



# Application Development for Mobile and Ubiquitous Computing

## Seminar Task

## Final Presentation

GroupNo. 6

Team: Nicolas Fricke, Johannes Maresch

Flashmeet

Flash meet

E-mail

Password

I'M NEW LOGIN

New User

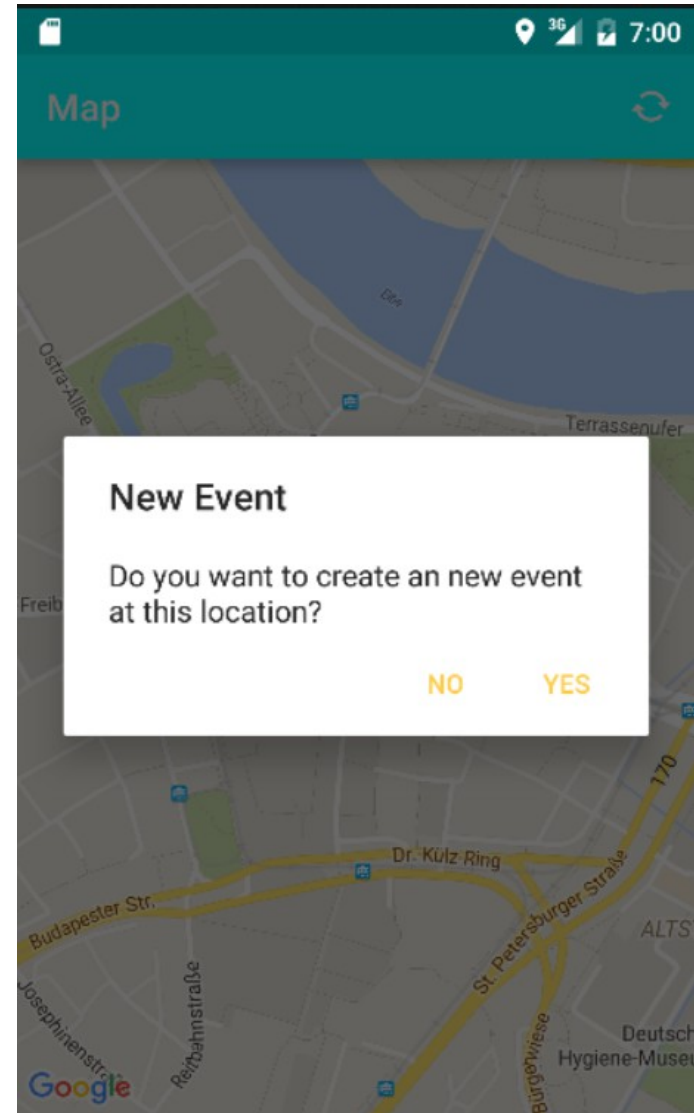
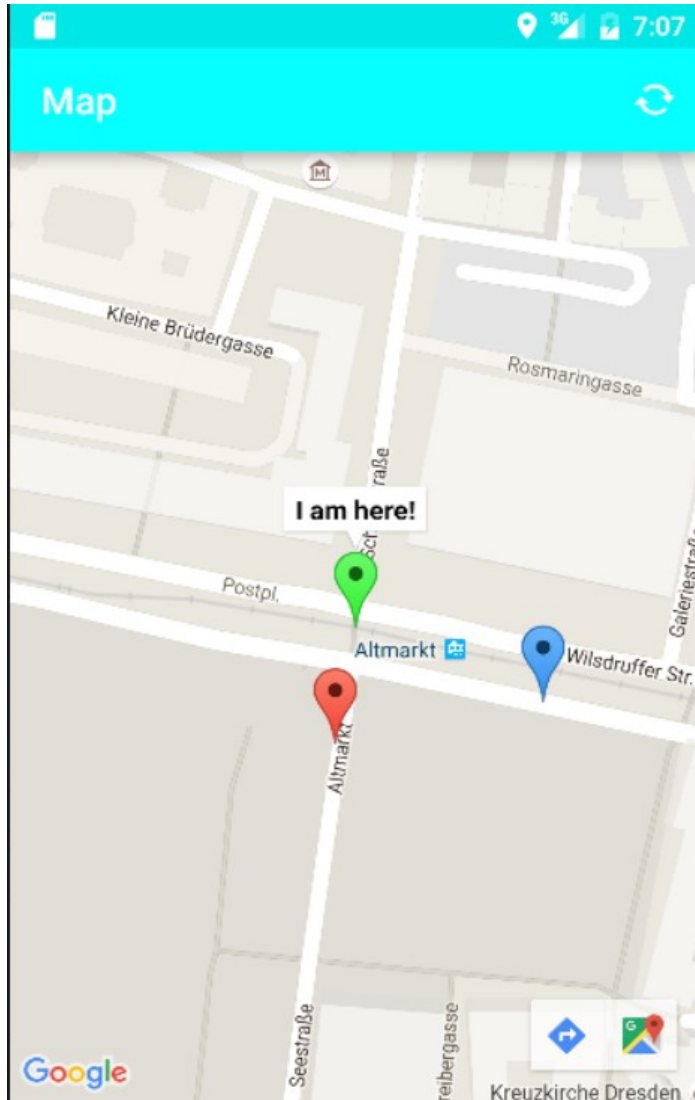
Username

