# Progetto Human Computer Interaction: sviluppo di un'applicazione android di navigazione e identificazione di stazioni di servizio

Bernardo Tiezzi
7036865
bernardo.tiezzi@stud.unifi.it

## Abstract

*Nella relazione seguente viene presentato il progetto finale per l'esame del corso di laurea magistrale in Ingegneria Informatica tenuto dal professor Andrew D. Bagdanov. L'obiettivo è realizzare un'applicazione di navigazione e geolocalizzazione di impianti di distribuzione di carburante per i veicoli a combustione interna. In questo paper vengono descritte le varie fasi di progettazione, partendo dai processi di ricerca a priori (NeedFinding), per arrivare alla fase di sviluppo ed ai test di usabilità. L'ambiente di sviluppo utilizzato è Android Studio con linguaggio di programmazione Java.*



## 1. Introduzione

Al giorno d'oggi, la diffusione dei sistemi mobile (smartphone, tablet, etc) ha incrementato enormemente l'interazione tra sistemi software e utenti. L'HCI (Human Computer Interaction), detta anche interazione uomo-macchina, è lo studio di come interagiscono i computer con l'utenza, in modo da progettare sistemi interattivi che risultino affidabili, usabili e che supportino e facilitino le attività umane. Uno degli aspetti principali da tenere in considerazione durante lo sviluppo software è la *user experience* (**UX**). Lo sviluppo di un'applicazione con sistema operativo Android consente di ottenere un prodotto che rispecchi le caratteristiche richieste dall'utente utilizzatore.

### 1.1. Applicazioni di Geolocalizzazione

Le applicazioni di geolocalizzazione sono tra le più diffuse ad oggi per i dispositivi mobile in circolazione. L'accesso alla localizzazione dell'utente avviene sempre previa richiesta autorizzazione che, una volta concessa, a meno di modifica dei permessi, è sempre garantita. Uno dei metodi più comuni di utilizzo della geolocalizzazione è la collocazione all'interno di una mappa che consenta all'utente di orientarsi. Un ulteriore metodo di impiego è l'individuazione delle località vicine che possono risultare di interesse per l'utilizzatore. La geolocalizzazione deve comunque essere gestita in modo da non risultare troppo intrusiva; un suo utilizzo incauto può recare danno al successo dell'applicazione stessa.

### 1.2. App Carburanti

L'applicazione realizzata consente all'utente di individuare le stazioni di rifornimento nelle vicinanze, fornite da un database apposito. A seconda delle esigenze è possibile eseguire un filtraggio sia in base al gestore della stazione di servizio, sia in base al tipo di carburante desiderato. Il cliente è in grado di ottenere ulteriori informazioni al momento della selezione dell'impianto. Al fine di raggiungere la destinazione desiderata, viene fornito l'itinerario del percorso e assistenza durante il tragitto.

### 1.3. Android Studio

Android Studio è un ambiente di sviluppo per la creazione di applicazioni distribuite sulla piattaforma Android. Fu annunciato il 16 mag-

gio 2013 in occasione della conferenza **Google I/O** dal product manager Google, Katherine Chou. Integra un editor IDE con layout visuale; è riconosciuto come il tool ufficiale per l'implementazione di applicazioni del sistema operativo omonimo [3]. Il linguaggio di programmazione utilizzato in Android Studio è **Java**. Il download è gratuito e disponibile su sistemi operativi *Windows*, *Linux*, *macOS*. La prima build stabile dell'IDE risale al dicembre 2014; con la sua distribuzione Google ha sostituito Eclipse, fornendo un ambiente di sviluppo notevolmente migliorato, sia dal punto di vista delle funzionalità che dell'efficienza nel sviluppare applicativi.

## 2. Needfinding

Prima ancora di iniziare a sviluppare un'applicazione è necessario avviare un processo di scoperta a priori, riguardante i bisogni e le necessità delle persone; tale fase è nota come **NeedFinding**. Infatti, prima di procedere allo sviluppo software, un team di sviluppo deve comprendere le motivazioni che lo portano a sviluppare un determinato applicativo. Per conoscere i bisogni delle persone si possono ottenere informazioni da più fonti: focus group, sondaggi, interviste ed altro. Le informazioni raccolte ci consentono di definire delle Personas e degli scenari in modo da estrarre una prima idea di software. Le esigenze dei clienti sono l'elemento che influisce maggiormente su come viene progettato un prodotto.

