UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA IN INGEGNERIA INFORMATICA

"App mobile per la fidelizzazione clienti"

Relazione sul tirocinio svolto presso la startup Unipiazza

Relatore: Prof. Mauro Migliardi

Laureando: Davide Colussi, 1189067

ANNO ACCADEMICO 2020-2021

Data di laurea: 20/09/2021

Indice

1. Introduzione

1.1. Descrizione dell'azienda

Unipiazza è una startup di Padova che si occupa di fidelizzazione clienti e digitalizzazione delle attività commerciali. È un servizio che premia i clienti che tornano nel punto vendita e lo fa in maniera coinvolgente. Infatti, attraverso l'accumulo di gettoni virtuali, i clienti possono ottenere dei prodotti omaggio e dei premi speciali da parte dell'attività commerciale.

In ogni attività commerciale convenzionata con Unipiazza è presente un tablet tramite il quale gli utenti possono registrarsi a Unipiazza (se non lo hanno già fatto tramite l'App smartphone) e cominciare a raccogliere gettoni.
La raccolta dei gettoni avviene tramite l'utilizzo di una tessera RFID, oppure del proprio smartphone.

Oltre a questo, Unipiazza offre alle attività commerciali convenzionate un servizio che consente di profilare i clienti, ottenendo statistiche sulle vendite, sul tasso di clienti che ritornano grazie a Unipiazza e altre informazioni utili.

Unipiazza fornisce anche, su richiesta, un servizio di e-commerce, tramite il quale le attività possono mettere in vendita online alcuni dei loro prodotti in modo facile e veloce.

Tutti questi servizi sono racchiusi in un gestionale tramite il quale i gestori delle attività commerciali possono personalizzarli a proprio piacimento.

I servizi di Unipiazza hanno un costo mensile per le attività commerciali convenzionate. Il costo comprende la fornitura ed eventuale manutenzione dei dispositivi per la raccolta gettoni, il servizio di profilazione clienti e l'e-commerce.

1.2. Scopo della tesi

La startup Unipiazza offre diversi servizi a clienti e gestori delle attività commerciali. L'obiettivo di questa tesi è analizzare il processo che ha portato alla creazione del servizio per la fidelizzazione clienti, partendo dall'analisi dei requisiti per poi proseguire con la realizzazione delle app.

Il capitolo "2. Analisi dei Requisiti" conterrà l'analisi dei requisiti necessari alla creazione del servizio, operazione svolta inizialmente dall'azienda. Tramite questo processo verranno raccolte tutte le informazioni utili a creare un servizio intuitivo e facile da utilizzare, con lo scopo di fidelizzare il cliente e supportare la crescita delle attività commerciali. Saranno analizzati lo scopo del servizio, l'interazione dell'utente finale, i requisiti prestazionali, hardware e software dei dispositivi impiegati.

Nel capitolo "3. Overview dell'architettura informatica" illustrerò come avvengono le comunicazioni tra i vari dispositivi impiegati, descrivendo l'architettura informatica che è stata implementata.

Nel capitolo "4. Realizzazione delle app" verrà illustrata la logica delle app realizzate per smartphone e tablet, con l'obiettivo di comprendere come avviene l'effettiva interazione dell'utente con il servizio. Verranno discusse alcune problematiche sorte in fase di sviluppo, legate alla comunicazione tra i dispositivi e la necessità di un continuo aggiornamento del servizio. Spiegherò come sono stati affrontati alcuni problemi e valuterò le diverse soluzioni che sono state adottate.

In questa tesi farò spesso riferimento a due entità: clienti e "business".
I clienti sono le persone che acquistano beni o servizi presso le attività commerciali utilizzando il sistema di fidelizzazione di Unipiazza.
Per "business" intendo le attività commerciali che dispongono del servizio di fidelizzazione fornito da Unipiazza; i gestori di queste attività dovranno anch'essi interagire con il servizio di fidelizzazione per garantirne il funzionamento.
Il sistema di fidelizzazione verrà implementato tramite tre app di interesse:

- app installata negli smartphone lato clienti
- app installata nei tablet lato "business"
- app installata negli smartphone lato "business"

2. Analisi dei requisiti

2.1. Perché è necessaria l'analisi dei requisiti

L'analisi dei requisiti serve a esplicitare puntualmente quali sono le funzionalità che il prodotto finale (nel nostro caso un servizio) deve avere e a quali vincoli deve sottostare.
É importante soffermarsi su questa fase di progettazione, infatti sarà la base per l'intera realizzazione del progetto. Un'eventuale mancanza di attenzione nell'analisi dei requisiti potrebbe portare a situazioni in cui l'intero progetto viene realizzato ignorando alcuni vincoli importanti; in una situazione del genere è problematico tornare indietro, poiché molte risorse sono già state impiegate e correggere eventuali mancanze significherebbe averle sprecate.

