TECHNISCHE UNIVERSITÄT DRESDEN

DRESDEN concept

APPLICATION DEVELOPMENT FOR MOBILE AND UBIQUITOUS COMPUTING

Final Presentation

CheckIt

Group 14:

João Rosário

Tiago Caldinhas

Application Scenario

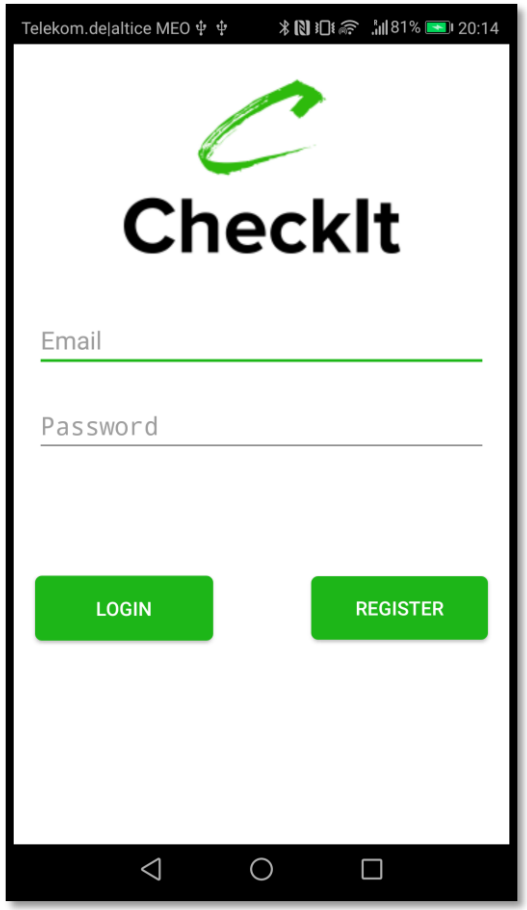
Idea:

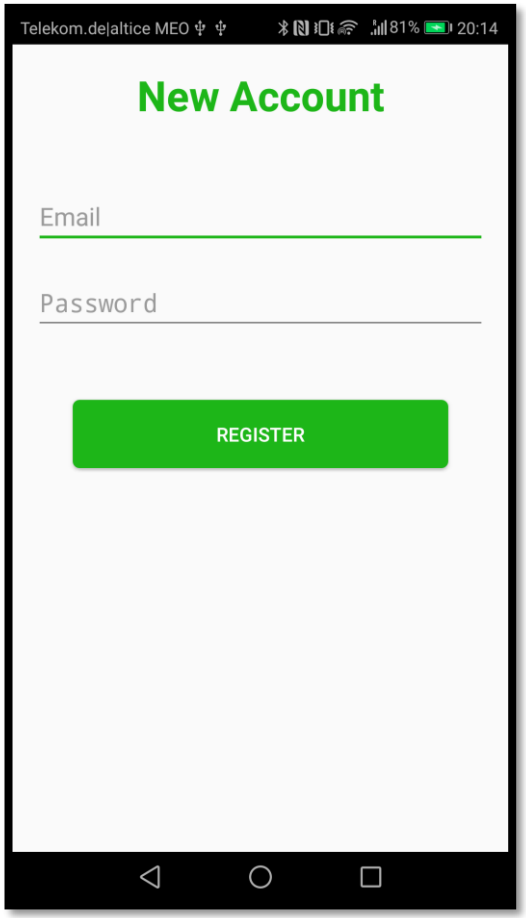
- See all interesting places around the device, shown in a list generated by our app;
- Go to one of those places;
- Get prompt with a push notification when you arrive;
- Take a picture at the place and **CHECK IT**.

Basically, we give a tourist a **simple** and **interactive** way to know the **most interesting places** around himself.

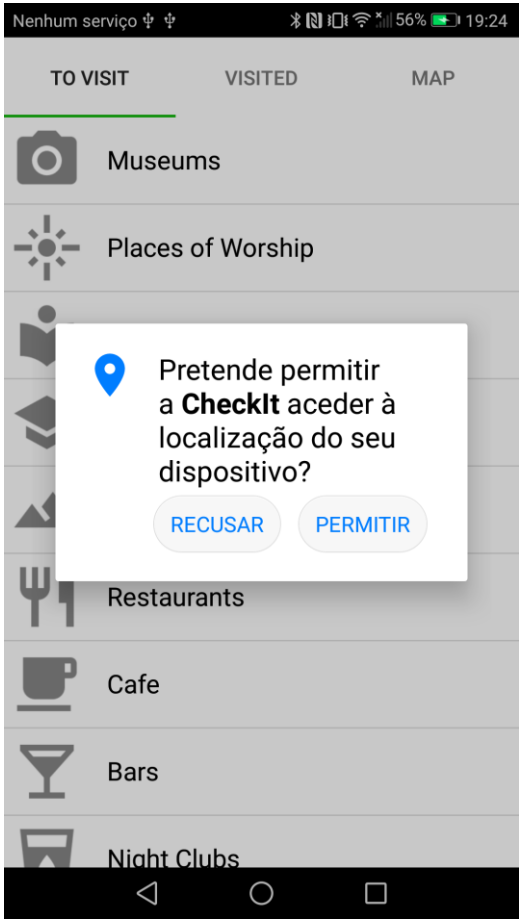
The best application for small travels to **unknown** places!