### 2.1. Personas

Per individuare le personas sono state raccolte informazioni attraverso interviste a familiari e amici, così come agli addetti presso le stazioni di rifornimento. A seguito di una fase di brainstorming, che consente di produrre un arsenale di potenziali soluzioni alle richieste dei clienti, sono state identificate le seguenti personas:

1. **Nome**: Manuel
   - **Età**: 18-24 anni
   - **Residenza**: Campi Bisenzio, Firenze
   - **Professione**: Dipendente privato
   - **Lavoro**: Manuel è un giovane dipendente per una ditta di materiali edili. Tale azienda si trova al di fuori del suo comune di residenza. Il suo impiego può richiedere di lavorare al banco o di effettuare consegne a clienti o aziende private. Per le consegne può utilizzare furgoni di piccola taglia o camion di medie dimensioni. L'itinerario delle consegne è variabile. L'azienda fornisce mensilmente al dipendente buoni benzina da poter spendere in stazioni di servizio con gestore *Agip Eni*
   - **Tempo Libero**: Manuel è solito frequentare amici di infanzia che abitano nel suo stesso comune. Ha una relazione ed esce abitualmente il sabato sera. Predilige visitare città o centri commerciali, frequentare discoteche.

2. **Nome**: Susanna
   - **Età**: 46-59 anni
   - **Residenza**: Montespertoli, Firenze
   - **Professione**: Docente scolastico
   - **Lavoro**: Susanna è un'insegnante che svolge la funzione di vicario del dirigente scolastico, lavora presso un'istituto comprensivo esterno al proprio comune di residenza. Durante la giornata lavorativa è solita percorrere diversi chilometri per raggiungere l'istituto e predilige strade a rapido scorrimento. Risulta proprietaria di un'utilitaria Fiat Punto a gasolio.
   - **Tempo Libero**: Susanna ha stretto amicizia con sue colleghe di lavoro, che risiedono al di fuori del suo comune. Le piace fare jogging ma predilige le lunghe camminate o organizzare viaggi in compagnia delle amiche.

### 2.2. Requisiti di base

Le **Personas** individuate devono condurre lo sviluppatore ad un ragionamento riguardo i requisiti di base per l'applicativo da realizzare. Essendo un'app di navigazione è necessario che

l'interfaccia risulti intuitiva e fornisca un buon feedback ad ogni azione intrapresa dall'utente. Questo implica che l'utente deve essere in grado di avviare la navigazione verso un determinato luogo in autonomia, a prescindere dalla propria dimestichezza nell'utilizzo dei dispositivi mobile. Bisogna considerare la possibilità che l'utente interagisca con il proprio cellulare durante la guida; la sequenza di comandi deve essere rapida e facilmente accessibile.

## 3. Paper Prototyping

In questa fase di sviluppo, partendo dai risultati ottenuti durante il processo di NeedFinding, si è realizzato il MockUp dell'applicazione, definendo un design dell'interfaccia utente che riassuma l'aspetto che l'applicativo avrà al termine dell'implementazione del codice. In Fig.1 e in Fig.2 viene mostrato il modello realizzato. Come


Fig. 1. Interfaccia principale

è possibile notare in Fig.1, l'applicativo dovrà rispettare i requisiti precedentemente descitti nel paragrafo 2.2. La schermata principale presenta:

- Una Mappa
- La posizione attuale del client tramite GPS
- Le icone dei distributori, posti nelle vicinanze dell'utente; ogni simbolo viene identificato in base al gestore del distributore ed in base al prezzo del carburante, posizionato al di sopra del marker.
- un tasto che consente l'accesso al menù di filtraggio in base al gestore del distributore
- un tasto, con funzionalità analoghe al precedente, dedicato alla gestione del carburante desiderato.