Dato che l'intero progetto è realizzato da una startup, l'analisi dei requisiti iniziale si concentrerà sulla realizzazione di un servizio funzionante da subito. Infatti, in generale, in una startup, è utile realizzare il proprio prodotto/servizio in maniera veloce, per valutare immediatamente la sua efficacia sul mercato.
Per questo motivo, non mi soffermerò troppo sui dettagli, ma solo sull'individuazione dei requisiti principali.
Durante la realizzazione del servizio emergeranno requisiti più specifici, questi potranno venir affrontati in un momento successivo.

2.2. Scopo del prodotto

La startup Unipiazza nasce con l'intento di offrire un servizio di fidelizzazione clienti alle piccole attività commerciali. Il servizio offerto è facile da utilizzare, in questo modo le attività potranno adottarlo senza dover imparare procedure complicate.
Le attività a cui Unipiazza si rivolge sono quelle che non stanno sfruttando i vantaggi che la fidelizzazione clienti può offrire.
Sul mercato attuale esistono aziende che offrono servizi simili, ciò che contraddistingue Unipiazza è l'approccio "user-centric", nel quale la priorità è l'utente finale.

Nel servizio di fidelizzazione verrà utilizzata la "gamification", verranno cioè inserite delle meccaniche ludiche per aumentare il coinvolgimento degli utenti nel servizio.
Quest'ultimo aspetto sarà un punto chiave nello sviluppo dell'App. Il CEO di Unipiazza, Edoardo Parisi, dispone di una particolare esperienza nel settore "gamification", perciò il servizio sarà maggiormente curato su questo aspetto, rispetto ad altre app già esistenti.

2.3. Casi d'uso

2.3.1. Interazione diretta tra cliente e attività commerciale

Quando un cliente acquista un bene presso un'attività commerciale, scansiona una tessera sul tablet dell'attività, in questo modo può accedere al servizio Unipiazza. Il tablet mostrerà al cliente i gettoni che ha accumulato presso quella attività, permettendo al gestore di aggiungerne altri, a seconda di quanto il cliente ha speso.

Per aggiungere gettoni al cliente, il gestore deve poter utilizzare uno smartphone dedicato, tramite il quale può decidere manualmente la quantità di gettoni da aggiungere.

Più il cliente spende, più il numero di gettoni aumenta. Il cliente può visualizzare nel tablet i premi disponibili presso l'attività (caffè omaggio, brioche gratis, ecc). Una volta raggiunto il numero di gettoni necessario per ritirare il premio, il cliente può decidere se ritirarlo.

Il cliente è invogliato a ritornare, poiché possiede un certo saldo di gettoni presso quella attività commerciale (che saranno validi solamente in quell'attività).

Il cliente che non dispone della tessera da scansionare potrà ritirarla gratuitamente in un qualsiasi attività commerciale convenzionata con Unipiazza.