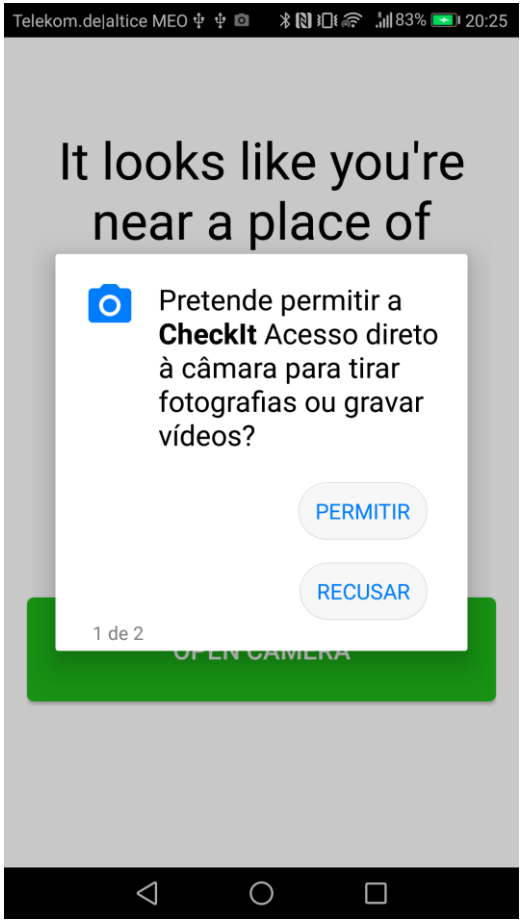
Screenshots – Login and Register

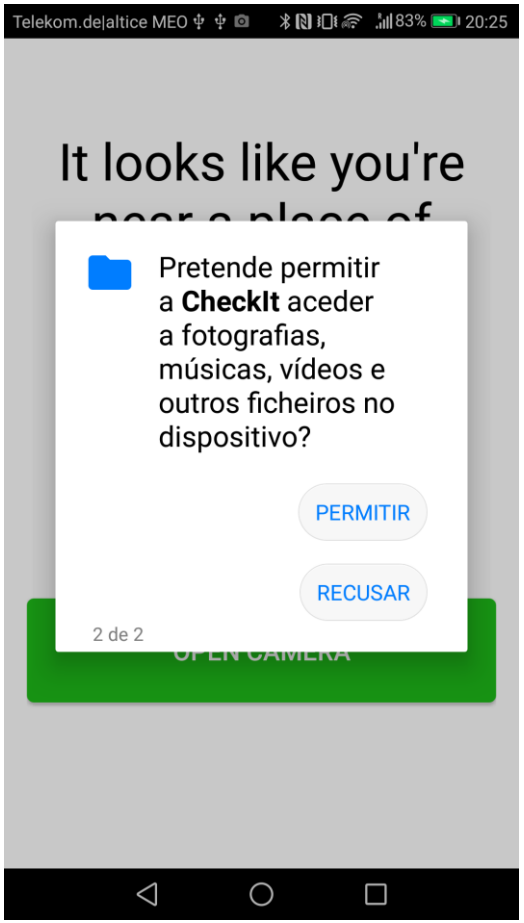




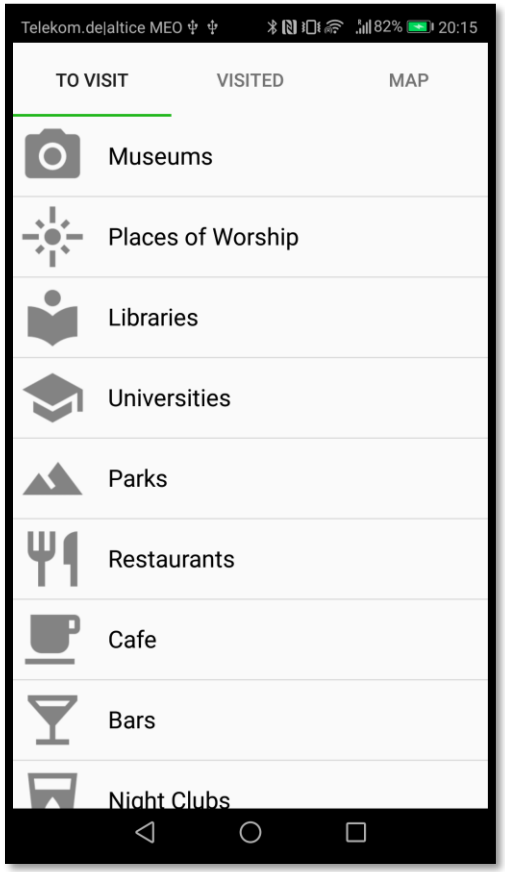
Screenshots – Permissions

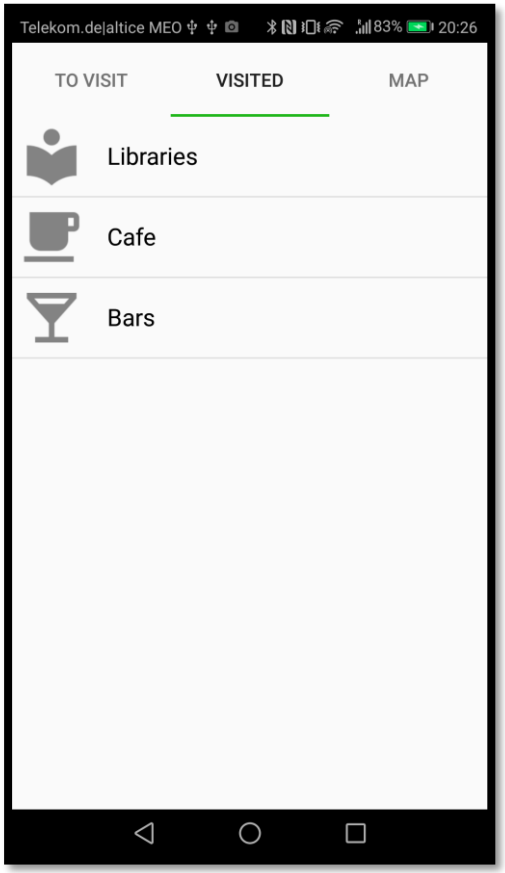


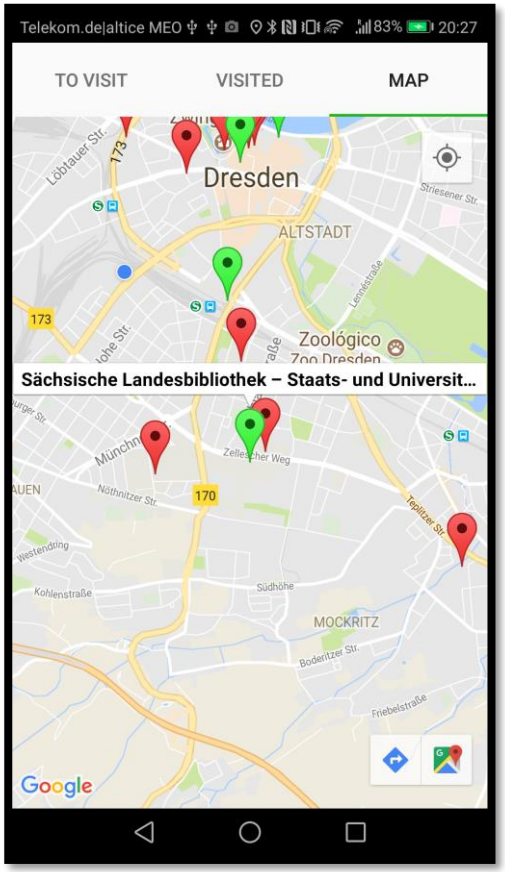




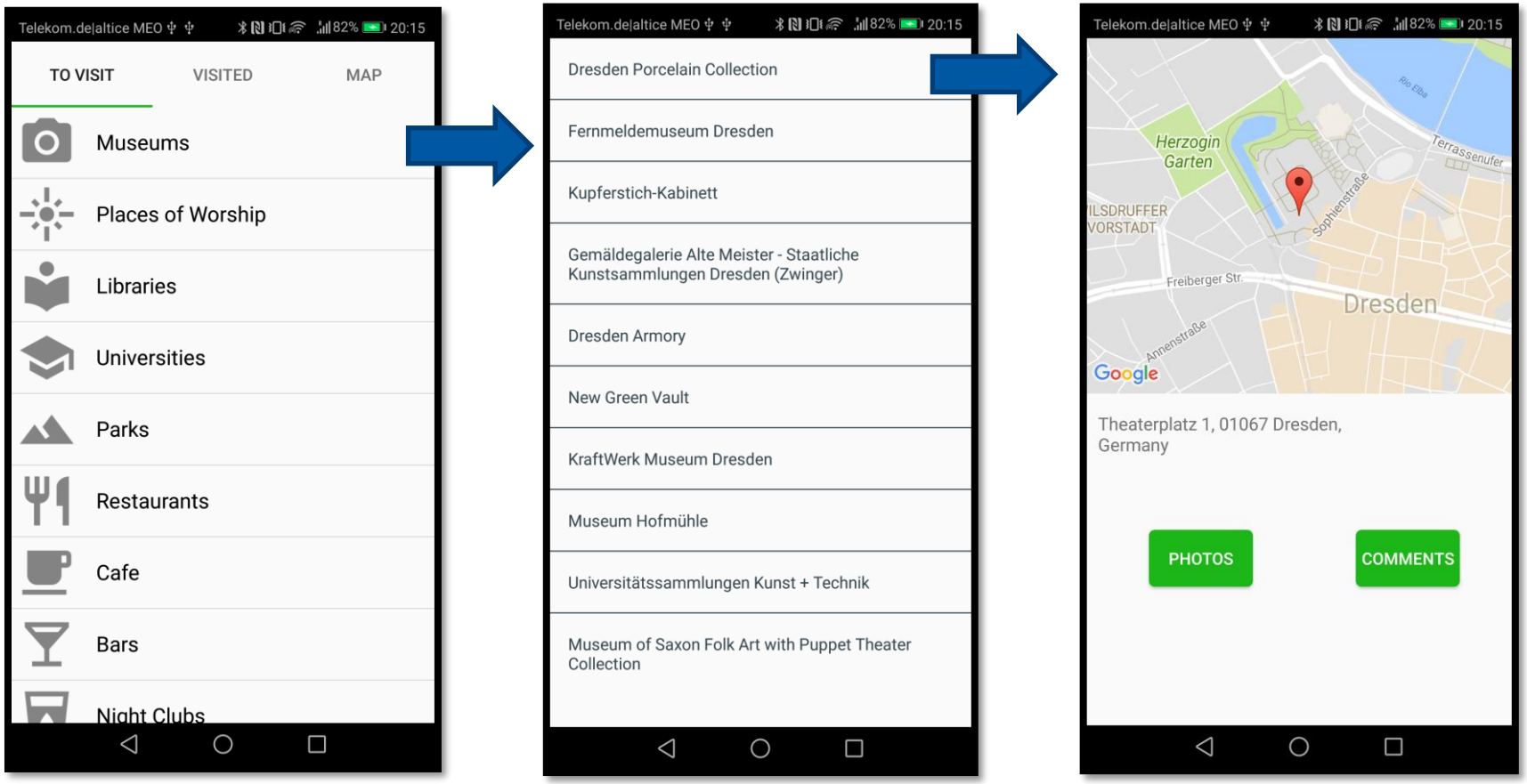
Screenshots – Main Page: TabsActivity



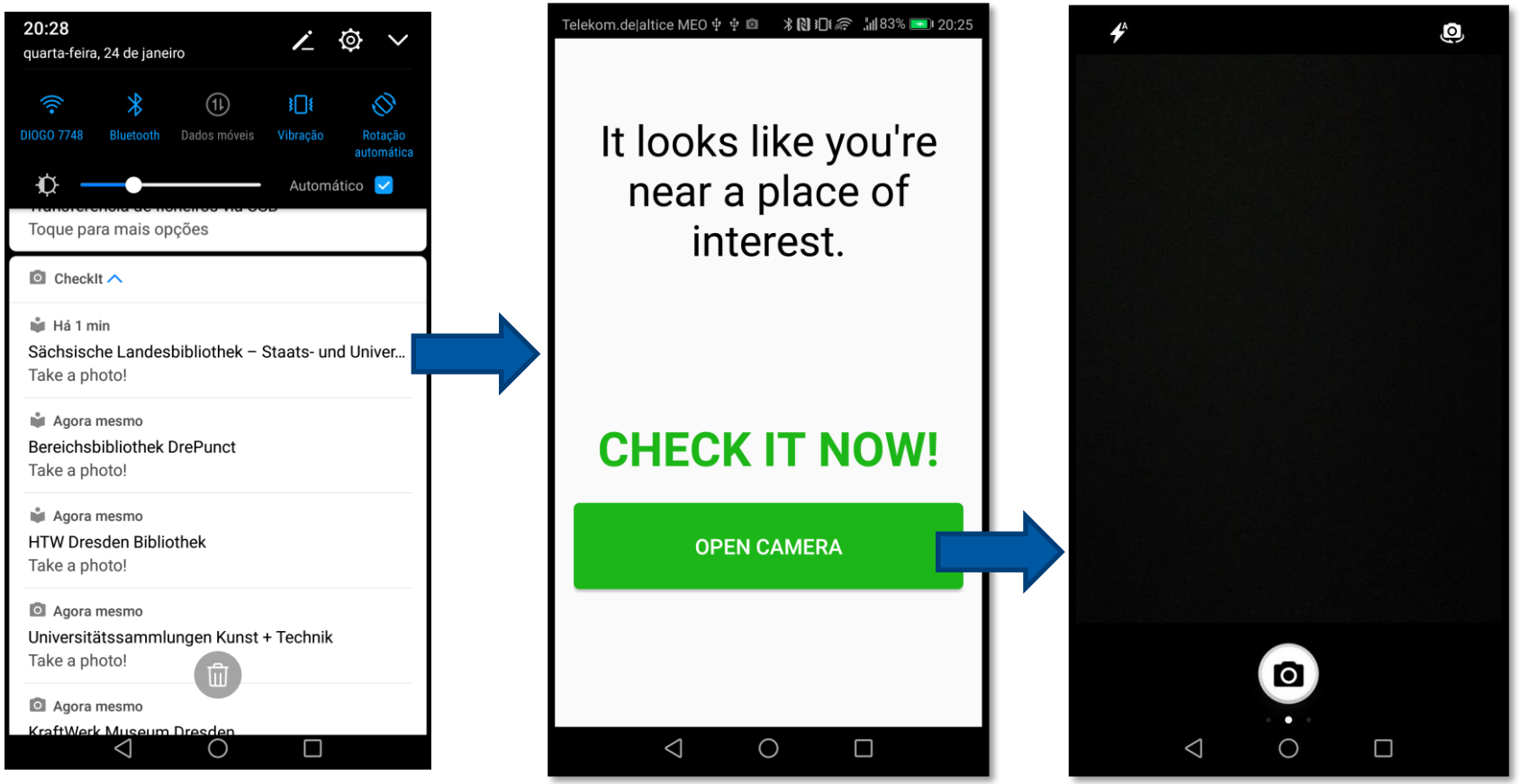




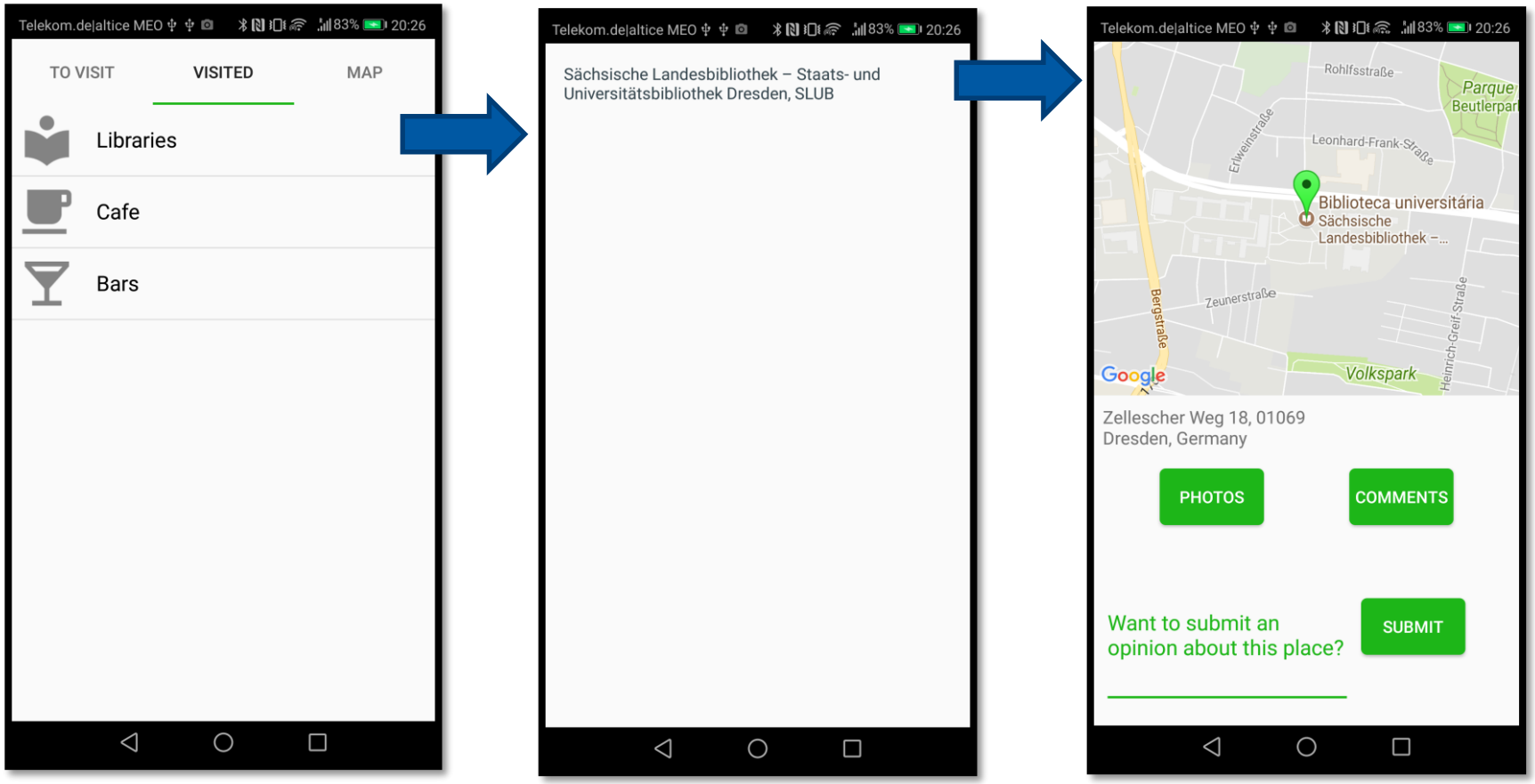
Screenshots – To Visit



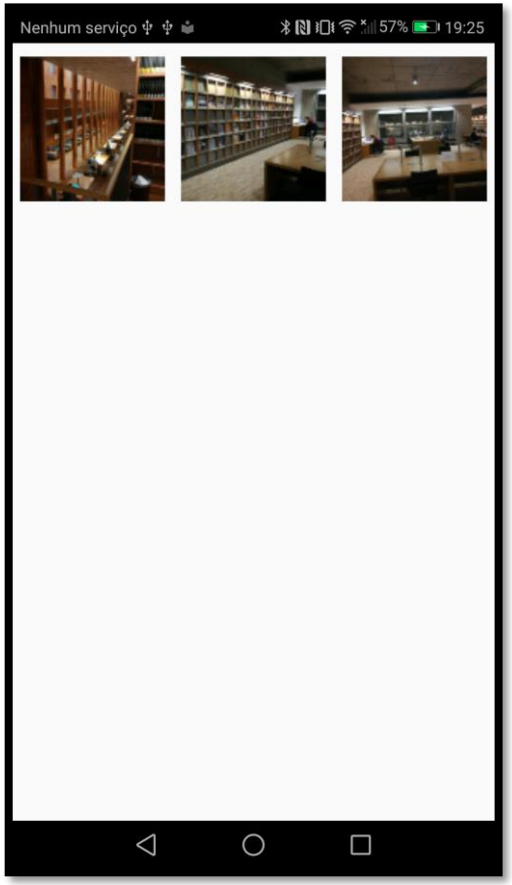
