Log File

Week 1

Work Package: Initiate Android App

- update Android Studio and gradle plugin
- download and examine yandex maps demo app:
    - https://github.com/yandex/mapkit-android-demo
- get yandex maps api key from yandex developer dashboard
    - https://developer.tech.yandex.ru/services/
- start empty android project
- Old version with yandex maps:
    - https://github.com/DorukTaneli/navigation-assistant-yandex
- New version with google maps:
    - https://github.com/DorukTaneli/navigation-assistant

Week 2

Work Package: Integrate Yandex Maps

- new Branch: yandex-maps
- add maven to project
- install Yandex MapKit via gradle dependencies
- Add bare yandex map and set camera to Istanbul
- Adjust Layout to show Route Information
- Merge to master

Other:

- research about Android themes
- change theme to DayNight.NoActionBar
    - to make the app easier on eyes and to have more available screen space
- research about google maps api

New Work Package: Switch to Google Maps

While building the app I noticed that the documentation for Yandex MapKit API is incomplete, so it is not possible to do this project using yandex maps API. yandex Mapkit documentation So we decided to switch to google maps, thinking the free monthly 200$ would be enough for our use.

- get API key from Google Cloud Platform
- restrict key so we do not go over the free monthly 200$
- create Android Project with google maps, push to github
- add camera animation to istanbul
- create billing profile in order to use maps api
    - create a sanal kart from yapikredi with 10 try limit so I do not get billed a huge amount if we do something wrong
- change api key to include directions api and places api
- start a youtube tutorial about google maps android api
    - https://www.youtube.com/watch?v=RQxY7rrZATU&list=PLgCYzUzKIBE-SZUrVOsbYMzH7tPigT3gi

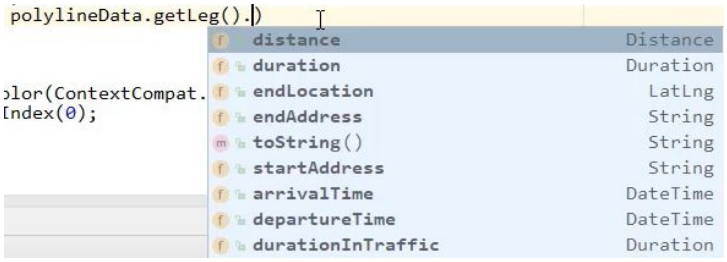
Week 3

Work Package: Preprocess Data from Google Maps

- read google maps API documentation
- research about how can we get what kind of data from the api
- The API gives us the directions as a set of points like in the image below. These points can be drawn on the map, however it is surprisingly hard to figure out which paid roads the route is passing from.

```
"overview_polyline" : {
   "points" : "knjmEnjunUbKCfEA?_@]@kMBeE@qIIoF@wH@eFFk@WOUI_@?u@j@k@`@EXLTZHh@Y`AgApAaCrCUd@cDp
   {MdZwAbDaKbUiB|CgCnDkDbEiE|FqBlDsLdXqQra@kX|m@aF|KcHtLm@pAaE~JcTxh@w\\`v@gQv`@}F`MqK`PeGzIyGf
   cIHkDXuDn@mCt@mE`BsH|CyAp@}AdAaAtAy@lBg@pCa@jE]fEcBhRq@pJKlCk@hLFrB@lD_@xCeA`DoBxDaHvM_FzImDz
},
```

- So instead of calculating cost information of route options, we decided to switch to the other example use case of the navigation assistant map, minimizing the time spent in traffic, a feature that will greatly help drivers using stick shift.
- We can easily get the durationInTraffic variable from the polyline that is returned from the API. The variable is in the very bottom of the image below.



- So I updated my work packages according to this new use case. The work packages that have changed since the project proposal start with an asterisk(*). The last week is still empty incase we encounter a problem and cannot finish the project on time.

Work Package	Doruk	1	2	3	4	5	6	7	8
Initiate Android App	R								
Integrate Yandex Maps	R								
*Switch to Google Maps	R								
Preprocess Data from Google maps	R								
*Display route options and durationInTraffic	R								
*Connect voice recognition to durationInTraffic feature	R								
*Add text to speech, read aloud route options	R								

- Write use case scenario for duration in traffic feature:

Precondition: User wants to plan an enjoyable drive with least traffic, even if it takes longer than the shortest route.

1. User prompts the search for the shortest duration-in-traffic route by touching the floating button or voice.
    a. Voice commands to prompt search with where he would like to go like: “Time to go to work”, “Let’s go home” or straight up “home”.
    b. Click to a place on the map and press ‘show routes’.
2. The system finds the route options and displays the routes as a polyline on the map, along with durationInTraffic, and total duration for every possible route.
3. The compares the route options out loud.
    a. If the shortest route and the least duration in traffic is the same route: System says: “The route with least traffic is also the shortest route”. Then highlights that route.
    b. If there is a shorter route than the least duration in traffic route: The system highlights the least duration in traffic route and says: “This is the route with the least traffic, it takes x minutes longer than the shortest route.”
4. The user checks the map, decides which route they prefer and drives to their destination accordingly.