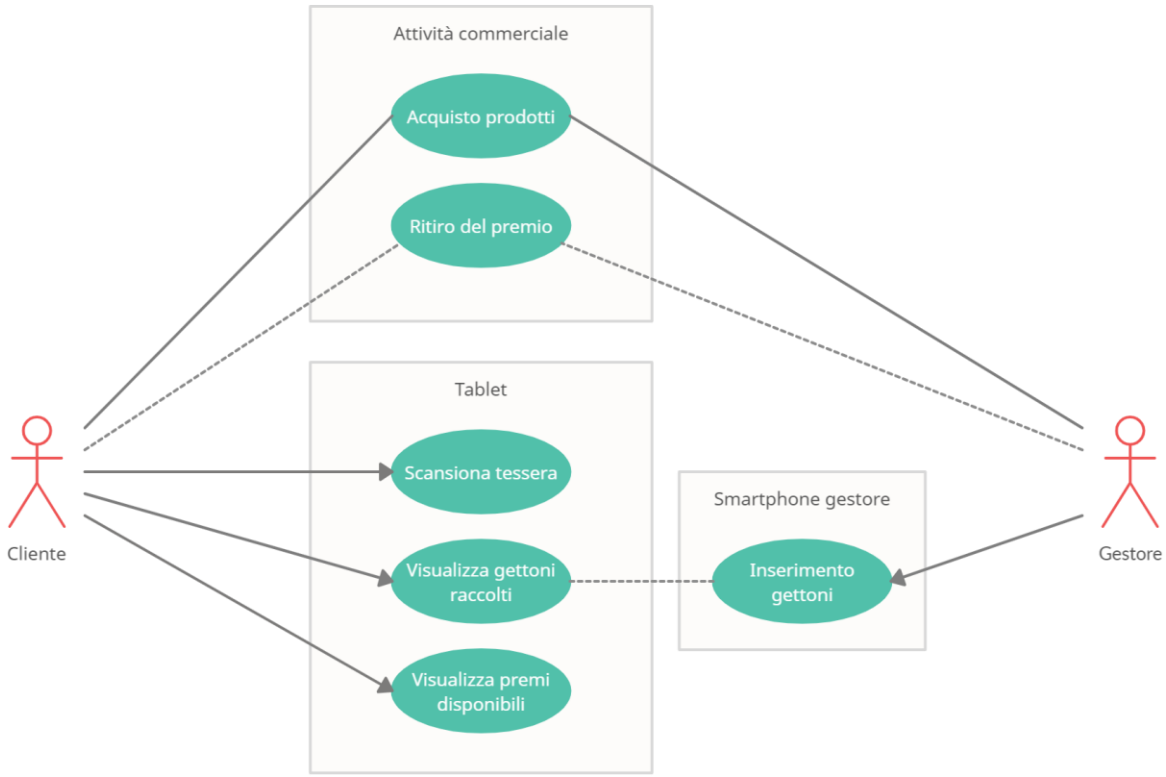

Figura 2.3.1.1: interazione diretta tra cliente e attività commerciale

2.3.2. Interazione indiretta tra cliente e attività commerciale

Il cliente può visualizzare sul proprio smartphone le attività commerciali convenzionate con Unipiazza vicino a lui. Il cliente può "seguire" certe attività, in questo modo potrà ritrovarle nella lista "attività seguite" dell'app smartphone. Il cliente può visualizzare in ogni momento il saldo dei gettoni raccolti in ogni attività.

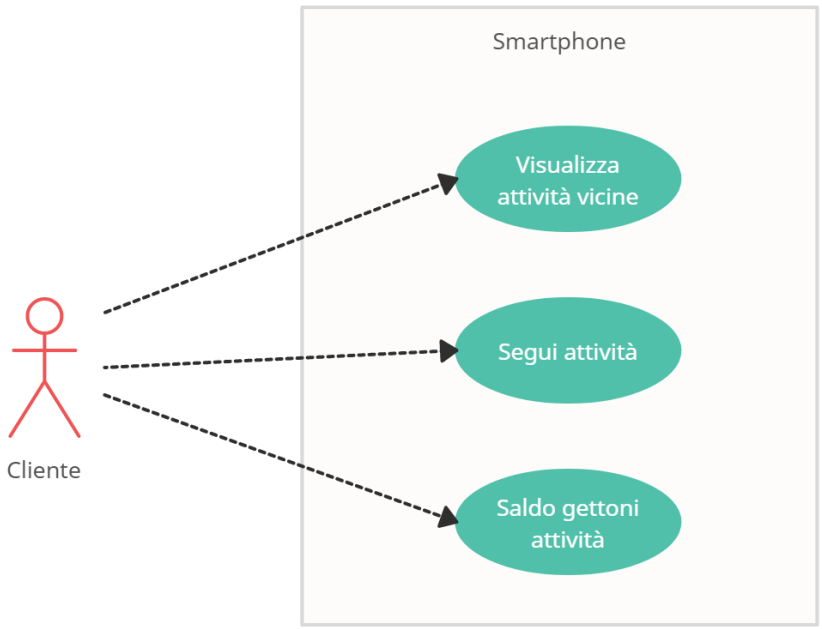

Figura 2.3.2.1: interazione indiretta tra cliente e attività commerciale

2.4. Requisiti prestazionali

Flusso di esecuzione dei casi d'uso:

- l'esperienza utente dev'essere la priorità: il flusso di esecuzione per ogni caso d'uso dev'essere rapido e di facile comprensione per i clienti e i gestori;
- implementazione di meccaniche di "gamification" per aumentare il coinvolgimento.

Gestione di anomalie:

- se l'utente può risolvere il problema da solo, dev'essere guidato nelle possibili soluzioni in modo che non si senta "abbandonato";
- per problemi legati ai dispositivi presenti nelle attività (tablet e smartphone), un incaricato di Unipiazza dovrà eventualmente recarsi sul posto per fornire aiuto al gestore.

2.5. Requisiti hardware

Tablet per le attività:

- devono essere grandi abbastanza per consentire ad ogni tipo di utente di utilizzare il servizio senza sforzo;
- deve disporre di touch screen;
- deve disporre di NFC per poter consentire la scansione delle tessere RFID;
- deve disporre di Wi-Fi;
- deve poter rimanere acceso per tutta la giornata lavorativa delle attività commerciali (rimanendo costantemente in carica);
- deve poter essere configurato dal personale tecnico di Unipiazza con facilità prima di essere consegnato per la prima volta ai gestori delle attività;
- la manutenzione non deve essere complicata.

Smartphone per le attività:

- devono essere grandi abbastanza per consentire ad ogni tipo di utente di utilizzare il servizio senza sforzo (Display di almeno 3,5 pollici);
- deve disporre di touch screen;
- deve disporre di Wi-Fi;
- deve poter rimanere acceso per tutta la giornata lavorativa delle attività commerciali (rimanendo costantemente in carica).