Screenshots – Notification & Camera

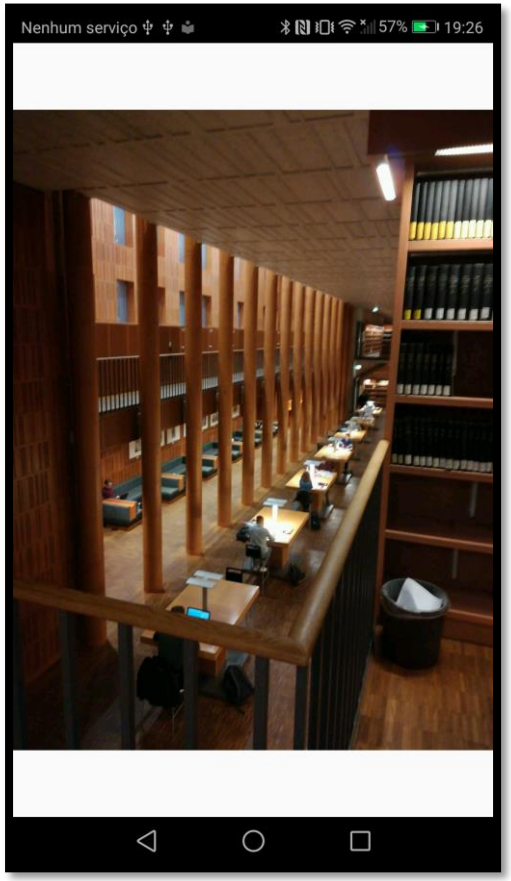


Screenshots – Visited

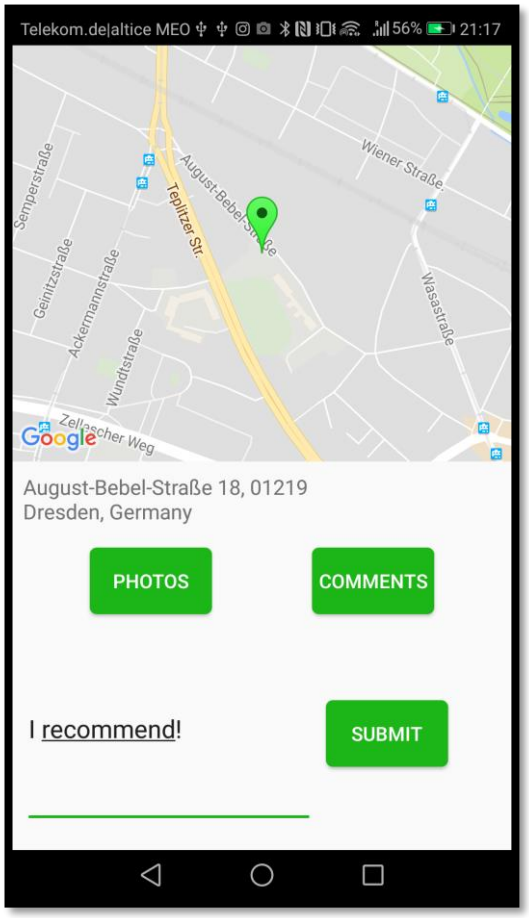


Screenshots – Photos





Screenshots – Comments



Adaptation and Context

Technical context: **Offline Usage**

- Capture if the device has a network connection, using android.net.ConnectivityManager and android.net.NetworkInfo.

Adaptation:

- **Adapt the loading of the results by choosing if they should be loaded from the server or from a file in