Formativ Utvärdering

Jag har valt en uppgift av NavigableMe applikationen som en användare skall genomföra. Meningen med detta är att jag själv som utvecklare ska kunna få under rådande applikationsutveckling feedback från andra användare, hur de uppfattar appen. På så vis får jag mer insikt om vart applikationen eventuellt brister och hur jag kan förbättra applikationens brister. Detta för att anpassa applikationen utifrån användarens perspektiv.

Genomförandet av standardfunktionen omfattar följande steg:

Steg	Genomförande
1.	Hitta kartsidan och tryck på den.
2.	Vid kartsidan tryck på sökfältet.
3.	Skriv in önskad destination.
4.	Tryck på enter efter inskriven destination.
5.	Se punkt av den inskrivna destinationen på kartan.

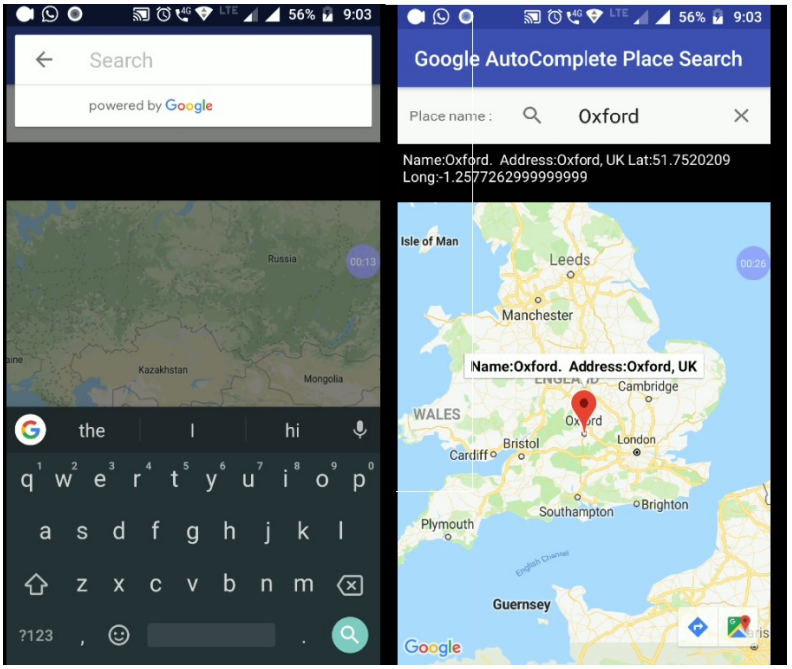


fig 1 - Kartexempel.

Min vän som användare tillämpade dessa steg men förstod inte helt riktigt applikationens syfte. Användaren menade på att den saknade standardfunktionaliteten saknade en viss typ av funktionalitet och betydelse. Användaren ville ha mer funktionalitet. Varför? Jo eftersom en koordinat för ett navigeringssystem säger inte så mycket. Användaren ville alltså ha ett visuellt förslag hur hen tar sig från sin nuvarande plats till den destination hen angivit.
Efter ha fått feedback från användaren fick jag insikten om att applikationens ursprungliga syfte inte hade manifesterats. Applikationen hade funktion av att söka efter en punkt i

MapFragment. Text som användare matar in hanteras och omvandlas till en Sträng. Strängen används av en instance av Geocoder. Geocoding är processen av att transformera en gatuadress *eller* andra beskrivningar av en plats till en latitud och longitud koordinat. Genom att använda GoogleMap klassen kunde min geocoder objekt bidra med koordinaterna och sätta ut ett Marker objekt på kartan.
Detta gav endast en punkt på kartan men inte hur man kan ta sig dit. Syfte hade således inte uppnåtts på något vis utifrån tidigare dokumentation. Detta korrigerades efter feedbacken.

Skriv en beskrivande och argumenterande utvärdering (*max två (2) A4-sidor med bilder*) för din app och ange fördelar/nackdelar med att göra denna typ av utvärdering av din app. Detta dokument ska med vid inlämning!

Fördelar:
Applikationens nuvarande funktionalitet för navigering är baserad på google maps applikationen.

