NearBy : progetto e sviluppo di un'applicazione Android

Human Computer Interaction

Alessio Venturi
Università degli Studi di Firenze
alessio.venturi@stud.unifi.it

Sommario

In questo elaborato abbiamo realizzato un'applicazione Android con lo scopo di offrire agli utenti il raggiungimento di punti di interesse come negozi generici e luoghi di ristoro. L'utente può partecipare attivamente all'inserimento e/o cancellazione di questi luoghi nell'applicazione, grazie al form che può essere compilato all'interno di questa. In questo modo l'utilizzo dell'applicazione può portare in tempi brevi all'aggiornamento dei locali e negozi attivi e non. Il lavoro si è svolto in tre parti: la prima composta dalla progettazione del design dell'applicazione e quindi dalla realizzazione del database e dalla ricerca degli elementi da utilizzare; la seconda parte è stata incentrata nell'implementazione e infine sono stati svolti dei test per comprendere l'usabilità, la portabilità su vari dispositivi e poter trarre delle conclusioni per sviluppi futuri.

1 Introduzione

La Human Computer Interaction (HCI) è una disciplina che si occupa di studiare l'interazione tra persone e computer e di progettare, costruire ed implementare sistemi interattivi di facile utilizzo e che garantiscano all'utente un'esperienza piacevole e positiva. L'utilizzo più intensivo degli smartphone ha reso l'utilizzo delle applicazioni mobili un fenomeno sempre più diffuso, in particolare grazie al loro facile utilizzo per l'utente. Le applicazioni espongono i loro servizi attraverso un'interfaccia grafica ed uno degli scopi dell'HCI è quello di fornire metodologie

appropriate per creare interfacce e applicazioni, al fine di garantire un'efficiente interazione tra il dispositivo mobile e l'utente. NearBy è stata pensata come un'applicazione utile soprattutto a trovare negozi e/o punti di ristoro nelle zone della città come centri-città e stazioni ferroviare. Lo scopo dell'applicazione è quello di offrire all'utente, che si trova o troverà in queste zone, una lista negozi catalogati in base al genere e distanza. Quante volte ci siamo trovati a dover aspettare il treno/bus per ore in stazione senza saper cosa fare? NearBy punta anche a questo aspetto, con la possibilità di inserire il tempo di attesa del nostro mezzo e ottenere una lista di quello che è vicino a noi. L'implementazione si compone di una parte client, che consente all'utente di scegliere quale categoria di negozio visitare, visualizzare i risultati su mappa e partecipare attivamente al progetto, richiedendo l'inserimento o cancellazione di un negozio, e di una parte server che restituisce i dati richiesti tramite query.
Infine è stata effettuata una fase di test, per valutare i punti di forza e le debolezze dell'applicazione al fine di fare considerazioni che permettessero di gettare le basi per eventuali sviluppi futuri.

2 Ionic FrameWork e Angular

Ionic[5] è un framework HTML5 open source, usato per scrivere applicazioni mobile ibride con tecnologie web come HTML, JavaScript, CSS e SASS.
Con Ionic si possono creare applicazioni web progressive (PWAs, Progressive Web Apps) multipiattaforma, che quindi funzionano su ogni piattaforma o dispositivo a partire da un'unica base di codice.
Questo framework offre una vasta libreria front end di componenti per l'interfaccia grafica, ottimizzati per mobile e compatibili con qualsiasi framework JavaScript, come Angular(come nel nostro progetto), React e Vue.
Ionic presenta anche una interfaccia a riga di comando (Ionic CLI) funzionale alla attività di creazione, testing e distribuzione delle app. Inoltre è stato utilizzato anche Typescript, molto simile a JavaScript, ma con funzionalità che ancora non sono state implementate per JavaScript. Ionic emula le linee guida dell'interfaccia utente delle applicazioni native e utilizza gli SDK nativi, unendo gli standard dell'interfaccia utente e le funzionalità dei dispositivi delle applicazioni native.
Ionic utilizza Capacitor (o Cordova) per eseguire lo sviluppo in modo nativo o può essere eseguito nel browser come un'applicazione Web progressiva.
Le applicazioni Ionic sono costituite da blocchi di alto livello chiamati Componenti, che consentono di creare rapidamente l'interfaccia utente. Angular è sempre stato al centro di ciò che rende Ionic performante. Mentre i componenti principali sono stati scritti per funzionare come libreria di componenti Web autonoma,

il pacchetto @ionic/angular rende l'integrazione con l'ecosistema Angular molto semplice e intuitivo.
Gli elementi essenziali per iniziare la creazione di un'applicazione con Ionic sono:

- Node.js per interegire con Ionic
- Un code editor (in questo caso Visual Studio Code [7])
- Una Command-Line interface/Terminal

Dopo la configurazione iniziale, il progetto si presenterà in questo modo :

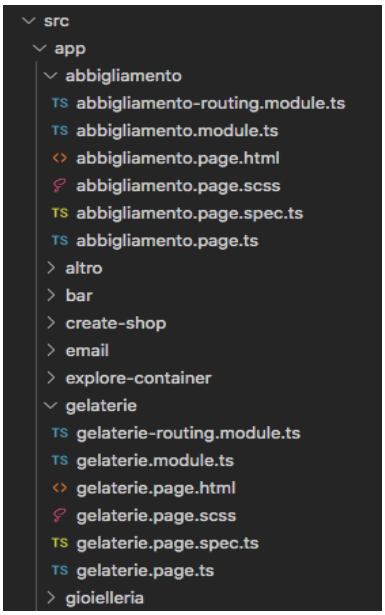

Figura 1: Progetto su Visual Studio Code

All'interno della cartella app, verranno istanziate le cartelle in base a quello di cui avremo bisogno. Ogni cartella è composta da sei file di cui uno HTML, uno CSS e quattro TypeScript. Una cosa molto utile è la visualizzazione del nostro progetto attraverso utilizzo del comando **" ionic serve "** , il quale esegue il codice e ci renderizza la nostra applicazione su Web o su piattaforma Android.

2.1 Android Studio e API utilizzate

Una volta scritto tutto il codice, è possibile esportare il nostro progetto su Android Studio [1], un'ambiente di sviluppo integrato (IDE), attraverso il comando **" ionic cap android open "**. Un progetto su Android Studio è composto da tre parti

principali: la cartella con il codice Java, la cartella res contenente le immagini, file XML e un file di configurazione denominato AndroidManifest.xml per registrare le activity che compongono l'applicazione. Inoltre sono presenti altre cartelle, relative al Capacitor installato attraverso Ionic.

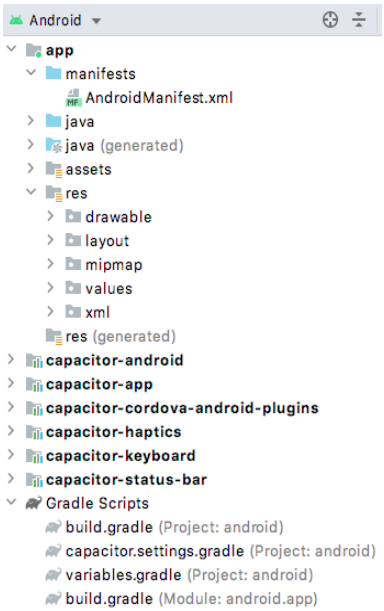

Figura 2: Progetto su Android Studio

Come possiamo notare dalla Figura 2, è presente anche una sezione Gradle Scripts, nella quale si trovano i file di build usati per trasformare il progetto in un'applicazione funzionante. Questa sezione include anche i build delle librerie utilizzate; Per quanto riguarda l'utilizzo delle mappe e della geolocalizzazione sono state impiegate le API Google[4], mentre per la creazione del database è stata utilizzata l'API Google Cloud Firebase[3], un database orientato ai documenti. Ogni documento è formato da :

- Indirizzo del negozio
- Città del negozio
- Numero Civico del negozio
- Descrizione del negozio
- Latitudine del negozio

- Longitudine del negozio
- Tipo di negozio
- Nome del negozio

La creazione del database è stata fatta personalmente all'interno della zona centro/stazione della mia città, in modo da comprendere solamente i negozi situati in quella zona. Attualmente sono presenti 150 ristoranti e 200 negozi.