cache:** if the device has a network connection, uses Google Services to obtain updated information, otherwise uses the results of the last search (cached).
- **Every time we get updated information from the server, this one is stored in cache.**

```
private boolean isNetworkAvailable(){
    ConnectivityManager connectivityManager =
                        (ConnectivityManager)getSystemService(Context.CONNECTIVITY_SERVICE);
    NetworkInfo activeNetworkInfo = connectivityManager.getActiveNetworkInfo();
    return (activeNetworkInfo != null && activeNetworkInfo.isConnected());
}
```

Adaptation and Context

Technical context: **Network Awareness**

- Detect type and quality of network connection using android.net.**NetworkInfo**, android.net.wifi.**WifiInfo** and android.telephony.**TelephonyManager**.

Adaptation:

- **Adapt the amount of data transferred:** depending on Wifi's connection speed or the type of mobile data connection, fetch more or less photos of a specific place (from the server).

```
if (activeNetwork.getType() == ConnectivityManager.TYPE_WIFI) {
            Checks Wifi connection speed
} else if (activeNetwork.getType() == ConnectivityManager.TYPE_MOBILE) {
            Checks type of mobile data connection
}
```

Adaptation and Context

Physical context: **Current location**

- Capture the device's location, using android.location.LocationManager.

Adaptation:

- **Adapt the results of the search for interesting places:** Using the current location (latitude and longitude) we use the Text Search Request from Google Places API to get a set of locations.
- **The results should be inside of a given radius, centered at the device's location:** The radius starts with a default value of 1000m, but if the search doesn't return enough results the radius increases gradually (to a maximum of 10Km).

Example of a request:

https://maps.googleapis.com/maps/api/place/textsearch/json?query=restaurant&location=51.0429730, 13.7223350&radius=**X**&type=point_of_interest&key=OUR_KEY