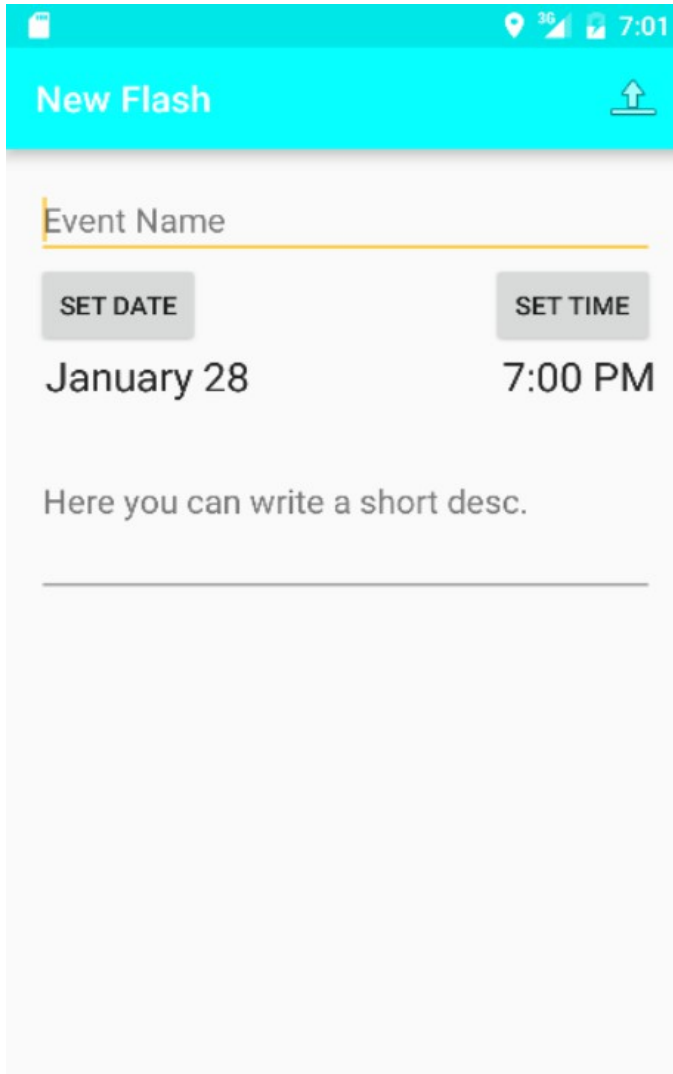
E-mail


Password

Password

REGISTER





**New Flash** 

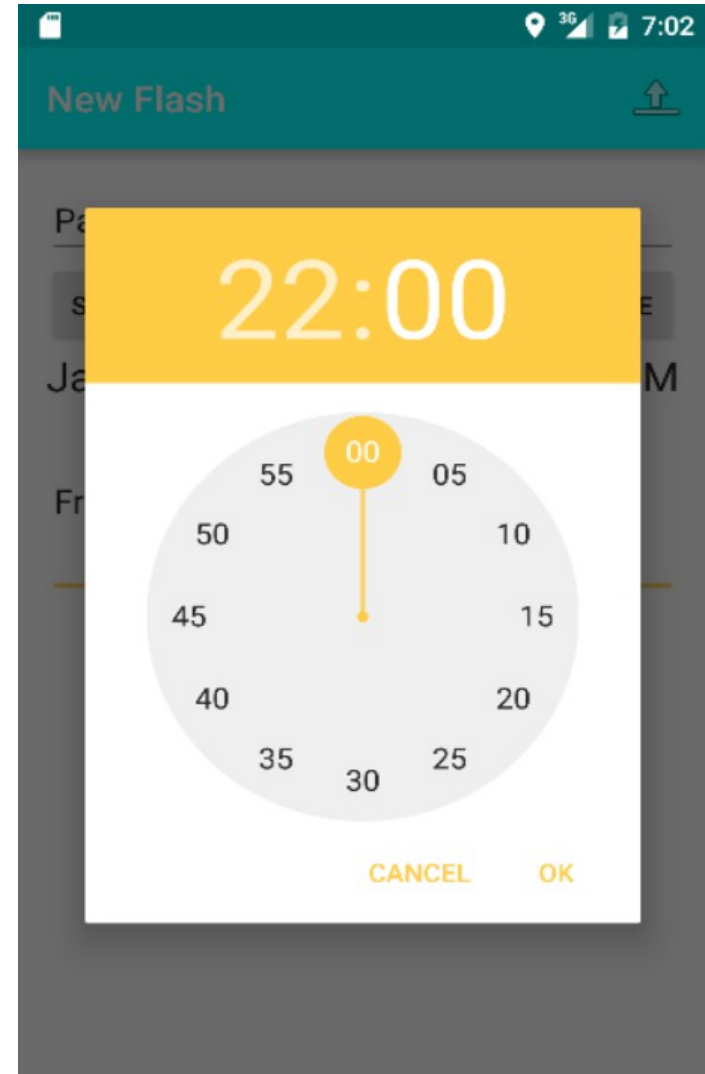
Event Name \_\_\_\_\_


**SET DATE** **SET TIME**

January 28 7:00 PM

Here you can write a short desc.