2.2 Implementazione, interfacce e funzionalità

Per una migliore organizzazione del codice e per rispettare il principio della separation of concern, l'implementazione è stata strutturata facendo riferimento al pattern Model View Controller (MVC). Seguendo questo principio, l'applicazione è suddivisa in tre componenti:

- Modello : contiene i dati principali e le relative funzionalità
- Vista : mostra le informazioni all'utente
- Controller: gestisce l'input e la comunicazione tra vista e modello.

La vista renderizza il modello in una forma adatta all'interazione ed è possibile avere piu' viste dello stesso modello; questo è il piu' grande vantaggio offerto dall'MVC per quanto riguarda la separation of concern. Separation of Concern è un principio di progettazione per separare un sistema in moduli distinti, in modo tale che ogni modulo si preoccupi di un certo compito producendo comprensibilità, riusabilità e scalabilità.
Per quello che riguarda un'applicazione creata attraverso Ionic, il pattern MVC è già implementato in quanto il modello è rappresentato dai documenti all'interno del database (nel nostro caso esterno, ma può essere anche interno) , la vista è rappresentata dal codice che permettere di rappresentare i dati nell'applicazione (nel nostro caso , il codice HTML descritto prima permette questa realizzazione) e infine il controller è rappresentato dal codice TypeScript che mette in relazione il modello e vista.
Una volta installata l'applicazione, l'interfaccia che viene presentata riguarda la prima tab (tre totali), contenente due bottoni che svolgono la funzione di menù a tendina. Il primo riguarda "Cibi e Bevande" mentre il secondo i "Negozi" raffigurati come due icone rappresentative, come vediamo in Figura 3.
Cliccando su questi, appariranno varie categorie le quali porteranno poi, alla scelta della categoria cercata, come possiamo vedere in Figura 4 e Figura 5.

Figura 3: Interfaccia iniziale

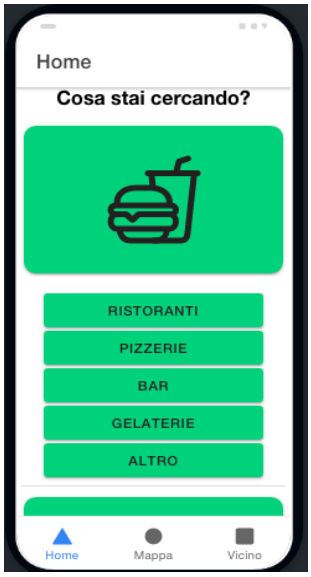

Figura 4: Menù di Cibi e Bevande

Figura 5: Menù di Negozi

Scegliendo una di queste categorie, verrà visualizzata una nuova pagina che caricherà dal database esterno, tutte le attività riguardati quella determinata categoria disponendole in base alla distanza (Figura 6).
Su questa vista è presente anche una "Search Bar" che permette di trovare, se presente, direttamente l'attività cercata. Ogni attività presenta, come possiamo vedere dalla Figura 6 , delle caratteristiche come :

- Nome Attività
- Descrizione Attività
- Indirizzo Attività
- Distanza metri (tempo) dalla nostra posizione.
- Un bottone per il raggiugimento guidato della destinazione

La distanza di metri/tempo è ottenuta grazie alla geolocalizzazione del disposito che, dopo aver dato il consenso alla richiesta di rilevamento, produrrà, ogniqualvolta che apriremo una lista, queste distanze. Premendo il bottone, si aprirà la navigazione, attraverso l'uso di Google Maps, dalla nostra posizione attuale fino alla destinazione, come il percorso preimpostato a piedi (Figura 7). Un importante