Smartphone dei clienti:

- deve disporre di touch screen;
- deve disporre di Wi-Fi o rete mobile.

Tessere:

- devono utilizzare la tecnologia RFID;
- devono poter essere personalizzate esteticamente con il logo di Unipiazza;
- devono avere dimensioni simili a una carta di credito per essere portabili.

Attività commerciali:

- Devono disporre di una rete Wi-Fi connessa a Internet.

2.6. Requisiti software

Applicazione smartphone per clienti:

- deve poter essere eseguita su sistemi Android e iOS, con una interfaccia quanto più possibile coerente a prescindere del sistema operativo utilizzato;
- deve poter essere scaricata e aggiornata dal "Play Store" per Android e dall' "App Store" per iOS;
- deve essere di facile utilizzo per i clienti.

Tablet per le attività:

- l'app deve poter essere aggiornata da remoto, senza che il personale di Unipiazza debba recarsi presso le attività commerciali;
- l'app deve essere di facile utilizzo per i clienti;
- la preparazione del sistema operativo per la prima volta (rimozione di app non necessarie, inserimento delle app di Unipiazza) non deve essere un processo complicato;
- il sistema operativo deve poter essere reimpostato alle condizioni iniziali con facilità.

Smartphone per le attività:

- l'app deve poter essere aggiornata da remoto, senza che il personale di Unipiazza debba recarsi presso le attività commerciali;
- l'app deve essere di facile utilizzo per i gestori;
- il sistema operativo deve poter essere reimpostato alle condizioni iniziali con facilità.

2.7. Metodo di sviluppo

Non è stata impiegata una metodologia di sviluppo fissa. Il motivo nasce dal fatto che l'obiettivo di Unipiazza è stato quello di realizzare dapprima un servizio funzionante, in seguito, dopo averlo consolidato, la gestione del progetto è stata organizzata in maniera più precisa.
Il servizio è stato testato sul campo non appena le sue funzionalità base erano pronte. In questo modo, eventuali problematiche sono emerse subito e sono state risolte già nelle prime fasi di sviluppo.

La realizzazione degli obiettivi è stata conseguita dall'intero team di Unipiazza, suddividendo lo sviluppo in diverse "task", ossia piccole unità di lavoro facilmente suddivisibili tra il personale.

2.8. Validazione requisiti

Il progetto non nasce in risposta a una richiesta di un particolare committente, infatti, i requisiti sono stati raccolti ipotizzando gli interessi dei clienti e dei gestori delle attività commerciali.

In questa situazione, alcune metodologie specifiche possono essere utilizzate per la validazione dei requisiti, anche dopo che il servizio è già stato lanciato sul mercato, per un miglioramento dello stesso.

Ecco alcune delle soluzioni possibili per ricevere un feedback da parte degli utenti del servizio:

- opinioni dirette dei clienti e gestori delle attività commerciali;
- recensioni su “Play Store” o “App Store”;
- realizzazione di un portale web per la raccolta di suggerimenti;
- realizzazione di statistiche automatiche sull’utilizzo del servizio.

3. Overview dell'architettura informatica

Per creare il servizio Unipiazza serviva un'architettura informatica per la gestione dei dati e del funzionamento del servizio. Per esempio, era necessario memorizzare il numero di gettoni raccolti da ogni cliente, i nomi delle attività commerciali presenti sul territorio e le transazioni effettuate da ogni cliente.

Per realizzare un servizio di questo tipo, si è preferito procedere secondo un'architettura informatica il più standard possibile, in modo da semplificare lo sviluppo e limitare possibili problematiche.

La descrizione della struttura che si è preferito utilizzare viene illustrata nello schema in *figura 3.1*.