Latitude found with the **LocationManager**

Longitude found with the **LocationManager**

Adaptation and Context

Physical context: **Current location**

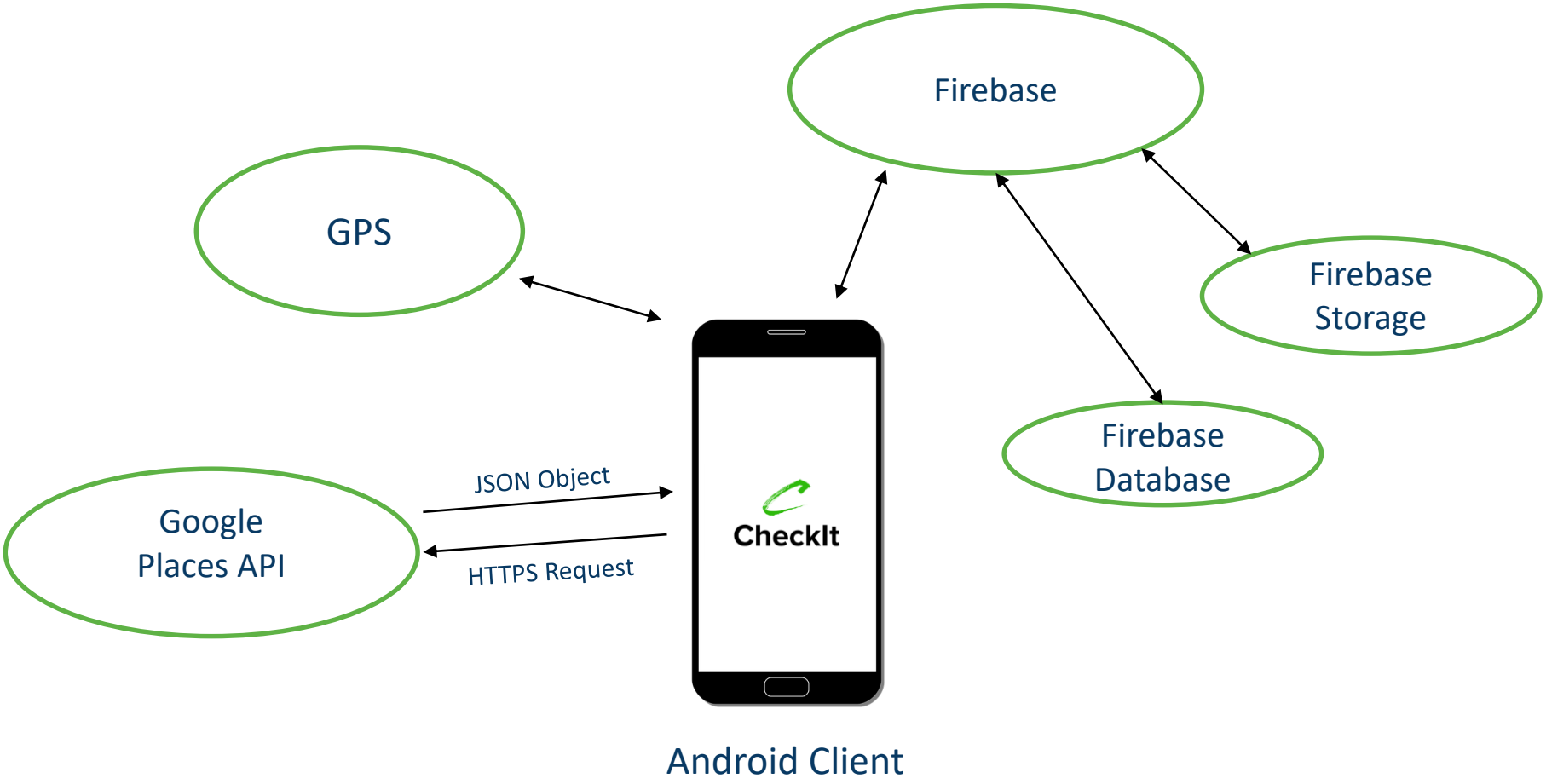
- Capture the device's location, using android.location.LocationManager.

Adaptation:

- **Send push-notification to notify the user about being near a place of interest:** When the device's location is close enough to one of the places to visit (inside a radius of ≈100m, centered in the place's location), prompt the user, so that he can take a photo and mark the place as "Visited".

```
if ((deviceLocation.getLatitude() >= placeToVisit.getPlaceLat() - 0.001)                    //-100m
            && (deviceLocation.getLatitude() <= placeToVisit.get(i).getPlaceLat() + 0.001)  //+100m
            && (deviceLocation.getLongitude() >= placeToVisit.get(i).getPlaceLng() - 0.001) //-100m
            && (deviceLocation.getLongitude() <= placeToVisit.get(i).getPlaceLng() + 0.001)) //+100m
            {
            Prompt  the user about being near a place to visit.
}
```

Architecture



Technologies

Client:

- Android
- GPS for location tracking
- Mobile Camera
- Google Maps API
- Google Places API

Server:

- Firebase

Firebase

Lessons Learned

- How to use AndroidStudio to develop an Adroid App.
- How to use Google Firebase.
- Good work plan = smooth execution (without major delays)
- Perform more tests in order to cover all the bugs (specifically in the push notification system)

Next Steps

- Implement a like/dislike system to photos and comments.
- Improve our app's power consumption (even more).
- Implement a gamification system, that would "reward" the user for visiting the places of interest.
- Give our app a little of a "Social Network" touch:
  - User profile;
  - Ability to share the places you visit with other users;
  - Ability to follow someone.
- …

There's always room to improve!