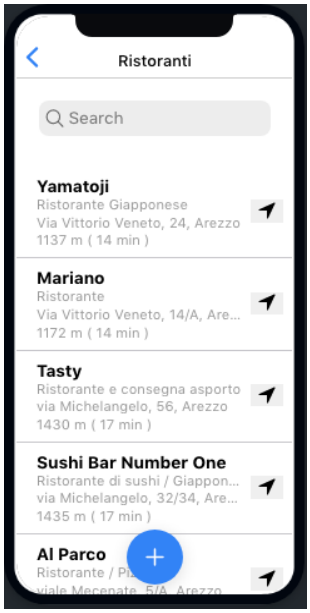

Figura 6: Lista Ristoranti

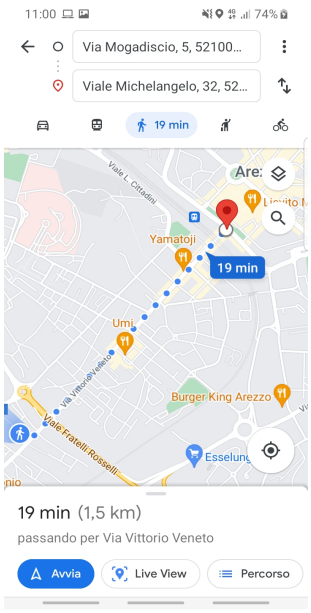

Figura 7: Apertura Maps da bottone

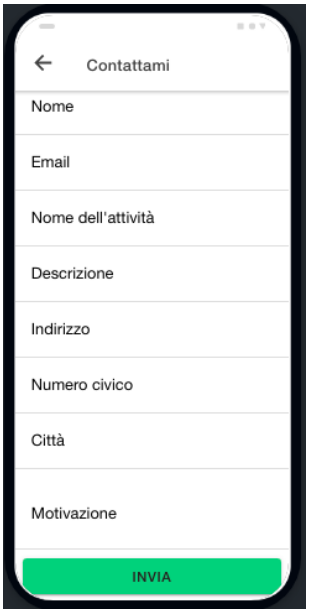

Figura 8: Modulo da compilare

aspetto dell'applicazione, utile per coinvolgere attivamente l'utente, è il form che si ottiene premendo il bottone "+" nella Figura 6.
Questo, ci re-indirizzerà ad un modulo da compilare (Figura 8) sia con i propri dati (Nome ed Email), sia con i dati dell'attività di cui vogliamo aggiungere o rimuovere dando un'apposita motivazione.
Una volta compilato, questo modulo verrà inviato direttamente su un foglio Excel creato da noi, per essere analizzato e verificato. L'utente sarà re-indirizzato all'interfaccia iniziale e riceverà un messaggio di invio avvenuto con successo. Se i campi del modulo presentano errori, questi verranno segnalati in rosso sotto il campo stesso.
Passiamo adesso alla Tab2 (Mappa), dove viene visualizzata una mappa Google, centrata sulla nostra posizione attuale.
Per la geolocalizzazione, sono state utilizzate le API di Google Maps. Come possiamo vedere dalla Figura 9 , è presente un bottone con scritto "Mostra", il quale dopo essere stato premuto, presenta un sottomenu che indica le categorie che si vogliono visualizzare (Figura 10).
Le icone utilizzate sono quelle di Ionic che, rappresentano nel modo più leale, le categorie che rappresentano, fatta eccezione per alcune che non erano presenti nello stack Ionic.