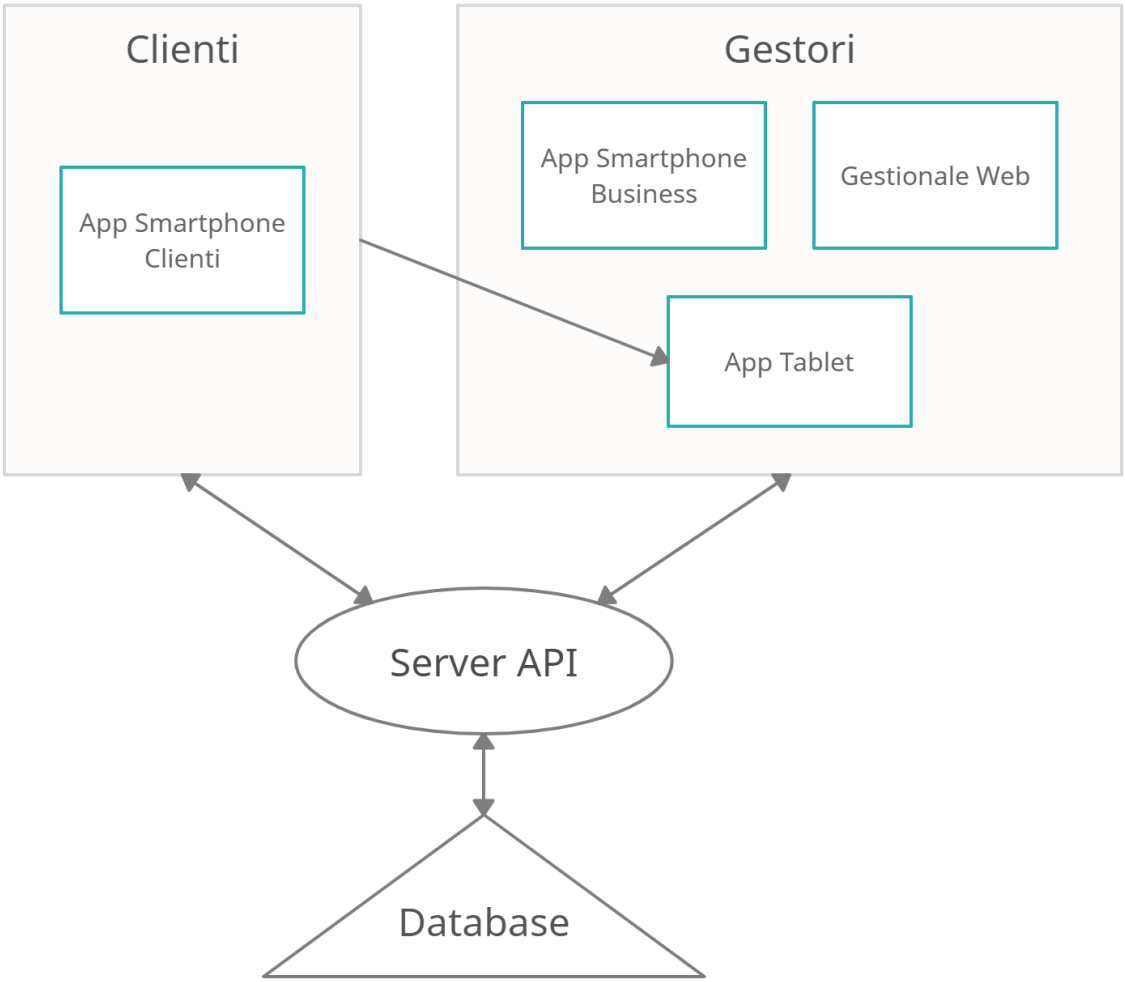

Figura 3.1: struttura informatica utilizzata per la gestione del servizio di Unipiazza

In questo schema compaiono le app e il gestionale, le entità nelle quali risiede la gestione dell'interfaccia utente, ossia la componente del servizio che permette ai clienti e ai gestori di interagire con il servizio di Unipiazza.

Il database è l'entità adibita al salvataggio e il mantenimento dei dati in maniera strutturata relativi alla gestione del servizio di fidelizzazione (come, ad esempio, il numero di gettoni raccolti per ogni cliente, i nomi delle attività commerciali, i nomi degli utenti…).

Le applicazioni comunicano con il database tramite un Server API: un'entità spesso utilizzata nelle architetture informatiche, il cui scopo è gestire la comunicazione tra due o più entità (nel nostro caso le app, il gestionale e il database), fornendo un'interfaccia ben definita per il dialogo.
Nel nostro caso, i client che vogliono richiedere delle informazioni al database, dovranno fare delle richieste "http" al server API, che farà da intermediario. Il server si occuperà di risolvere le richieste secondo una determinata logica, di interrogare il database e fornire una risposta al client.

3.1. Microservizi vs architettura monolitica

Per implementare questo tipo di struttura, esistono principalmente due approcci: quello orientato ai microservizi e quello monolitico.
In questo paragrafo illustrerò le differenze tra questi due approcci e discuterò quale sia la scelta migliore nel nostro dominio di applicazione.

3.1.1. I microservizi

I microservizi sono un approccio per sviluppare e organizzare l'architettura informatica nel quale vi è una scomposizione del servizio principale in servizi indipendenti più piccoli, chiamati microservizi, che comunicano tra di loro tramite API ben definite.

Ogni microservizio esegue una sola funzione e, poiché eseguito in modo indipendente, può essere aggiornato, distribuito e ridimensionato per rispondere alla richiesta di funzioni specifiche del servizio principale.

Spesso questo tipo di architettura utilizza un approccio cloud-based, dove i microservizi sono situati sul cloud, rendendoli così disponibili attraverso Internet, tramite un'architettura distribuita.