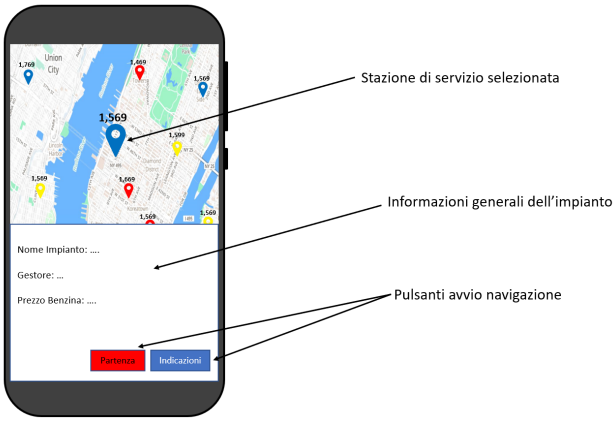


Fig. 2. Pop-Up con informazioni riguardo l'icona selezionata

La selezione di un impianto sulla mappa consente l'accesso ad una nuova view, che riporta ulteriori informazioni riguardo il distributore selezionato, assieme ai pulsanti di navigazione. La view è presentata come un pop-up, traendo ispirazione dell'UI di *Google Maps*. L'icona selezionata viene evidenziata rispetto alle altre, aumentandone le dimensioni sulla mappa.

## 4. Il Progetto

Dopo aver definito il modello dell'interfaccia utente si avvia la fase di implementazione del codice. L'ambiente di sviluppo utilizzato, descritto in precedenza, è Android Studio; il linguaggio di programmazione è Java.

### 4.1. Database: Google Firebase

Per visualizzare le icone sulla mappa si è reso necessario utilizzare un database che memorizzasse le informazioni di ogni impianto e consentisse l'accesso ad ogni utente in possesso dell'applicativo. Il database realtime utilizzato viene fornito da Google Firebase[2]. Quest'ultimo è un potente servizio online, ac-

quisito da Google nel 2014, che consente di salvare e sincronizzare i dati elaborati da applicazioni web e per dispositivi mobili. Si tratta di un database **noSQL** ad alta disponibilità ed interagibile in vari progetti software, sottoscrivendo un account di servizio. Ad ogni account viene associato un indirizzo che è impiegato all'interno del sorgente come riferimento alla base dati. L'URL consente di accedere, tramite browser, al pannello di controllo del database. I dati inseriti non sono strutturati in records o tabelle, ma attraverso la costruzione di un **albero JSON**. L'inserimento dei dati relativi agli impianti è stato effettuato precedentemente all'implementazione del software. Le informazioni sono state scaricate dal sito del MISE (Ministero dello Sviluppo Economico); comprendono ogni stazione di servizio presente sul territorio italiano. L'accesso in lettura ai dati contenuti nel database avviene durante la fase di inizializzazione dell'app.

### 4.2. MapBox

Per usufruire delle mappe di navigazione sono state incluse all'interno dei file *gradle* le MapBox API. MapBox è una piattaforma che offre servizi di mappatura a prezzi ragionevoli e non richiedenti informazioni di fatturazione se si usufruisce del livello gratuito. L'accesso open source a buona parte dei servizi di mappatura consente agli sviluppatori di testare le loro applicazioni in libertà e di fornire l'indirizzo di fatturazione solo quando i software sviluppati avranno ottenuto risultati soddisfacenti. Al termine della fase di avvio la schermata principale riporta sulla mappa, in base alle coordinate geografiche, la posizione dell'utente e le icone situate nelle vicinanze.

### 4.3. Implementazione

Il progetto è suddiviso in file differenti. Di seguito vengono presentati i file sorgente realizzati.

1. `MainActivity.java`
2. `PopActivity.java`
3. `CommonModelClass.java`
4. `FuelStation.java`
5. `Utils.java`

La view principale viene gestita dal file `MainActivity.java` che si occupa di realizzare il MockUp mostrato in Fig.1, con eventuali modifiche per migliorare l'usabilità dell'applicazione. Al momento dell'avvio, in

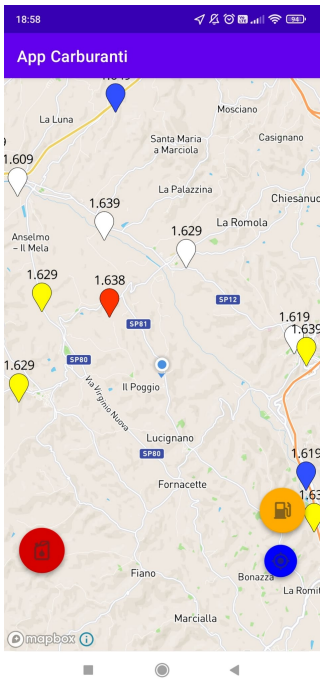


Fig. 3. View principale

fase di installazione, vengono richiesti i permessi per la geolocalizzazione. Nel caso non siano concessi l'applicazione mostra un dialogo all'utente, informandolo della necessità di attivare il permesso, al fine del funzionamento dell'applicativo. Una volta concessi i permessi, viene eseguito l'accesso al database Firebase, raccogliendo tutte le informazioni riguardanti gli impianti attivi sul territorio. La classe *FuelStationLatLng*, contenuta all'interno del file `FuelStation.java`, raccoglie internamente le informazioni riferite ad ogni distributore. Utilizzando la classe *SymbolManager* sono inizializzati i simboli sulla mappa con icone differenti in base al gestore della stazione:

- Rosso = Tamoil

- Blu = Q8
- Giallo = Agip Eni
- Bianco = altre stazioni

All'interno della classe `MainActivity` sono inizializzati due pulsanti *Floating Action Menu*, i quali mostrano due liste di *Floating Action Button* per la gestione del filtraggio sulle icone, come mostrato in Fig.4. Il menu di sinistra

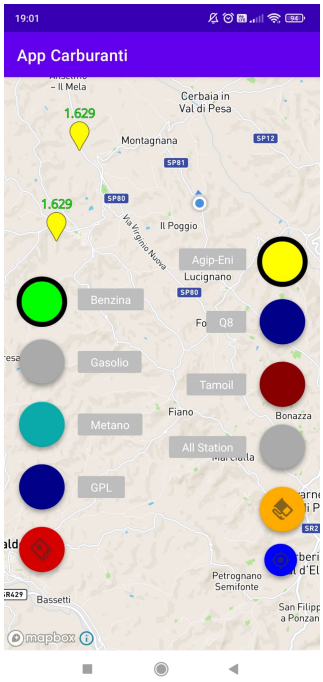


Fig. 4. Lista pulsanti per la selezione degli impianti

consente all'utente di selezionare il tipo di carburante desiderato, che viene riportato al di sopra dell'icona. Il menu posto a destra permette di eseguire un filtraggio sulle tipologie di distributori, visualizzando soltanto quelli che presentano come bandiera il gestore selezionato. Il bottone sottostante al menu è stato aggiunto successivamente, poichè ritenuto utile all'utilizzatore, durante l'esplorazione della mappa. Questo ricentra la mappa sulle coordinate dell'utente.

L'interazione con le icone disposte sulla mappa viene gestita tramite la funzione `symbol.addClickListener();` al memento del click viene inizializzata la classe `PopActivity`, che gestisce il pop-up contenente le informazioni relative al simbolo selezionato (Fig.5). I dati sul distributore sono riportati sotto forma di tabella, sotto la quale è posizionato il pulsante **Indicazioni**. Quest'ultimo costruisce l'itinerario del percorso dalla posizione dell'utente fino alla stazione selezionata. Una volta stabilito il tragitto, cliccando sul pulsante **Naviga**, che sostituisce **Indicazioni**, si avvia l'interfaccia di navigazione verso il luogo designato. Il pop-up viene chiuso interagendo con la porzione di mappa sovrastante. La classe

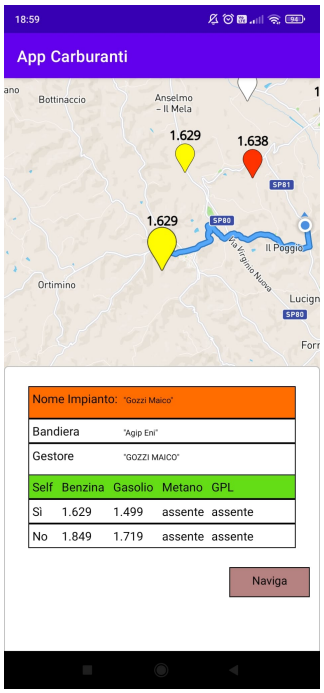


Fig. 5. Pop Up informativo per l'icona selezionata

`CommonModelClass` è impiegata durante l'interazione con un simbolo sulla mappa. Implementa il design pattern **Singleton**, per consentire di allocare una singola istanza per la classe `MainActivity`, accessibile globalmente dalle altre attività (*PopActivity*). Viene allocata in fase di avvio del software, a cui è assegnata l'istanza della classe `MainActivity` come attributo. La classe `Utils` definisce i metodi per la modifica dello stato dell'applicazione, andando a gestire il filtraggio sui distributori come richiesto dal cliente.

## 5. Scenari

Per comprendere quanto l'applicazione realizzata rispecchi i bisogni degli utenti che andranno a scaricarla, sono stati progettati dei possibili scenari di impiego del prodotto. Ogni scenario è stato definito traendo ispirazione dalle **personas** descritte nel paragrafo 2.