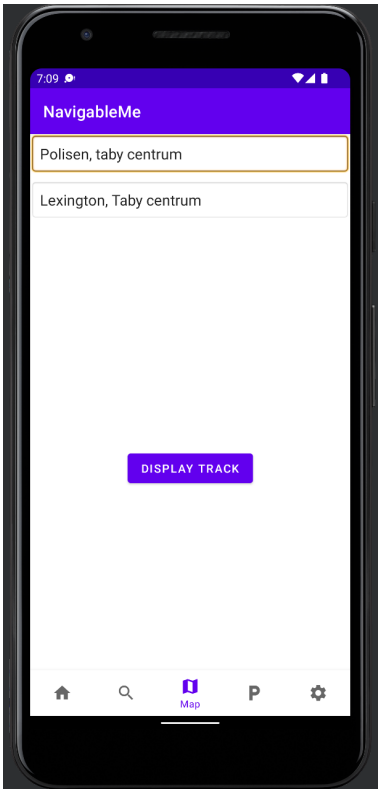


Fig 2
Navigationen för användaren är baserad på GoogleMaps. Vid MapFragment anger användaren sin **från**-punkt och sin **till**-punkt (Punkt **A** & **B**). När användaren trycker på *"Display Track"* knappen får användaren ett förslag på hur man tar sig från punkt A till B. Se resultat Fig 3.