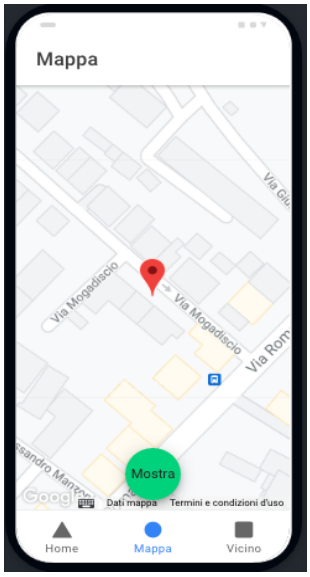

Figura 9: Seconda tab Mappa

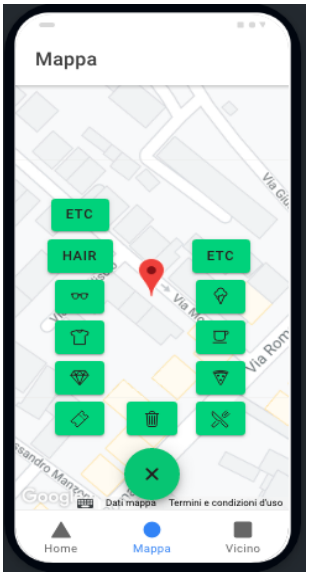

Figura 10: Mappa con apertura menù

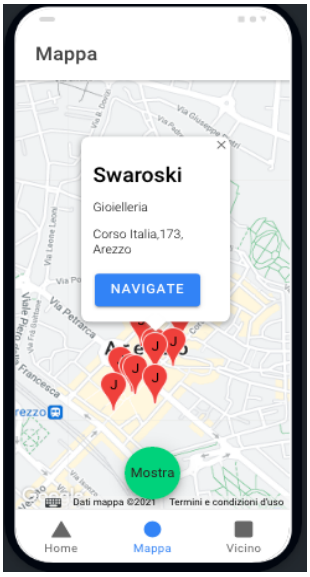

Figura 11: Mappa con selezione marker

Le categorie "ETC" rappresentano "Vario" e "Altro" per quanto riguarda i due grandi insiemi iniziali, mentre il cestino serve a pulire i marker e lasciare la mappa vuota. Premendo su una categoria, verranno visualizzati su mappa le attività di quella categoria, segnalate con una marker rosso ed una lettera, che a loro volta, premendo sul marker, apriranno una finestra di informazione che presenta il tasto per la navigazione, come possiamo vedere in Figura 11. L'ultima tab invece, rappresenta il vero motivo per cui è nata questa applicazione. Siamo nel centro di una città X e vogliamo sapere cosa ci circonda.
Come vediamo dalla Figura 12, ci sono due campi che indicano respettivamente il tempo che abbiamo a disposizione e il tipo di attività che vogliamo. Il tempo selezionabile è fino a tre ore, visto che solitamente non si presenta mai quel tempo di attesa, mentre per il tipo abbiamo le due categorie principali : "Cibi e bevande" e Negozi. Compilando, verranno visualizzate tutte le attività presente nelle vicinanze, in ordina crescente di distanza dal punto in cui ci troviamo (Figura 13). Anche in questo caso, le attività presentano le stesse informazioni che abbiamo descritto precedentemente. Un fattore importante è la selezione del tempo, perchè in questo caso, selezionando un'ora, verranno visualizzati solamente le attività che possiamo raggiungere, visitare per poi tornare entro questa soglia di tempo.
Ad esempio, selezionando un'ora, avremo un massimo di 20 minuti per raggiungere l'attività, altri 20 minuti per la permanenza e infine 20 minuti per tornare al

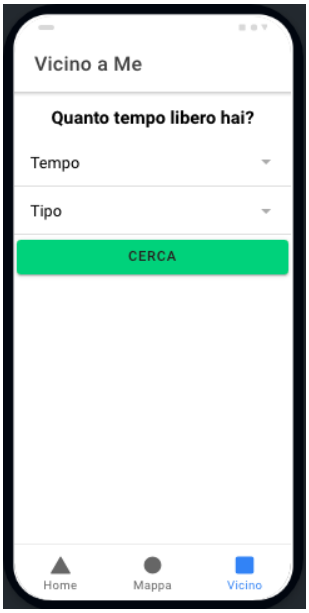

Figura 12: Tarza Tab : Vicino a me

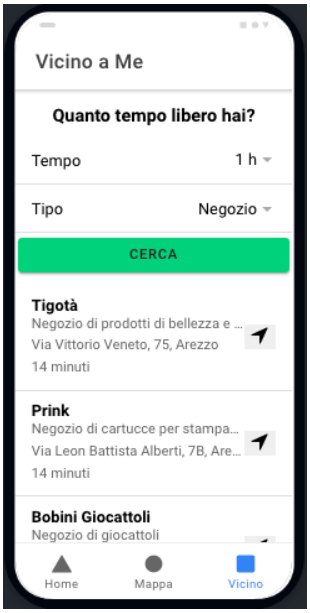

Figura 13: Lista attività vicino a me

punto di partenza, in tempo per il nostro mezzo di trasporto. Per la scelta dei colori, è stato utilizzato "ColorTool"[2] di Material [6], un sistema adattabile di linee guida, componenti e strumenti che supportano le migliori pratiche di progettazione dell'interfaccia utente. Utilizzando questo colore verde, è risultato che la scritta nera, per essere leggibile, deve avere un'opacità minima del 46% per le scritte grandi e del 63% per le scritte piccole, mentre la scritta bianca non risulta leggibile.