\_\_\_\_\_



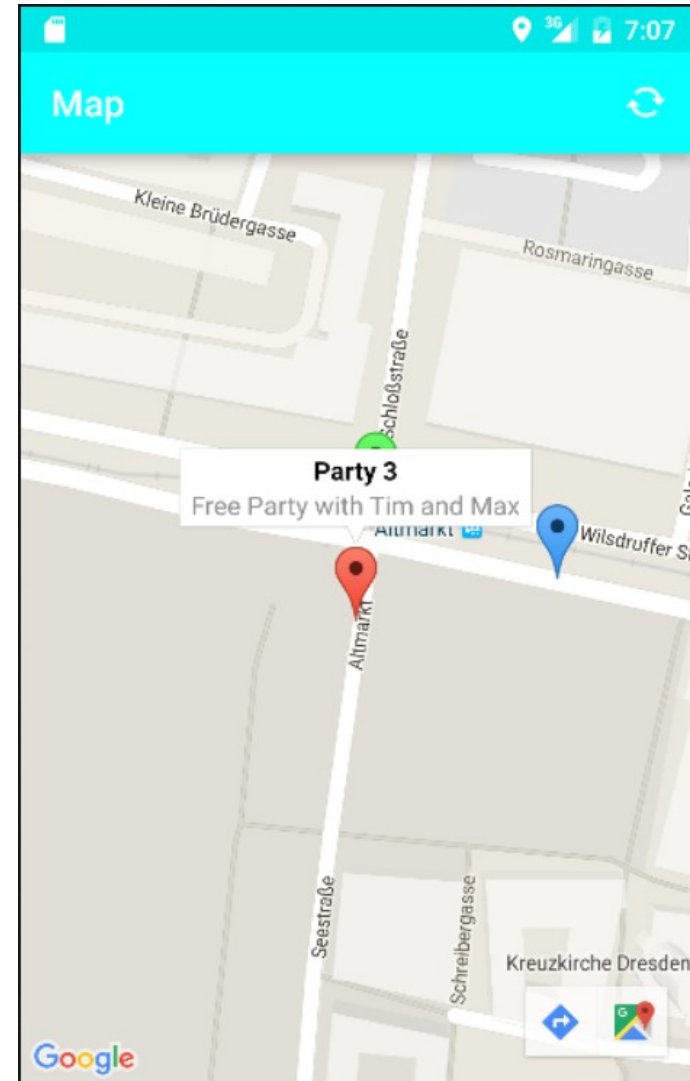
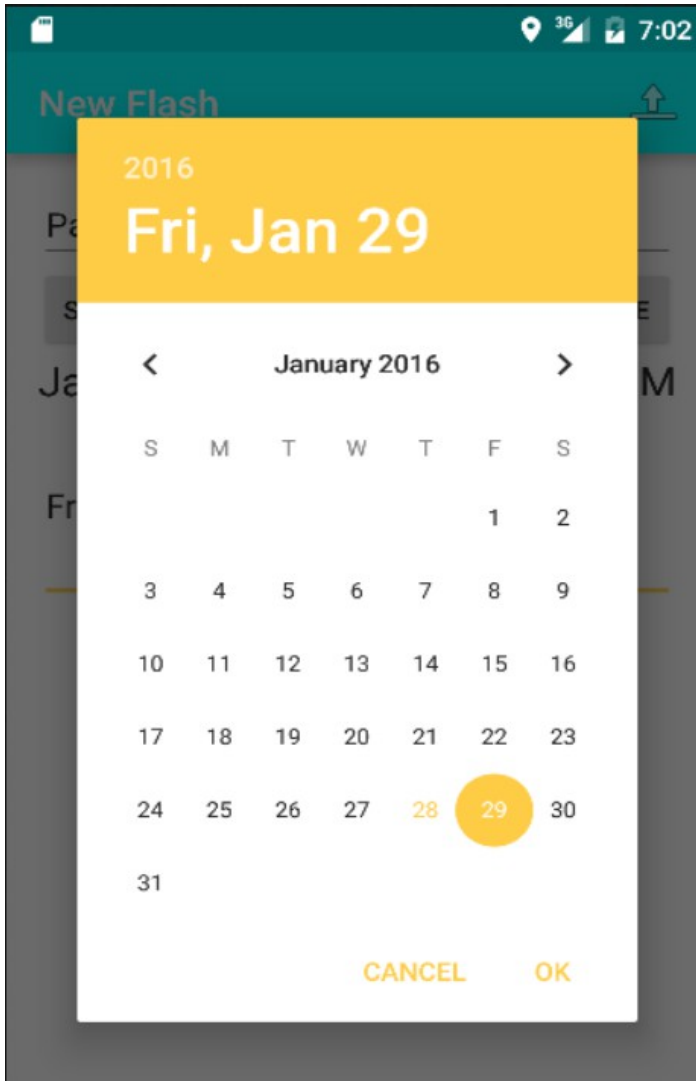
**New Flash** 