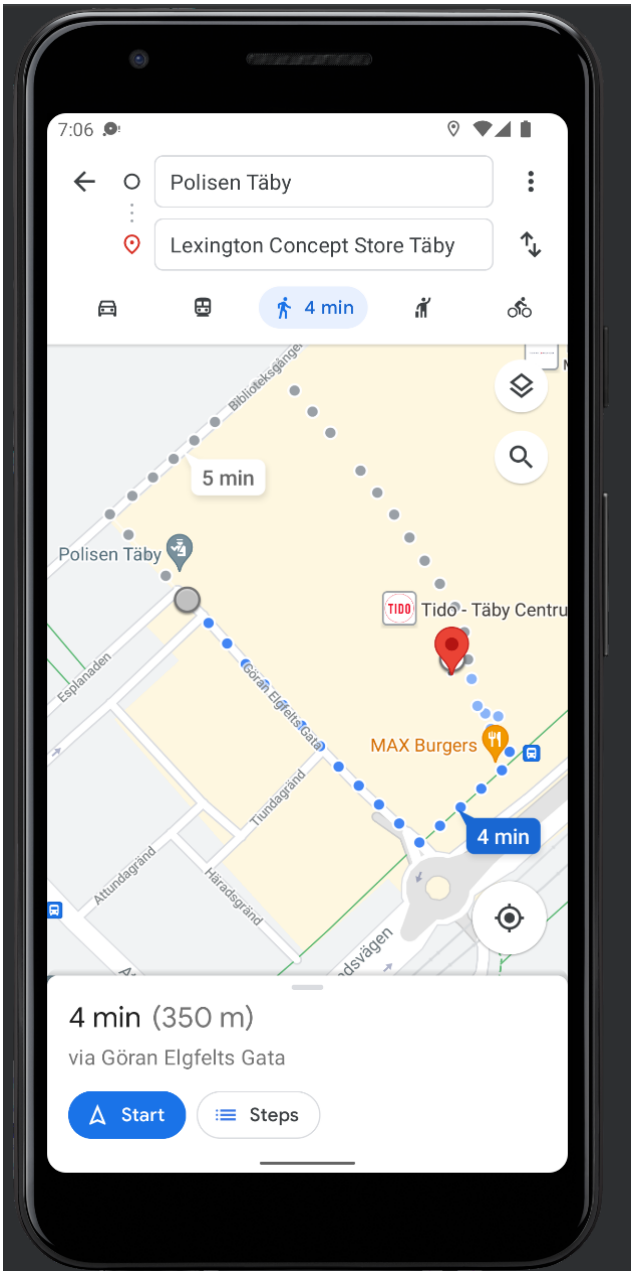
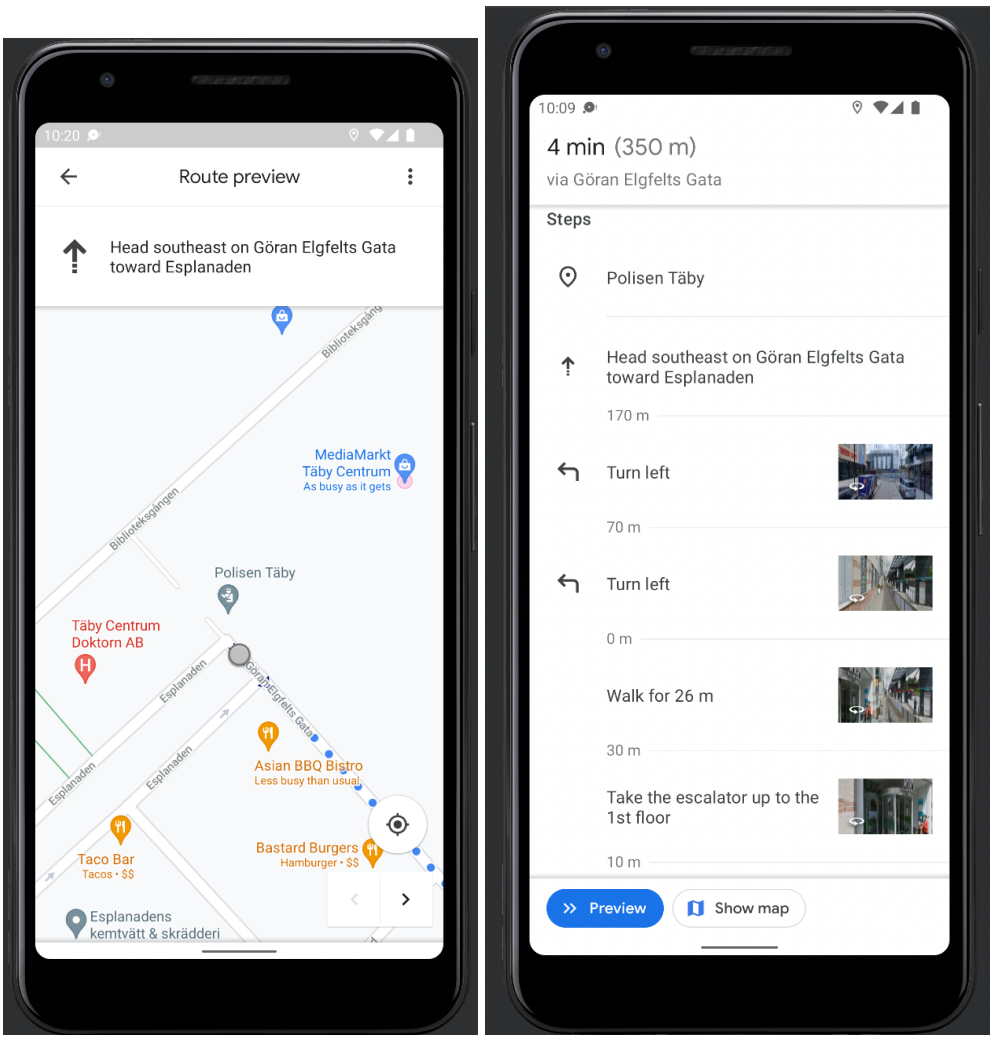


Fig 3

Det kan vara svårt att tyda navigation i en stor byggnad som Täby Centrum. Därför har google maps funktionalitet av att dela in navigationen i steg. Trycker du på “Start” så får du upp det första steg förslaget för att komma till din destination (se fig 4 - Route Preview). Du kan även trycka på “Steps” för att få upp en vy på alla steg (se fig 5 - Steps).



(fig 4 - Route Preview), (fig 5 - Steps)

Fördelen med denna implementation i appen är att användaren kan skriva vilka platser i världen som hen vill och få ett förslag på hur hen tar sig dit (förutsatt att användaren har internet).