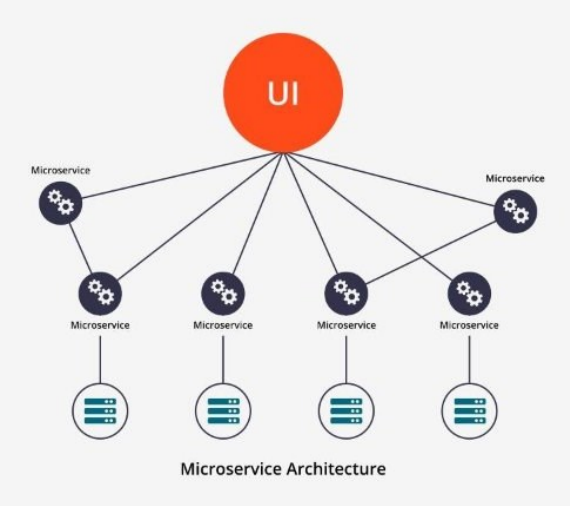

Figura 3.1.1.1:architettura orientata ai microservizi

3.1.2. Architettura monolitica

Nelle architetture monolitiche, tutti i processi del servizio sono strettamente collegati fra loro e sono contenuti all'interno di un unico blocco (detto "monolite").
L'architettura monolitica, essendo una delle più semplici, è stata una delle prime tipologie di architetture informatiche.

Tramite questo approccio, l'interfaccia utente, la logica del servizio e l'accesso ai dati sono racchiusi tutti all'interno di un'unica entità. Se fosse necessario aumentare le prestazioni del servizio, occorrerebbe intervenire sul blocco, migliorando le sue caratteristiche o replicandolo più volte.

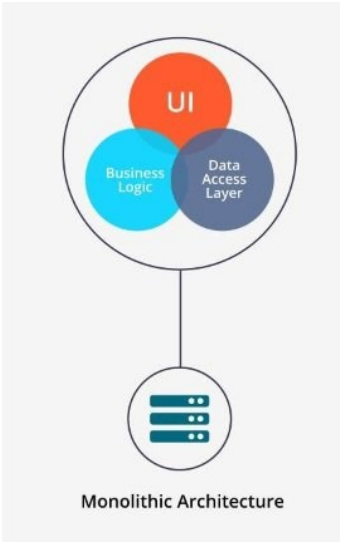

Figura 3.1.2.1: architettura monolitica

3.1.3. Il confronto

L'architettura monolitica si presta bene alla realizzazione di piccoli servizi poco soggetti ai cambiamenti. In queste condizioni, questo tipo di architettura è la più facile da implementare, poiché è possibile concentrare l'intero sistema in un'unica entità.

Le architetture orientate ai microservizi, però, permettono di scalare e sviluppare le applicazioni in modo più rapido e semplice. Inoltre, delegando ad entità distinte compiti ben definiti, ogni compito può essere svolto con maggiore efficienza e in maniera indipendente, diminuendo eventuali "single point of failure", ossia punti dell'infrastruttura su cui regge l'intero servizio, i quali, se dovessero presentare malfunzionamenti, minerebbero il funzionamento del servizio principale.

La prospettiva è che, durante lo sviluppo del servizio di fidelizzazione di Unipiazza, probabilmente emergeranno nuovi requisiti, rendendo necessaria qualche modifica all'architettura informatica. É utile quindi implementare un'architettura che sia flessibile nel tempo, per questo l'approccio ai microservizi è preferibile.

In futuro, la prospettiva è che il servizio di Unipiazza sia utilizzato da sempre più utenti, è necessario quindi utilizzare un approccio che permetta una scalabilità del servizio. Nell'approccio orientato ai microservizi, è possibile ridimensionare ogni microservizio in base alle necessità, permettendo ad un maggior numero di utenti di usufruire del servizio contemporaneamente.
Utilizzando l'approccio monolitico invece, se volessimo aumentare le performance di una singola componente del servizio, saremmo costretti a ridimensionare l'intero blocco, ciò comporterebbe un aumento dei costi.

Per le motivazioni sopra citate, è stato scelto di utilizzare un'architettura informatica orientata ai microservizi per la realizzazione del servizio di fidelizzazione di Unipiazza.

3.2. Implementazione dell'architettura informatica

In questa sezione descriverò come è stata implementata l'architettura informatica utilizzata per la realizzazione del servizio di fidelizzazione di Unipiazza.

Dal momento che molte delle scelte implementative sono state fatte in base alle conoscenze pratiche degli sviluppatori all'interno dell'azienda, con una preferenza verso l'impiego di soluzioni standard, non entrerò nel merito di quali potevano essere le soluzioni alternative.
Mi limiterò a descrivere le varie entità coinvolte nella soluzione adottata, al fine di comprendere meglio l'architettura informatica sulla quale dovranno operare le applicazioni sviluppate per smartphone e tablet.