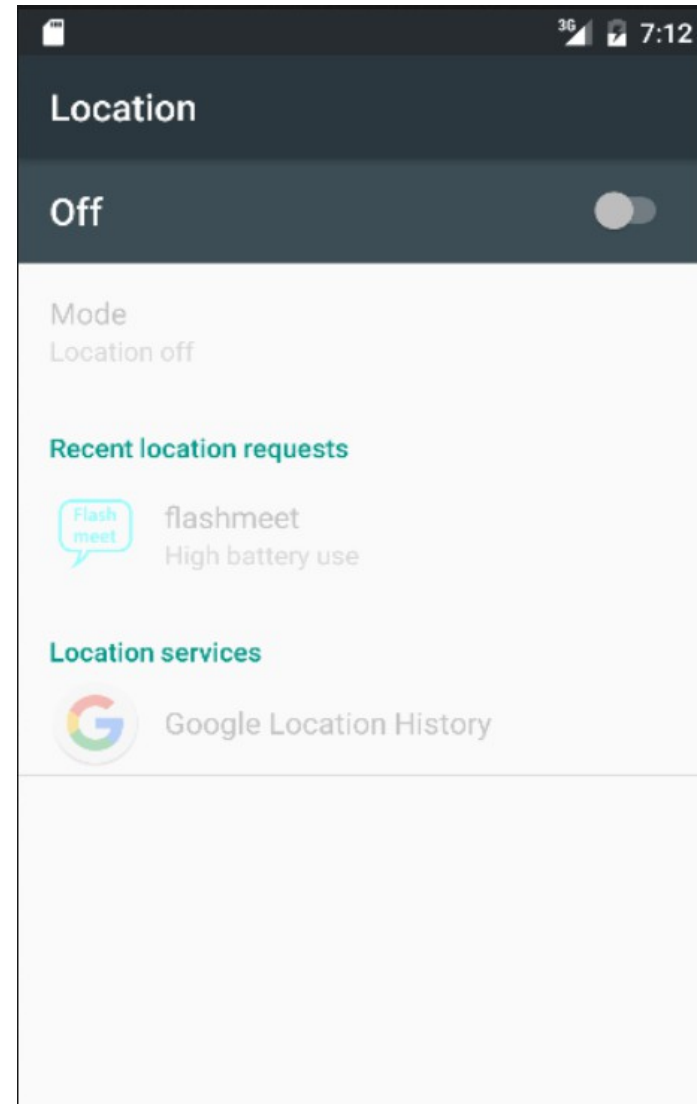
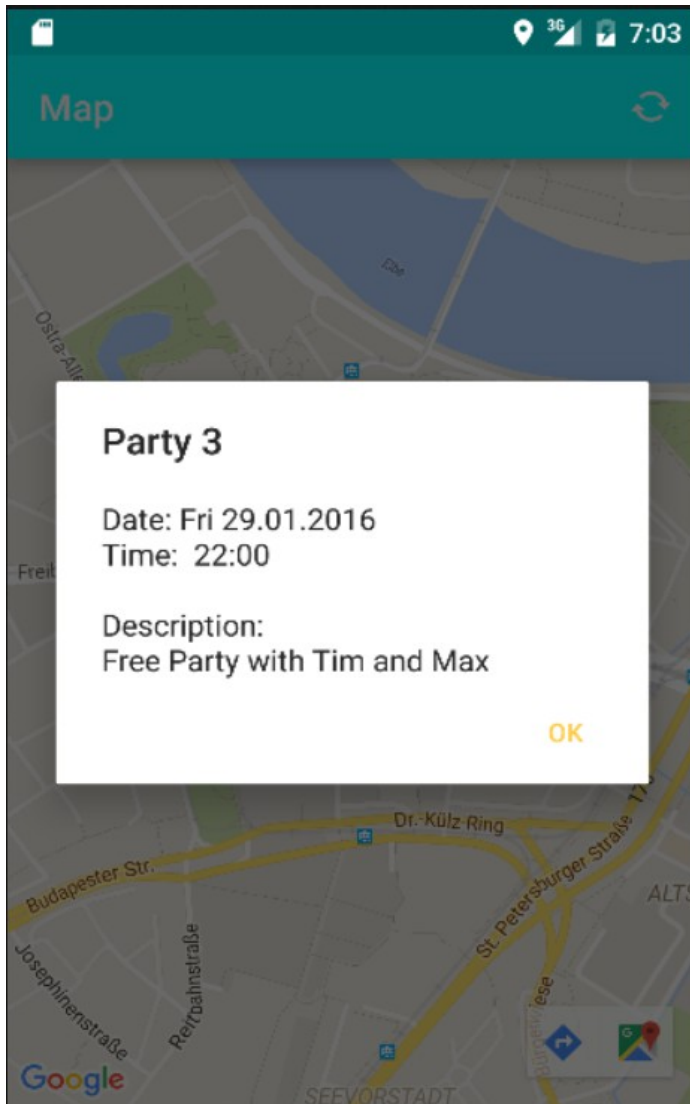
Pa  
s  
Ja  
Fr