En annan fördelen med programmet är att i ParkFragment tar emot användarens registreringsnummer och anger en plats för användaren att parkera.
  Dessa platser är dock fiktiva då de är genererade från ett Random objekt. När användaren trycker på “Park” knappen sparas information av användarens registreringsnummer och plats i samma fragment. Trycket på “Park” knappen startar även en CountDownTimer med ett hårdkodat värde på tio minuter. Om användaren väljer att byta aktivitet eller stänger av applikationen ligger informationen fortfarande kvar i samma fält. CountDownTimern fungerar som tänkt mellan växling av fragmenten. Däremot om appen stängs av återgår timer objektet till sin preliminära tillstånd. Detta är en nackdel eftersom användaren kan då inte veta hur långt tid det är kvar tills parkeringen har gått ut.

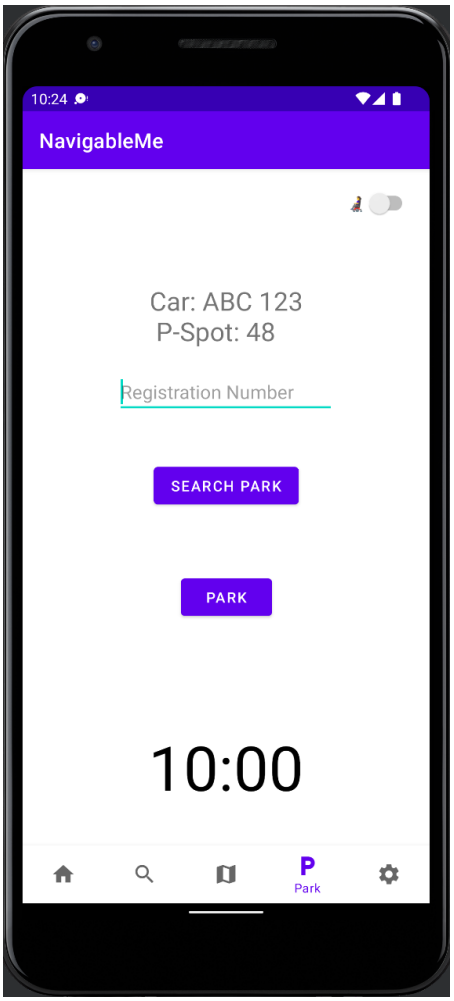


fig 6 - ParkFragment

@KOMMENTAR - I Täby Centrum har de en skärm precis vid parkeringsentrén och med hjälp av kameror fotograferas registreringsnumret och visar upp följande.

BIL 123, välkommen
2h fri parkering. Alltså behövs ingen P-skiva eller dylikt. Allt är redan registrerat med Täby Centrums system. De skulle även kunna visa information om en ledig plats i den skärmen.

Användaren kan även se hårdkodade affärer i form av en listview. Om användaren väljer att trycka på en av butikerna i listview:n så hänvisas användaren till MapFragment. När användaren kommit till MapFragment får användaren sin destination föreskriven.

