Documentazione di Analisi di Progetto Informatico

Progetto d’esame

Coda Negozi

App Android client / server

Realizzato da: Sestito Simone

Piano di Progetto

1 Introduzione

1.1 Scopo

Lo scopo del presente documento è quello di specificare gli obiettivi del progetto, descrivendone i dettagli, identificando le tappe e, in relazione ad esse, gli obiettivi da raggiungere, le attività, la gestione della comunicazione e i tempi di massima di attuazione del Progetto.

1.2 Riferimenti

Sviluppo progetto riepilogativo per le esperienze maturate nel percorso scolastico.
Si predispone per il collegamento tra tutte le materie d'indirizzo, dovendo utilizzare le conoscenze acquisite in determinati ambiti per la corretta realizzazione del progetto.

2 Ambito e obiettivi

Il progetto consiste nella realizzazione di un servizio, con annesso client app Android, per la **prenotazione della coda** in esercizi commerciali convenzionati.
La coda è **virtuale**, pertanto la prenotazione può essere effettuata **da remoto**.
Nel contesto storico sociale odierno permette di evitare lunghe code al di fuori di supermercati, alimentari o altre attività commerciali
Si ispira al sistema già esistente da tempo di prenotazione code degli uffici postali.

Non esistono sul mercato soluzioni simili e ben diffuse tra la popolazione. Esistono casi isolati non del tutto pubblicizzati o popolari.
Questa soluzione qui proposta non si pone un target specifico di utenza, in quanto chiunque sia dotato di uno smartphone e abbia sufficienti abilità nel suo utilizzo basilare sarà facilmente in grado di utilizzare questo servizio.

Vista la situazione in cui si sta vivendo, è ideale per evitare assembramenti non necessari al di fuori dei negozi, sui marciapiedi e sulle strade.
Per sottolinearne l'esigenza, si riporta il post su Facebook datato 21 marzo 2020 di Teresa Bellanova, ministra delle politiche agricole alimentari e forestali, in cui si legge "Servono soluzioni alternative come ad esempio [...] l'uso di strumenti tecnologici (ma non solo) per verificare la lunghezza delle code e prenotare il proprio posto" (fonte: https://m.facebook.com/story.php?story_fbid=2814783488643375&id=310949775693438)

3 Requisiti funzionali

3.1 Requisiti per l'utente

Senza accesso con un account di qualunque tipo, non è consentito l'utilizzo dell'app.
Al fine di evitare la creazione di account fittizi, è richiesta la verifica dell'indirizzo e-mail utilizzato. Bisognerà aprire un link temporaneo che verrà inviato via e-mail per attivare l'account.
L'utente, previa registrazione, ha a disposizione una **mappa** con evidenziati gli esercizi commerciali convenzionati nelle vicinanze. Potrà effettuare una **ricerca** in base al nome del negozio presso cui vuole prenotarsi. I risultati dovranno essere ordinati per distanza crescente dalla propria posizione.
Potrà effettuare una **prenotazione** su un punto vendita, consultare il numero di persone già in coda e vedere l'elenco delle prenotazioni da lui effettuate, nonché annullarle.
Infine, riceve una **notifica push** sul proprio dispositivo quando manca poco al suo turno o se la prenotazione è stata annullata dal commerciante.

3.2 Requisiti per il commerciante utilizzatore del servizio

L'esercente, dopo aver ricevuto un **account** apposito da un amministratore del servizio, è pronto per lavorare. Il suo negozio comparirà tra quelli convenzionati a tutti gli utenti dell'app.
Può consultare **l'elenco** delle persone prenotate e **chiamare** la prossima persona in coda. In caso lo desideri, può provvedere ad annullare tutte le prenotazioni finora effettuate dai vari utenti. Questi ultimi verranno avvisati. Un commerciante può gestire un unico negozio. Per gestire più negozi si devono usare account diversi, l'app provvederà al salvataggio delle credenziali di ciascuno di essi per un rapido passaggio da uno all'altro. Al servizio non riguarda la gestione di una coda tradizionale. L'esercente può organizzare la situazione indipendentemente, ad esempio gestendo due file, una tradizionale e una virtuale.

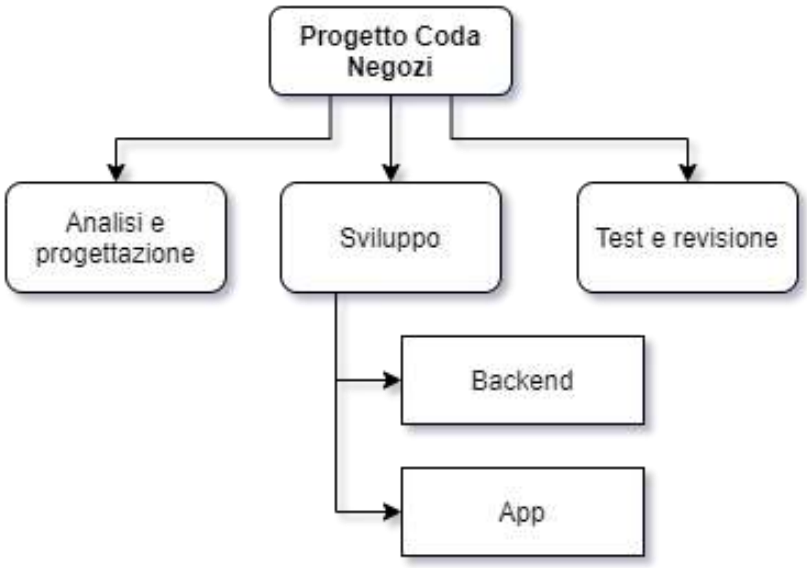
3.3 Requisiti per il fornitore/amministratore del servizio