3 Scenari ed Usability test

In generale, ogni test viene suddiviso in task ed è possibile distinguere tra task diretti, intesi come istruzioni di natura piu‘ tecnica, e scenari, formulati come istruzioni in un contesto reale in modo che l'utente possa dimenticarsi di essere sottoposto ad un test. Nel nostro caso, per capire quanto l'applicazione fosse user-friendly, sono stati ideati alcuni scenari da presentare alle persone. Prima di effettuare i test , è stato svolto un test pilota, i cui risultati non sono stati analizzati, ma sono semplicemente serviti ad assicurarsi che gli scenari ideati e il questionario a cui sottoporre l'utente fossero adeguati. Lo scenario più utilizzato è stato:

- Sei un utente pendolare e/o lavori nel centro città e hai sentito parlare di questa applicazione NearBy, che ti permette di sapere quali attività si trovano

vicino alla zona stazione/centro e quanto tempo impiegheresti approssimativamente per andare, stare e tornare indietro. Decidi quindi di scaricarla e provare a cercare un negozio.

- Selezioni la categoria Negozi e successivamente la rispettiva che ti interessa. Appare una lista, scegli il negozio e premi sul bottone navigazione. Si aprirà Google Maps con le coordinate già inserite.

- Andando verso quel tipo di negozio , noti che è presente un negozio dello stesso tipo non segnalato. Così torni nella Lista per controllare e non lo trovi. In basso noti la presenza di un bottone e decidi di premerlo, cosi da aprire il modulo per l'inserimento del negozio. Lo compili e lo invii.

- Mentre stai tornando, decidi di aprire la Mappa per vedere dove ti trovi e se nelle prossimità vi è una gelateria. Premi sul bottone "Mostra" e noti delle icone, tra cui quella a forma di cono gelato che decidi di premere. Vengono così visualizzati i marker delle gelaterie ed incuriosito da una , decidi di premere il suo marker. Appare una finestra di info con un tasto di navigazione.

- Il giorno dopo, stai aspettando il treno in una città che non conosci e devi aspettare un'ora per la coincidenza. Decidi quindi di aprire l'applicazione e di andare nella sezione "Vicino a me". Inserisci il tempo e la categoria che ti interessa e appare una lista con le attività vicine a te, che ti permetteranno di andare, stare e tornare in tempo per il tuo mezzo di trasporto.

Questo scenario è stato cambiato leggermente da utente ad utente in modo da cambiare la selezione delle categoria , l'inserimento e/o cancellazione dell'attività. Abbiamo cosi valutato i seguenti aspetti :

- Intuitività

- Facilità d'uso

- Apprendibilità

Le persone sono state reclutate in base al metodo Hallway testing, principalmente in ambito familiare. In totale i partecipanti al test sono stati 14 con età compresa tra i 26 e i 58 anni e di entrambi i sessi. La Tabella 1 riporta la distribuzione dell'età.
Ad ogni candidato, una volta eseguito il test, è stato chiesto di compilare un questionario di 22 domande, di cui le prime personali, altre riguardanti i task svolti e le ultime di carattere generale sull'applicazione. Le domande presentate sono di tipo SEQ, con scala che va da 1 a 7, dove 1 rappresenta **Fortemente in disaccordo** e 7

< 25	25-30	31-50	51-58
0%	85.8%	0%	14.2%

Tabella 1: Età

Fortemente d'accordo. E' stata poi aggiunta una sezione a testo libero nella quale inserire eventuali suggerimenti. Per evitare che l'utente potesse rispondere sempre con lo stesso punteggio, alcune domande sono state formulate con la logica opposta, per esempio : "quanto è stato difficile compilare il questionario?" (domanda negativa) oppure "quanto è stato facile compilare il questionario?" (domanda positiva).