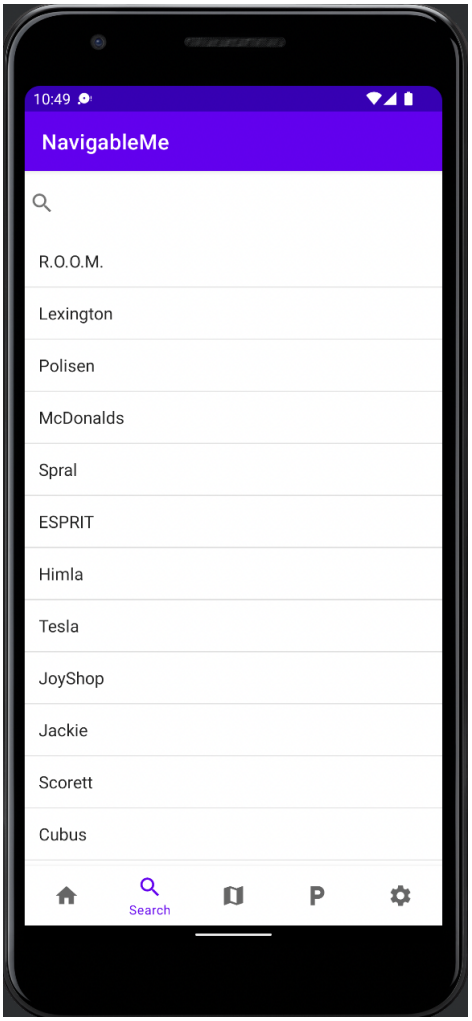


Fig 7 - SearchFragment

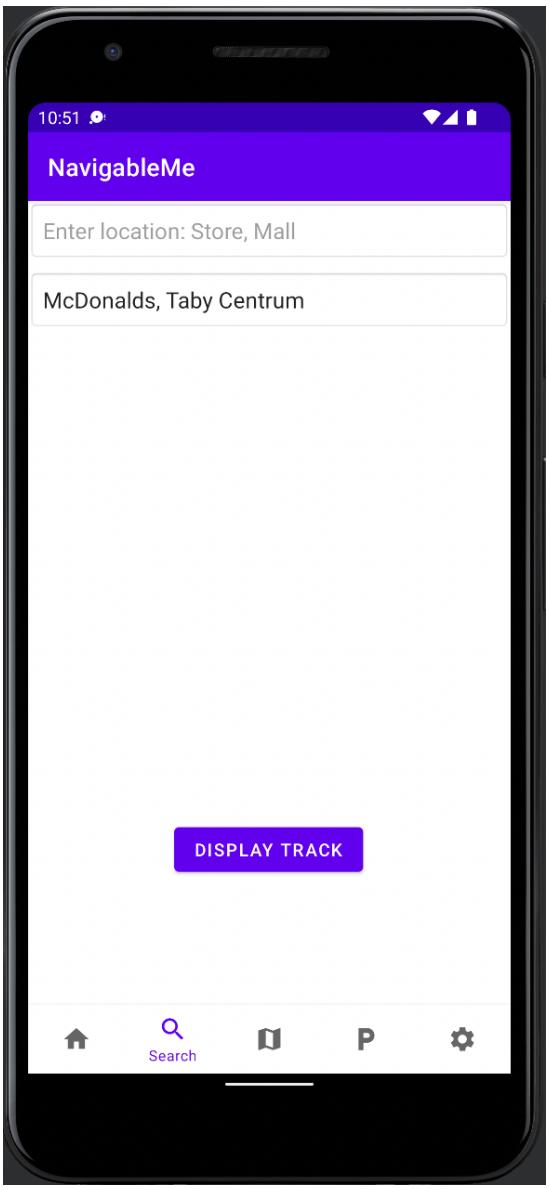


Fig 8 - Vald butik från ListView:n

Efter att användaren har fått sin butik föreskriven i edittext komponenten måste användaren skriva sin startplats vilket jag själv upplever frustrerande. Detta anses som en nackdel.

```java
@Override
public void onItemClick(AdapterView<?> adapterView, View view, int i, long l) {

    // get user's clicked item from ui list view
    String selectedItem = (String) adapterView.getItemAtPosition(i);
    // set text to clicked item
    textUpdate.setText(selectedItem);
    // Create new fragment and transaction
    MapFragment mapFragment = new MapFragment();
    Bundle args = new Bundle();
    args.putString("selectedItemKey", selectedItem);
    mapFragment.setArguments(args);

    getFragmentManager() FragmentManager
            .beginTransaction() FragmentTransaction
            .replace(R.id.container, mapFragment, tag: "TAG_MAP_FRAGMENT")
            .addToBackStack("TAG_MAP_FRAGMENT")
            .commit();

}
```

Fig 9 - Kod för vald butik.

I föregående figur 8 illustrerades en vald butik från ListView:n. Vid denna funktion uppmärksammade jag två problem. Min implementation av att byta fragment vid klick inte tillämpas som jag velat. Detta eftersom att BottomNavigationView:n indikerar att användaren fortfarande är på SearchFragmentet (se fig 8 nedre navigationsfält “Search” icon”). Detta ger inte en intuitiv känsla om vart användaren befinner sig. Detta är således en nackdel.