L'amministratore dell'app è libero di eseguire operazioni CRUD (creazione, lettura, aggiornamento ed eliminazione) su utenti e negozi.
Non ha la facoltà di visualizzare le informazioni sensibili degli utenti (commercianti e altri amministratori inclusi) come le password: esse verranno salvate nel database usando una funzione hash, rendendone impossibile la lettura in chiaro e garantendo un alto livello di sicurezza.
Il primo accesso avviene tramite l'invio da parte del client delle credenziali (e-mail e password) al server, il quale risponderà fornendo un token di accesso. Questo sarà conservato dal client e usato in tutte le successive operazioni. Non ha scadenza, e non servirà più fornire e-mail e password, eccetto dopo un log out che revocherà immediatamente il token di accesso.

4 WBS (Work Breakdown Structure) di Progetto

La WBS è una scomposizione gerarchica del progetto nei suoi elementi ed azioni costitutivi, generata allo scopo di migliorarne la gestione e il controllo.

Di seguito si riporta uno schema sintetico della WBS identificata per il Progetto.



5 Architettura generale

Si prevede un'architettura del progetto di tipo client server, dove il ruolo di client è svolto da un'app Android mentre lato server sarà presente un backend che gestirà le richieste del client.

5.1 Architettura server

Il server espone una REST API a cui il client si potrà interfacciare.
Si prevede il rilascio di una documentazione della già menzionata API sotto forma di elenco di endpoints disponibili con tanto di esempi di utilizzo.
Il programma server è pensato seguendo le best practices per la creazione di una REST API.
Il file index.php gestirà tutte le richieste: questo rende possibile la costruzione degli URL seguendo le linee guida per i servizi RESTful. Ad esempio, per ottenere un utente con ID 1, l'URI sarà /users/1 anziché /users?id=1. I dati vengono inviati e ricevuti in formato JSON.
Il progetto prevede una divisione logica dei layer applicativi secondo un approccio coerente con il pattern architetturale MVC (Model View Controller), rimanendo però di tipo dual tier e single host (tutti i livelli sono ospitati sulla stessa macchina server, ma c'è il client che è il secondo tier).
Non è previsto l'uso di alcuna libreria di terze parti.
Per una documentazione più dettagliata, si può fare riferimento alla pagina del Progetto (in inglese).

5.2 Architettura client

Il client utilizza i più recenti standard in materia di sviluppo app Android, con tutti i vantaggi che ne derivano. Segue il **pattern** architetturale MVVM (Model View ViewModel).
Prevede diverse schermate per i vari ruoli utente (utente semplice, commerciante, amministratore). All'inizio si ha la fase di **login e/o registrazione** nuovo utente, poi si verrà reindirizzati sulla schermata più idonea.
Il token di autenticazione viene conservato all'interno delle SharedPreferences, ovvero un database chiave/valore (non SQL) inaccessibile dalle altre app sul dispositivo. Esso verrà salvato crittografato sfruttando le API di sistema: garantiscono che la chiave di cifratura sia inaccessibile e la crittografia eseguita su hardware apposito nel dispositivo, ove disponibile. Per le **notifiche** push utilizza Firebase Cloud Messaging.
Per eseguire le richieste viene usata la libreria Retrofit, che si occupa di eseguire la trasformazione da oggetto Java a JSON e viceversa, insieme alle sue dipendenze per l'esecuzione della richiesta HTTP vera e propria.
Per la realizzazione dell'interfaccia grafica, al fine di mantenere la coerenza con il sistema operativo e con le altre app, garantendo quindi una maggiore facilità d'uso per l'utente meno esperto, si è scelto di usare la libreria ufficiale di Google: Material Components. Per le **mappe**, si usa la libreria Mapbox, essendo Google Maps a pagamento. L'app prevede automaticamente una modalità tema chiaro e scuro in base alle impostazioni del dispositivo (salvo incompatibilità).
Compatibile da Android 5.0 in poi, il 94.1% dei dispositivi (fonte: Android Studio, 05/2020). Il codice sorgente è disponibile sulla pagina del Progetto (in inglese).

5.3 Linguaggi di programmazione

Per la realizzazione dell'app si utilizza il linguaggio **Java**, studiato durante l'A.S. 2018/2019 come parte del programma di 4^ superiore ITIS.
Per il backend è usato il linguaggio **PHP** per il programma server, studiato durante l'A.S. 2019/2020 come parte del programma di 5^ superiore informatica, e il linguaggio **SQL** per la gestione del database relazionale su cui si poggia il servizio.