N	Domande	Val Medio	σ
1	Trovo comprensibile la navigazione all'interno dell'app	6.25	0.70
2	L'app rileva correttamente la mia posizione sulla mappa	6.73	0.59
3	Trovo complicata la compilazione del modulo	2.66	2.02
4	Trovo facilmente la categoria richiesta/cercata	6.46	0.63
5	Vorrei avere categorie più dettagliate	3.33	1.39
6	Vorrei più dettagli dei negozi	3.66	1.63
7	Le icone rispecchiano il comportamento dell'app	6.26	0.88
8	Trovo corretta la locazione dei negozi nella mappa	6.6	0.63
9	I tempi di percorrenza/distanza rispecchiano quelli del navigatore	6	1
10	Vorrei avere la possibilità di inserire foto/recensioni dei negozi	5.6	1.50
11	Trovo intuitivo la funzione dei tasti presenti nell'applicazione	6.2	0.86
12	I negozi rispecchiano la categoria selezionata	6.66	0.48
13	La mappa visualizza correttamente i marker dei negozi	6.33	0.72
14	Trovo noioso l'uso dell'applicazione	1.6	0.73
15	Trovo utile la funzione "Vicino a me"	6.66	0.48
16	Il servizio fornito risulta esauriente	6.06	0.88
17	In generale sono soddisfatto dell'applicazione	6.46	0.51
18	Consiglierei l'app ad un amico	6.60	0.50

Tabella 2: Risultati usability test

Complessivamente, tra test e questionario, ogni utente ha impiegato 10 minuti. Dai

test svolti sono emerse alcune problematiche. La prima riguarda le icone presenti nella Mappa una volta che viene utilizzato il tasto Mostra. Non subito è stato compreso da tutti l'attività del cestino presente, che ha l'utilità di ripulire i marker dalla mappa, e di altre icone. Nel caso in cui non ci siano marker presenti, il bottone non ha alcuna funzione. La seconda riguarda le info del negozio date quando si preme su di esso nella lista: circa il 40% degli utenti, avrebbe preferito avere più informazioni, mentre quasi l'80% avrebbe voluto la possibilità di inserire foto e/o recensioni nelle informazioni.
Per quanto riguarda la domanda 3, l'alta deviazione standard è probabilmente indice del fatto che questa è stata posta in modo negativo e alcuno candidati potrebbero non essere stati troppo attenti durante la sua lettura. Contrariamente dalle aspettative è invece risultata piuttosto chiara la presentazione del trend. I risultati delle domande 1,4,7,11 dimostrano che le persone sono state in grado di capire come muoversi correttamente all'interno dell'applicazione e come portare a termine i task relativi. In generale gli utenti sono rimasti soddisfatti dall'applicazione non è stata trovata noiosa, tanto da essere d'accordo nel consigliarla ad altre persone. Infine, alcuni candidati hanno rilasciato dei suggerimenti interessanti:

- Inserire dei filtri per personalizzare e facilitare ancora di più la ricerca.
- Inserimento di immagini (es. entrata principale) e recensioni nei dettagli delle attività.
- Poter cercare la città (in cui sto per andare) anche se mi trovo in un'altra città per sapere in anticipo cosa fare/dove andare.

Per quanto riguarda l'ultimo suggerimento, essendo stato creato il database nella città di Arezzo, l'opzione di selezionare altre città non è stata presa in considerazione inizialmente per motivi logistici e tempistici.

4 Sviluppi futuri e conclusioni

La sezione suggerimenti, presente nel questionario compilato dai candidati crea le basi per gli sviluppi futuri, che possono riguardare un arricchimento delle funzionalità già presenti. La possibilità di inserire foto, nelle info generali delle attività e piccole recensioni degli utenti sono le più richieste dagli utenti. L'incremento della categorizzazione potrebbe aiutare ad essere più mirati nell'obbiettivo finale. Inoltre è stato suggerito di inserire la categoria "parchi", in quanto nella città di Arezzo, molti di questi sono vicino alla zona centro/stazione. L'idea che la comunità possa interagire attivamente all'aggiornamento della lista delle attività è stata

particolarmente apprezzata. Le persone che hanno effettuato i test hanno apprezzato la scelto del logo , del nome e del design in generale, trovandola oltre che utile, anche piacevole da usare interagendo con essa. Inoltre la funzione "Vicino a Me" è stata apprezzata in quanto, molti di questi utenti, sono pendolari e/o lavorano nella zona centro/stazione.

Riferimenti bibliografici

[1] Android studio. https://developer.android.com/.

[2] Color tool. https://material.io/resources/color/#!/?view.left=0&view.right=0.

[3] Google firebase. https://firebase.google.com/.

[4] Google maps platform. https://developers.google.com/maps/documentation/geolocation/overview.

[5] Ionicframework. https://ionicframework.com/.

[6] Material. https://material.io/.

[7] Visual studio code. https://code.visualstudio.com/.