Week 4

Display Route Options and DurationInTraffic - first week

- Add a draggable marker on the touched location. To be used for specifying the route destination.
- Clone and examine the official Google Maps tutorial CurrentPlaceDetailsOnMap.
    - https://github.com/googlemaps/android-samples/tree/master/tutorials/CurrentPlaceDetailsOnMap
- Get another API key to run CurrentPlaceDetailsOnMap, as the first key was restricted only for the first project.
- Ask for location permission on the first run.
- Get current location using FusedLocationProvider.
- Show current location on map with the default blue indicator.
- Add show my location button to the top left corner of the UI.
- Animate camera to current location on start.
- Add Istanbul as the default location. Camera will move to Istanbul if getting location fails.

Other

- Add README file to the repo that explains how to recreate the project.

Week 5

Display Route Options and DurationInTraffic - second week

- Update gradle and android studio
- Add gmaps Directions Api dependencies
- Add compatibility compile options to solve build error
- Initialize GeoApiContext
- Create CalculateDirections function. The function creates a DirectionsApiRequest using the GeoApiContext, using the users location as origin and the marker location as destination. Then, it prints the result in log.
- Configure restrictions on Api key to allow directions api
- Add InfoWindowListener so that you can click on the marker's InfoWindow to call CalculateDirections function
- Directions api works by server requests, so restricting the key to my android project blocks it from working. I made the following changes to let the api work and hide my api keys from public
    - Add google_maps_api.xml files to .gitignore

    - Regenerate api key
    - Change api key on the project
    - Update README file on how to recreate project
- Add addPolylinesToMap function which draws the routes to map.
- Add blue and grey colors to resources
- Create a new data structure in models package called PolylineData, to keep track Polyline and DirectionsLeg of each route
- Add onPolylineClick function that makes the selected route blue and brings it forward, while making the other routes grey and sends them backwards.
- Show inTraffic and Total duration of different routes when clicking on the route.
- Highlight the route with shortest durationInTraffic initially instead of a random route.
- Animate the camera to the shortest durationInTraffic route using zoomRoute function.
- Add reset map button on top right that clears map and routes data structure contents.
- Adjust reset button to match native google Maps buttons

Week 6

Work Package: Voice Recognition

- Create a white microphone icon drawable
- Add a floating action button with microphone icon and dark blue background on the bottom right corner
- Create RecogniceSpeech function that starts the Google Speech Recognizer Intent.
- Create onActivityResult function that gets the result from the Intent.
- Change language to English instead of default so it can work as intended on Turkish devices.
- Relocate the floating button so other controls are not blocked

Other:

- Write Proposal Revision after separating projects with Aslı
- Create a video for mid-demo

Week 7

Work Package: Text to Speech

- Instantiate Text to Speech
- Test US and UK English text to speech versions, US sounds better.

- Set language to US English
- OnPause: stop the speak and release the resources used by the TextToSpeech engine.
- Start text to speech when showing the best route
- Stop releasing the resources used by the TextToSpeech engine onPause as it causes a bug.
- Make speech 30% faster than the original
- Shutdown TextToSpeech onDestroy to prevent memory leak
- Pick the speech randomly for a more natural feeling

Other:

- Add comments to make code easier to understand

Week 8

Work Package: IUI Improvements - Voice commands

- Hard code home and work addresses
- Clear map after eligible voice commands
- Fix home LatLng
- Refactor AddMarker code for reuse
- Calculate directions to home/work on voice command

Places Search

- Add google maps places api dependency to project
- Add places functionality to api key
- Add search fragment to layout
- Move reset and myLocation buttons
- Initialize Places API
- Initialize Search fragment and connect it to Places API
- Limit search to establishments in Turkey
- Add location bias to search
- add AddPlaceMarker function and call it on search box: onPlaceSelected
- fix marker infoWindow bug
- get places programmatically for voice connection
- add market, gas station, and restaurant use cases

Other:

- add more speech options
- remove unused code
- fix in traffic duration being more than total duration
- fix marker title bug
- disable rotate gestures because compass is blocked by new search box
- Text to speech also says in-traffic and total duration

Last Week

Adding User Scenarios

- Research about chatbot APIs.
- Add Dialogflow to project from google cloud console.
- Dialogflow API v1 was deprecated

https://github.com/dialogflow/dialogflow-android-client

- And v2 client library does not support android

https://cloud.google.com/dialogflow/docs/reference/libraries/java

- So I decided to implement the scenarios manually myself.
- add eczane, hastane, manav, fırın, cafe scenarios
- Call emergency services in hospital scenario
- Modify gas station scenario:

Adapting to User Gas Station Preferences

- People generally prefer to buy gas from a specific brand of gas station
- On the first use, starting gas station search will return all gas stations
- When the user picks a specific gas station brand over and over, starting gas station searches will show that brand of gas stations.

- The user will still be able to see all nearby gas stations. If the user searches for all gas stations, the preference is reset and regular searches will show all gas stations until a preference is observed again.
- Make the number of gas station brand selection to set up preferences 1 for the demo.

Suggest Multiple options instead of a single one

- Change AddPlaceMarker function to allow showing multiple places on map
- Change LocationRequest function to add markers on suggested locations
- Fix infowindow bug when showing multiple markers, you now need to tap on markers to see information.
- Create StartPlacesSpeech function. Will be called while showing places.

Other

- Update text to speech sentences
- Fix showing old routes instead of new ones bug
- create .xml dictionary to remove typo warnings
- change work and home addresses plus boundaries to Istanbul from Izmir
- fix gas station case sensitivity problem
- fix app crash when calculating directions
- fix gas stations not showing up as intended
- viewing all gas stations instead of only preferred brand now works
- resolve ArrayList warnings
- Comment out duration in text to speech for demo as it takes too long