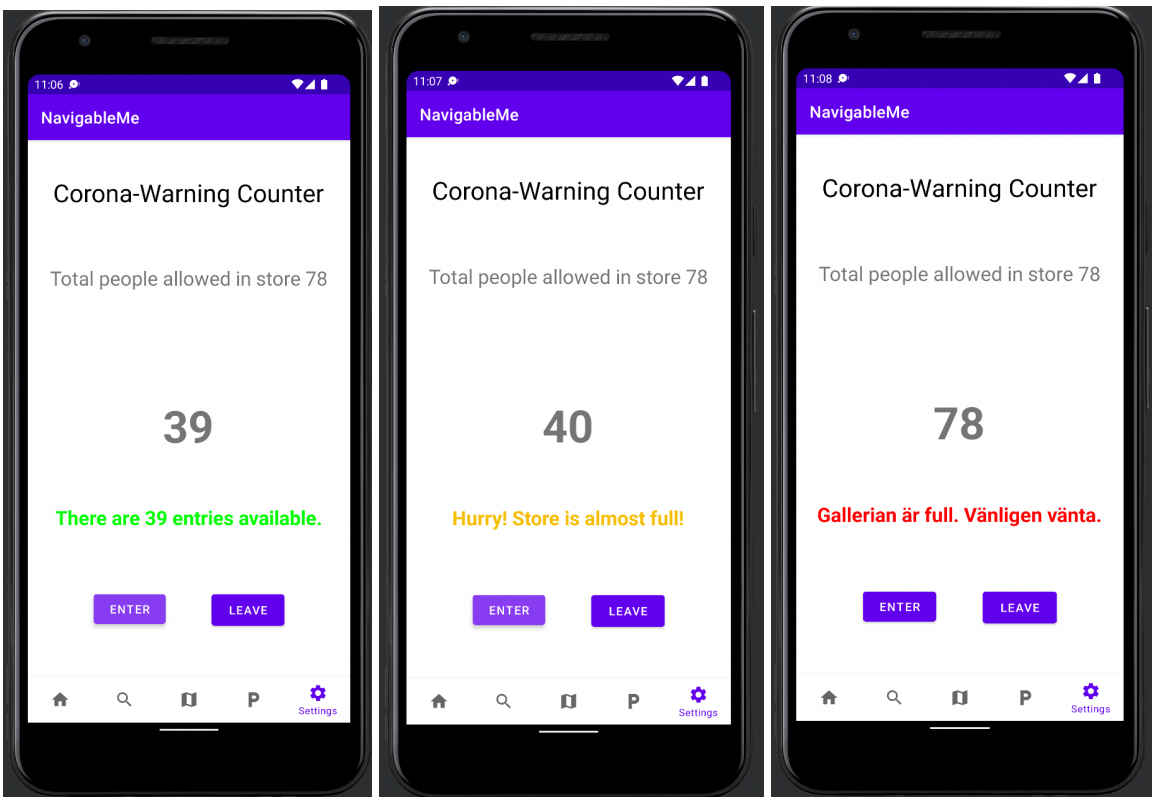


fig 10, fig 11, fig 12

En fördel är att appen implementerar logik för en fiktiv butik vad gäller antalet kunder i SettingsFragment. Den fiktiva butiken har ett maxantal. Överstigs maxantalet är inte affären “corona-vänlig”. SettingsFragment signalerar med sin logik med färger och text. Är antalet kunder i butik < hälften av maxantalet visas grön text (fig 10) är antalet > hälften av maxantalet visas gul text (fig 11), är antalet = maxantalet visas röd text (fig 12).
Något som brister är att SettingsFragment saknar implementation för att spara data om antalet kunder i butik och genererar ett random värde vid varje instans av SettingsFragment. Detta går dock att implementera med SharedPreferences som jag gjorde i ParkFragment. Detta var något jag inte hann med helt enkelt.

Jag är dock lite missnöjd med applikationen i sig. Jag anser att den inte är fullständigt och behöver mer av attribut av korrekthet och robusthet. Eftersom att allt är hårdkodad så är den inte så användbar i slutändan.

Om jag hade utvecklat appen på nytt hade jag använd mig av en loosely coupled arkitektur dvs ett system där delarna är svagt kopplade med varandra och kan därmed anpassas utifrån implementation. Jag hade även skapat Modeller för butiker och köpcentrum. På så vis skulle jag undvika det hårdkodade temat som denna applikation besitter sig.