Nella *figura 3.2.1,* presente alla pagina successiva, viene illustrata l'implementazione dell'architettura informatica del servizio di fidelizzazione di Unipiazza. Di seguito spiegherò le varie entità che compongono lo schema.

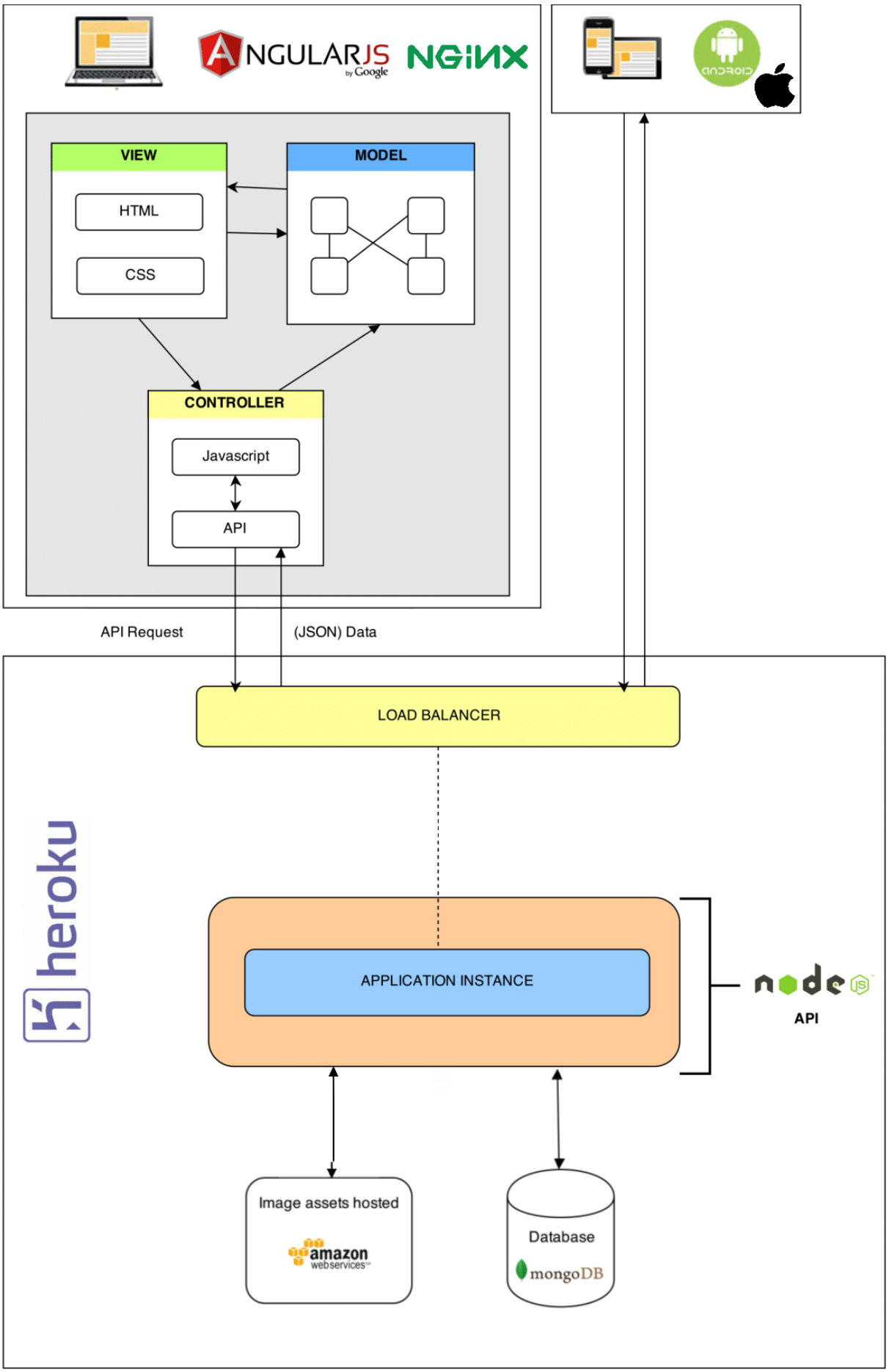

Figura 3.2.: implementazione dell'architettura informatica di Unipiazza

3.2.1. I client

I client che si affacciano al servizio di fidelizzazione di Unipiazza sono rispettivamente:

- Smartphone clienti
- Smartphone gestori
- Tablet
- Gestionale web per i gestori delle attività e gli amministratori di Unipiazza

Gli smartphone dei gestori e i tablet utilizzano Android come sistema operativo.
Gli smartphone dei clienti possono utilizzare sia Android che iOS come sistema operativo.
Tramite il gestionale, i gestori possono controllare le impostazioni del servizio specifiche per il la loro attività commerciale.
Gli amministratori di Unipiazza possono utilizzare un altro gestionale web per avere un controllo su tutto il servizio di fidelizzazione.

