

```

```

private void postData(final String key) {

final ProgressDialog progressDialog = new
ProgressDialog(this);
progressDialog.setMessage("Saving...");
progressDialog.show();

readMode();

String name = mName.getText().toString().trim();
String lokasi = mlokasi.getText().toString().trim();
String koordinat =
mKoordinat.getText().toString().trim();
String tanggal = mTanggal.getText().toString().trim();
String picture = null;
if (bitmap == null) {
picture = "";
} else {

```

Mengambil File dari server

post

```

        picture = getStringImage(bitmap);
    }

    apiInterface =
ApiClient.getClient().create(ApiInterface.class);

    Call<Masjid> call = apiInterface.insertMasjid(key,
name, lokasi, koordinat, tanggal, picture);

    call.enqueue(new Callback<Masjid>() {
        @Override
        public void onResponse(Call<Masjid> call,
Response<Masjid> response) {

            progressDialog.dismiss();

            Log.i(EditorActivity.class.getSimpleName(),
response.toString());

            String value = response.body().getValue();
            String message = response.body().getMessage();

            if (value.equals("1")) {
                finish();
            } else {
                Toast.makeText(EditorActivity.this, message,
Toast.LENGTH_SHORT).show();
            }
        }
    });

    @Override
    public void onFailure(Call<Masjid> call, Throwable t)
    {
        progressDialog.dismiss();
        Toast.makeText(EditorActivity.this,
t.getMessage().toString(),
Toast.LENGTH_SHORT).show();
    }
}

```

```

private void updateData(final String key, final int id) {

    final ProgressDialog progressDialog = new
ProgressDialog(this);
    progressDialog.setMessage("Updating...");
    progressDialog.show();

    readMode();

    String name = mName.getText().toString().trim();
    String lokasi = mlokasi.getText().toString().trim();

```

update

```

String koordinat =
mKoordinat.getText().toString().trim();
String tanggal = mTanggal.getText().toString().trim();
String picture = null;
if (bitmap == null) {
    picture = "";
} else {
    picture = getStringImage(bitmap);
}

apiInterface =
ApiClient.getClient().create(ApiInterface.class);

Call<Masjid> call = apiInterface.updateMasjid(key, id,
name, lokasi, koordinat, tanggal, picture);

call.enqueue(new Callback<Masjid>() {
    @Override
    public void onResponse(Call<Masjid> call,
Response<Masjid> response) {

        progressDialog.dismiss();

        Log.i(EditorActivity.class.getSimpleName(),
response.toString());

        String value = response.body().getValue();
        String message = response.body().getMessage();

        if (value.equals("1")) {
            Toast.makeText(EditorActivity.this, message,
Toast.LENGTH_SHORT).show();
        } else {
            Toast.makeText(EditorActivity.this, message,
Toast.LENGTH_SHORT).show();
        }
    }
});

@Override
public void onFailure(Call<Masjid> call, Throwable t)
{
    progressDialog.dismiss();
    Toast.makeText(EditorActivity.this,
t.getMessage().toString(),
Toast.LENGTH_SHORT).show();
}
});
}

```

```

private void deleteData(final String key, final int id, final
String pic) {

    final ProgressDialog progressDialog = new

```

delete

```

ProgressDialog(this);
progressDialog.setMessage("Deleting...");
progressDialog.show();

readMode();

apiInterface =
ApiClient.getClient().create(ApiInterface.class);

Call<Masjid> call = apiInterface.deleteMasjid(key, id,
pic);

call.enqueue(new Callback<Masjid>() {
    @Override
    public void onResponse(Call<Masjid> call,
Response<Masjid> response) {

        progressDialog.dismiss();

        Log.i(EditorActivity.class.getSimpleName(),
response.toString());

        String value = response.body().getValue();
        String message = response.body().getMessage();

        if (value.equals("1")) {
            Toast.makeText(EditorActivity.this, message,
Toast.LENGTH_SHORT).show();
            finish();
        } else {
            Toast.makeText(EditorActivity.this, message,
Toast.LENGTH_SHORT).show();
        }
    }
});

@Override
public void onFailure(Call<Masjid> call, Throwable t)
{
    progressDialog.dismiss();
    Toast.makeText(EditorActivity.this,
t.getMessage().toString(),
Toast.LENGTH_SHORT).show();
}
});
}

```

```

void readMode() {

    mName.setFocusableInTouchMode(false);
    mlokasi.setFocusableInTouchMode(false);
    mKoordinat.setFocusableInTouchMode(false);
    mName.setFocusable(false);
}

```

Mode, untuk menampilkan/menghilangkan button

<pre> mlokasi.setFocusable(false); mKoordinat.setFocusable(false); mTanggal.setEnabled(false); mFabChoosePic.setVisibility(View.INVISIBLE); button.setVisibility(View.VISIBLE); } private void editMode() { mName.setFocusableInTouchMode(true); mlokasi.setFocusableInTouchMode(true); mKoordinat.setFocusableInTouchMode(true); mTanggal.setEnabled(true); mFabChoosePic.setVisibility(View.VISIBLE); button.setVisibility(View.INVISIBLE); } </pre>	
<pre> public void openMaps(View view) { Intent intentz = getIntent(); String coor = intentz.getStringExtra("koordinat"); String[] values = coor.split(","); String x = (values[0]); String y = (values[1]); Intent intent = new Intent(this, MapsActivity.class); intent.putExtra("x", x); intent.putExtra("y", y); startActivity(intent); } </pre>	

Maps

<pre> public class MapsActivity extends FragmentActivity implements OnMapReadyCallback { private GoogleMap mMap; @Override protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.activity_maps); // Obtain the SupportMapFragment and get notified when the map is ready to be used. SupportMapFragment mapFragment = </pre>	
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--

```
(SupportMapFragment) getSupportFragmentManager()
    .findFragmentById(R.id.map);
    mapFragment.getMapAsync(this);
}
```

```
@Override
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;
    Intent intentz = getIntent();
    String x = intentz.getStringExtra("x");
    double v = Double.parseDouble(x);
    String y = intentz.getStringExtra("y");
    double w = Double.parseDouble(y);

    // Add a marker in Sydney and move the camera
    LatLng masjid = new LatLng(v, w);
    mMap.addMarker(new
    MarkerOptions().position(masjid).title("Ini masjidnya"));

    mMap.animateCamera(CameraUpdateFactory.newLatLngZoom(masjid, 16), 5000, null);
}
```

Menampilkan map sesuai koordinat yang diberikan di intent dari EditorActivity serta menampilkan perlahan