5.4 Strumenti utilizzati

Per la realizzazione del client Android, si usa l'IDE (Integrated Development Environment) Android Studio.
Per le risorse grafiche, come icone, si è utilizzato il sito materialdesignicons.com che dà accesso a innumerevoli icone gratuitamente.
Per il programma server si è scelto di usare l'IDE PhpStorm, per permettere una scrittura del codice più veloce grazie alle funzionalità di auto completamento avanzate, reso disponibile in licenza gratuita di utilizzo a tutti gli studenti.
Per il controllo versione del codice sorgente di entrambe le parti software si utilizza lo strumento **Git**, e per la pubblicazione e condivisione del codice sorgente si sfrutta la piattaforma **GitHub**.

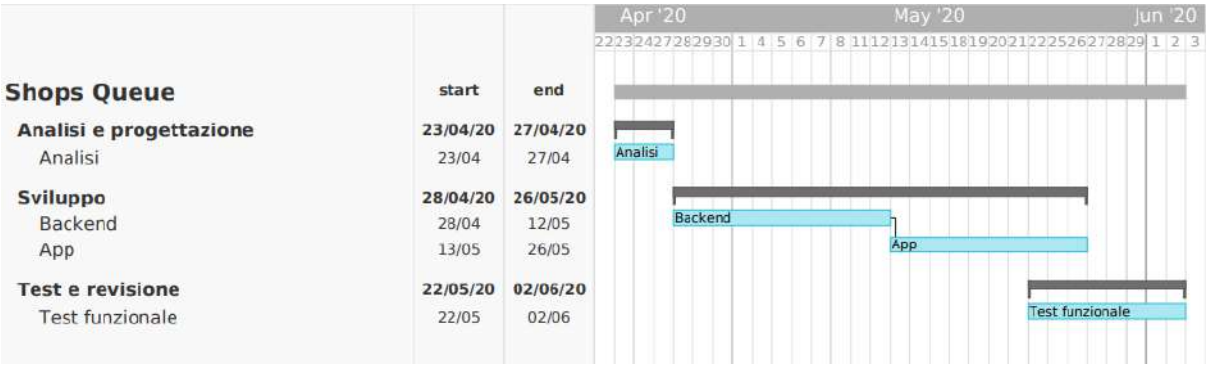
6 Analisi dei Tempi

Il Progetto ha il vincolo fondamentale di termine entro e non oltre il 10 giugno, esattamente 7 giorni prima dell'inizio degli esami orali.

6.1 Piano di lavoro

Il Piano di lavoro contiene il dettaglio delle fasi realizzative redatto in formato GANTT al fine di rappresentare i parallelismi e le dipendenze tra le diverse attività.

La pianificazione contiene l'indicazione delle macro-fasi in cui il Progetto verrà suddiviso e dei tempi massimi entro i quali dovrà avvenirne il completamento.



7 Analisi dei costi

"Non c'è una seconda occasione per dare la prima impressione" (Oscar Wilde)
Il lancio del prodotto, in questo caso di un determinato servizio, è fondamentale per dare fin da subito una prima impressione positiva.

7.1 Costo risorse umane

Si prevedono giornate di lavoro da 6 ore ciascuno.
Stando alla pianificazione formalizzata nel Gantt, si vede come la durata complessiva sia di 40 giorni, che diventano 32 giorni feriali. In totale si hanno a disposizione, quindi, 192 ore di lavoro.
Essendo i lavoratori impegnati nel progetto soltanto uno, si hanno 192 ore/uomo.
Considerando una retribuzione oraria di 25.00EUR/ora, si ha un costo delle risorse umane complessivo che ammonta a **4800.00€**.

7.2 Costi promozionali

È prevista una fase iniziale di promozione nei negozi personalmente, per cui i costi si considerano molto ridotti rispetto a campagne pubblicitarie su larga scala.
Per ammortizzare questi costi e altri necessari ove il passaparola del commerciante soddisfatto non sia sufficiente, si prevede di far pagare il servizio come abbonamento mensile dal costo da decidersi, in un secondo momento. In una prima fase rimane gratuito per permetterne la diffusione.
Si può affiancare da una campagna e-mail per raggiungere manager, imprenditori locali o anche piccoli alimentari. Molti servizi offrono campagne pubblicitarie gratuite per un periodo di tempo.

7.3 Costo infrastruttura

Per la messa in funzione del servizio sarà necessario noleggiare un server per permettere l'esecuzione del backend del sistema. Si prevede di appoggiarsi alla piattaforma di cloud **Microsoft Azure** per il noleggio di una VPS di discrete dimensioni. Per ridurre al massimo i costi di questa voce, si è deciso di sfruttare il **piano per studenti**, che prevede 100$ di bonus per tutti gli studenti maggiorenni, non solo universitari.
In base al preventivo visualizzabile nel dettaglio all'indirizzo https://azure.com/e/8b4d48ef994945248fb2754c87b803d7 (in fondo alla pagina), si vede come è possibile rientrare ampiamente nel credito.