- **Personas**: Manuel
- **Descrizione**: Stai effettuando una consegna per la ditta per cui lavori o ti stai recando sul luogo di lavoro. Durante il percorso necessiti di fermarti per fare rifornimento di carburante. Hai a disposizione dei buoni carburante, i quali possono essere spesi solo presso stazioni convenzionate. Utilizza l'applicazione per individuare il distributore più vicino che ti consenta di utilizzare il buono.
- **Personas**: Susanna
- **Descrizione**:Ti sei recato a trovare un tuo amico/a per fare un viaggio assieme a lui/lei durante il fine settimana. Decidete di utilizzare una sola macchina per spostarvi, in modo da potervi dividere le spese di viaggio. Prima di partire decidete di fare il pieno, così da non averne bisogno durante il week end. Utilizza l'applicazione per individuare il distributore più vicino e che fornisca il prezzo del carburante desiderato più economico.

La genericità degli scenari presentati consente di valutare l'applicazione da più punti di vista, tenendo in considerazione alcune caratteristiche di fondamentale importanza per il successo del prodotto finale. La User Experience verrà valutata in base all'intuitività dell'interfaccia utente, alla semplicità nell'impostazione dei comandi di navigazione e durante la fase di installazione. Il prodotto deve risultare creativo ed innovativo per attirare l'interesse del cliente, con una GUI ben strutturata ed accattivante.

## 6. Usability test

La fase conclusiva per lo sviluppo dell'applicazione consiste nella valutazione del prodotto finale attraverso i testi di usabilità. Prima di procedere oltre bisogna comunque apportare alcune considerazioni: durante la varie fasi di implementazione non si è mai perso il contatto con i consumatori. Il codice sorgente è stato più volte modificato in base alle risposte ottenute dagli utenti; si può definire dunque una fase di sviluppo ciclica, caratterizzata da varie modifiche dei metodi e delle classi che gestiscono l'interfaccia. Al termine dei primi processi di scrittura del codice, sono stati eseguiti dei *pilot test*, rivolti ad un numero ristretto di persone, per aiutare lo sviluppatore a riconoscere eventuali problemi di struttura che, in un primo momento, non erano stati individuati. Grazie ai test preliminari è stato possibile apportare modifiche che hanno aumentato notevolmente l'usabilità del prodotto. Per esempio sono stati rilevati alcuni **bug nella gestione dei menu per la selezione delle icone e del carburante fornito** oppure nella fase di **impostazione dei processi di navigazione**. Al termine dello sviluppo del codice, sono state selezionate per la fase di testing persone appartenenti all'ambito familiare, legate da rapporti di amicizia o conoscenti. In totale sono risultati 12 partecipanti, di età compresa tra i 18 e i 59 anni per entrambi i sessi. Di seguito sono riportate le percentuali di distribuzione per età degli intervistati:

| **Fascia d'età** | 18-23 | 24-39 | 40-59 |
|---|---|---|---|
| **Percentuale** | 25% | 33.3% | 41.2% |

Table 1. Età dei partecipanti

Ad ogni intervistato, una volta eseguito il task assegnatogli, è stato rischiesto di rispondere ad un questionario, realizzato utilizzando **Moduli Google**. Il servizio consente di realizzare un form per immagazzinare e rielaborare i dati con estrema facilità. I questionari sottoposti ai clienti presentano domande di tipologia *Single Ease Question*[1] (SEQ) con una scala da 1 a 7, in modo da valutare il livello di gradimento

dell'utente, una volta concluso il task assegnato. Un esempio di SEQ è riportato in Fig.6. Le do-


Fig. 6. Simple Ease Question

mande poste sono state riportate all'interno della tabella 2, con la media dei risultati ottenuti e lo scarto quadratico $\sigma$. Dalle informazioni ottenute tramite i test di usabilità si possono apportare alcune considerazioni:

1. la maggioranza degli utenti non ha avuto problemi nella rilevazione della posizione; infatti le risposte non presentano un valore di $\sigma$ elevato, mentre la media delle valutazioni è alta.

2. gli utenti hanno interagito facilmente con i menu di selezione ed hanno compreso quali pulsanti consentono di selezionare la tipologia di distributore desiderata.

3. le informazioni riportate all'interno delle view dei distributori sono ritenute esaustive e il processo di navigazione è risultato piacevole.