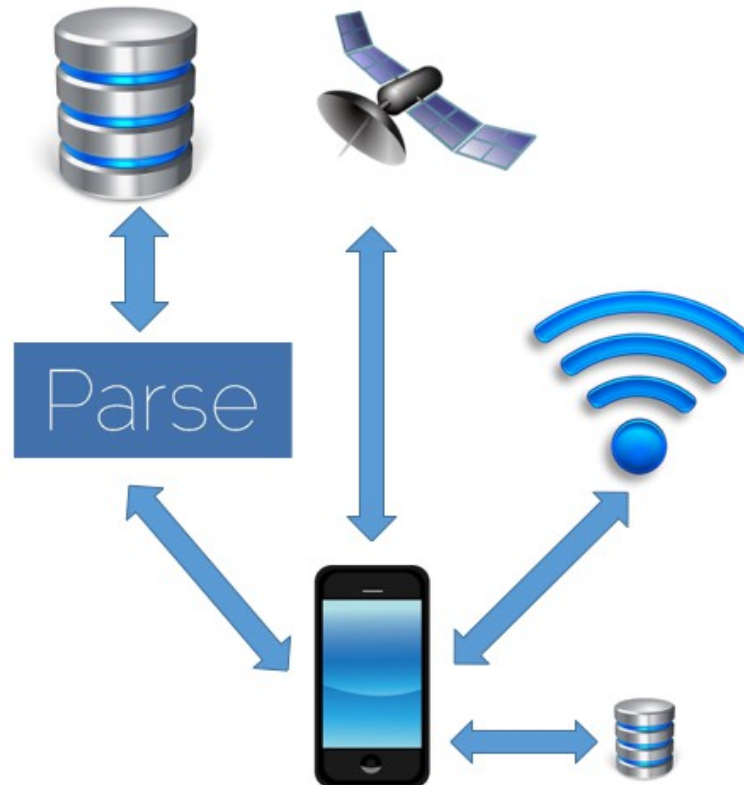
**22:00**

55 00 05  
50 10  
45 15  
40 20  
35 25  
30

**CANCEL OK**







- Energy Challenge
  - Battery Manager class used to capture the battery level
  - make a state variable (current level in relation to max scale) to represent battery status
  - scale <15%(0,15) increases the intervall between the gps updates

```
public float getBatteryLevel(Context context)
{
    Intent batteryIntent = context.registerReceiver(null, new IntentFilter(Intent.ACTION_BATTERY_CHANGED));
    float level = batteryIntent.getIntExtra(BatteryManager.EXTRA_LEVEL, -1);
    float scale = batteryIntent.getIntExtra(BatteryManager.EXTRA_SCALE, -1);

    state= level/scale;

    return state;
}
```



```
state=getBatteryLevel(MapsActivity.this);
if(state >0.15) {
    if(stateA == 0) {
        lm.requestLocationUpdates(GPS_PROVIDER, 5000, 0, ll);
        stateA = 1;
    }
}
else{
    if(stateA == 1){
        lm.requestLocationUpdates(GPS_PROVIDER, 40000, 0, ll);
        stateA = 0;
    }
}
```

```
state=getBatteryLevel(MapsActivity.this);
if(state >0.15) {
    lm.requestLocationUpdates(GPS_PROVIDER, 5000, 0, ll);
    stateA=1;
}
else{
    lm.requestLocationUpdates(GPS_PROVIDER, 40000, 0, ll);
    stateA=0;
}
```

- Connectivity Challenge
  - Connectivity Manager class used to get answer about network connectivity
  - Network Info saved in NetworkInfo variable
  - via if clause and return value range of search radius determined
  - distinguished between wifi and all mobile data connections

```
public static int state (Context context)
{
    NetworkInfo info = myCheck.getNetworkInfo(context);
    if(info != null && info.isConnected())
    {
        if(info.getType() == ConnectivityManager.TYPE_WIFI) return 1;
        if(info.getType() == ConnectivityManager.TYPE_MOBILE) return 2;
    }
}
```

```
if(myCheck.state(MapsActivity.this) != 1) {
    query.whereGreaterThan("lat", lat1 - 0.005);
    query.whereLessThan("lat", lat1 + 0.005);
    query.whereGreaterThan("long", long1 - 0.01);
    query.whereLessThan("long", long1 + 0.01);
}
else{
    query.whereGreaterThan("lat", lat1 - 0.02);
    query.whereLessThan("lat", lat1 + 0.02);
    query.whereGreaterThan("long", long1 - 0.04);
    query.whereLessThan("long", long1 + 0.04);
}
```

- Offline Challenge
  - LocalDataStore function of our backend service
  - multiple events stored in list of ParseObjects
  - ParseQuery with condition statements run in backend database and creates a list
  - list gets saved and Objects displayed on the map
  - in an offline state we run the query in the local data store

```
ParseQuery<ParseObject> query = ParseQuery.getQuery("Events");
```

```
query.whereGreaterThan("date", datefrom);  
query.whereLessThan("date", dateto);
```

```
query.fromLocalDatastore();
```

- Experiences and Pitfalls
  - Android specific functions (lifecycle etc.)
  - using backend services for easier backend management
  - GPS related features(handling location changes)
  - google maps and important cases to consider while setting it up