4. gli utenti hanno opinioni divergenti sull'utilità di visualizzare impianti distanti dalla propria posizione, così come riguardo l'eventualità di sviluppare una versione IOS dell'applicazione.

5. nel complesso gli utenti hanno giudicato positivamente la loro esperienza con l'applicazione e sarebbero disposti eventualmente a consigliarne l'utilizzo.

### 6.1. Analisi dei Risultati

Al termine della fase di testing sono stati eseguiti ulteriori aggiornamenti, basandosi sulle risposte ottenute dai questionari. La modifica principale ha riguardato la gestione delle icone in base alla posizione dell'utente. Grazie ai permessi concessi dall'utente è stato possibile ridurre il numero di marker disposti sulla mappa, limitandosi a visualizzare i distributori in prossimità dell'utilizzatore. L'applicazione esegue un aggiornamento dell'interfaccia in base alla posizione del dispositivo mobile. Per facilitare il riconoscimento del comando impostato utilizzando i menu di selezione, si è modificato il colore dei pulsanti non premuti, aumentandone l'opacità. Questa soluzione ha consentito di ridurre l'utilizzo di colori troppo accesi sulla mappa.

## 7. Conclusioni e Sviluppi Futuri

Il progetto realizzato è un'applicazione android di geolocalizzazione e di navigazione. Le fasi di sviluppo descritte hanno consentito di ottenere un prodotto finale ritenuto utile, gradevole e semplice da utilizzare. Gli utenti intervistati sono risultati entusiasti dell'applicativo tanto da essere disposti a consigliarlo ad altre persone. Per ampliare le possibilità di impiego, in futuro, potrebbe essere utile aggiungere le posizioni di stazioni di rifornimento per le auto elettriche, ad oggi non diffuse quanto i veicoli a combustione interna, ma in aumento negli ultimi anni. Una funzionalità aggiuntiva dovrebbe indicare all'utente la presenza di luoghi di ristoro (bar, ristoranti, autogrill), nei pressi delle stazioni di rifornimento; tale update porterebbe un vantaggio economico anche ai ristoratori, incentivando il download del software. La realizzazione di una versione IOS aumenterebbe ulteriormente il numero di dispositivi compatibili. In conclusione, va sottolineato come un aspetto fondamentale per lo sviluppo di un applicativo, oltre alla correttezza del codice, è il continuo rapporto con l'utente. Il lavoro di uno sviluppatore deve essere rivolto ad un miglioramento costante dell'UX, venendo incontro ai bisogni dei suoi clienti potenziali.

| n° | Domanda | **media** | $\sigma$ |
|---|---|---|---|
| 1 | L'app consente di individuare facilmente la propria posizione? | 6.84 | 0.54 |
| 2 | La posizione rilevata è coerente con la locazione effettiva? | 6.67 | 0.62 |
| 3 | L'interazione con la mappa risulta semplice e di rapida esecuzione? | $5,83$ | 0.9 |
| 4 | L'accesso ai pulsanti per la selezione del carburante risulta intuitivo? | 5.67 | 0.94 |
| 5 | L'associazione tra tipologia di carburante e valore riportato sull'icona è comprensibile? | 5.75 | 0.92 |
| 6 | Le informazioni riguardanti ogni stazione sono soddisfacenti ed esaustive? | 6.17 | 0.99 |
| 7 | Il processo di navigazione presso il distributore selezionato è di facile attuazione? | 6.33 | 0.62 |
| 8 | Avrei preferito non visualizzare distributori troppo lontani | 4.59 | 1.6 |
| 9 | La navigazione è fluida e piacevole | 5.58 | 1.04 |
| 10 | In futuro vorrei fossero aggiunte le stazioni di ricarica per le auto elettriche | 5.83 | 1.67 |
| 11 | Mi piacerebbe fosse disponibile una versione IOS del prodotto | 4 | 2.52 |
| 12 | Nel complesso, come giudichi l'esperienza di viaggio? | 6.25 | 0.6 |
| 13 | Consiglieresti l'applicazione ad un amico? | 6.42 | 0.76 |

Table 2. Risultati usability test

## References


[1] A. D. Bagdnanov. Human computer interaction, 2020. Usability Testing. 6

[2] G. Maggi. Html.it, 2015. Firebase: cos'è e come funziona il backend NoSQL. 3

[3] L. Tecnologia. Android. Android: cos'è e come Usarlo. 2