Il gestionale è implementato tramite "AngularJS": un framework per le applicazioni web open source. Questo framework è stato scelto poiché semplifica lo sviluppo utilizzando l'architettura "model-view-controller"; il "model" rappresenta l'entità adibita al salvataggio dei dati utili all'applicazione, il "view" si occupa di gestire la vista dei dati per l'utente e il "controller" ha il compito di rispondere agli input dell'utente ed eseguire di conseguenza operazioni che vadano a modificare i dati contenuti nel "model". Il "controller" ha anche il compito di richiedere al "view" la visualizzazione di certi dati, in determinati istanti.

NGINX è il server web che si occupa di fornire l'accesso al gestionale, infatti ospita le pagine "HTML", i file "CSS" e i file "Javascript". Questo server web è stato scelto poiché fornisce un bilanciamento del carico, migliorando quindi le prestazioni nel caso di accessi multipli contemporanei al gestionale.

I vari client comunicano con il server effettuando delle chiamate API, ricevendo come risposta dei documenti "JSON" che contengono i dati richiesti.

3.2.2. Il server API

Il server API è basato su “Node.js”: un runtime “Javascript” guidato da eventi asincroni con la caratteristica di essere facilmente scalabile e utilizzabile per applicazioni real time.
Questo server si occupa principalmente di ricevere le chiamate API provenienti dai client, processarle, interrogare il database e inviare le risposte ai client.
Questo server fa quindi parte del “back end” dell'applicazione di fidelizzazione clienti, essendo l’entità adibita alla logica per il salvataggio e il recupero dei dati dal database.

In questo server vengono definite le API da utilizzare; i client dovranno quindi conoscere la loro grammatica per poterle utilizzare. Inoltre, per poter utilizzare le API private di Unipiazza, i client si devono autenticare. Non entrerò nei dettagli che riguardano il meccanismo di autenticazione, è utile però sapere che sono stati previsti tre tipi di utente autenticato:

- Cliente
- Gestore
- Admin Unipiazza

3.2.3. MongoDB

MongoDB è un database non relazionale “NoSQL” orientato a documenti; non si basa perciò su una classica struttura a tabelle (tipica dei database relazionali), bensì utilizza documenti “JSON” per la rappresentazione dei dati. In questo modo, l’integrazione con alcuni tipi di applicazioni è più facile e veloce.

MongoDB ha il vantaggio di offrire un replica set, ossia una copia identica dei dati all’interno del database, in modo da aumentare la disponibilità e gestire meglio il carico.

MongoDB offre la possibilità di eseguire query (interrogazioni) ad hoc, consentendo ricerche per campi, intervalli e regular expression. Le query possono restituire parti specifiche del documento o funzioni definite dall’utente in “Javascript”.

3.2.4. Amazon Web Services

Amazon Web Service è una delle più famose piattaforme di cloud computing on demand. Nella nostra specifica architettura è stata scelta come entità adibita al salvataggio delle risorse “RAW”, ossia tutte quelle risorse di tipo multimediale come immagini, audio, video, file, di

dimensione non trascurabile. Nella nostra applicazione, i client hanno necessità di accedere a questi tipi di risorse tramite la rete internet; AWS è stata decretata una delle soluzioni migliori per gestire questo tipo di comunicazione.

Inoltre, AWS è stato anche utilizzato per contenere i file “apk”, ossia i file che dovranno essere scaricati sullo smartphone dei gestori e il tablet, per aggiornare da remoto l’app di Unipiazza presente al loro interno.

3.2.5. Heroku

Heroku è una piattaforma cloud altamente scalabile per lo sviluppo e il monitoraggio di applicazioni. Nel nostro caso è stata utilizzata come “involucro” per tutto il “back end” dell’applicazione di fidelizzazione clienti Unipiazza.

Heroku si occupa in automatico della gestione delle richieste “http” in arrivo, offrendo un bilanciamento del carico per offrire migliori prestazioni al sistema, anche in caso di molte richieste contemporanee.

4. Realizzazione delle app

In questo capitolo descriverò come sono state realizzate le app lato “business” e lato clienti. L’obiettivo principale è comprendere il funzionamento e la logica sulle quali esse operano e interagiscono tra loro, descrivendo i punti salienti e le problematiche emerse.

4.1. Descrizione del funzionamento dell'app clienti

Nel mercato attuale che riguarda gli smartphone, la prevalenza dei dispositivi utilizza Android o iOS come sistema operativo. Per questo motivo si è deciso di sviluppare l’app smartphone in queste due piattaforme.

Durante lo sviluppo dell’app sono state progettate le varie “activity”, ossia le interfacce che l’utente visualizza e tramite le quali può interagire.
Di seguito vengono mostrate le “activity” dell’app smartphone.

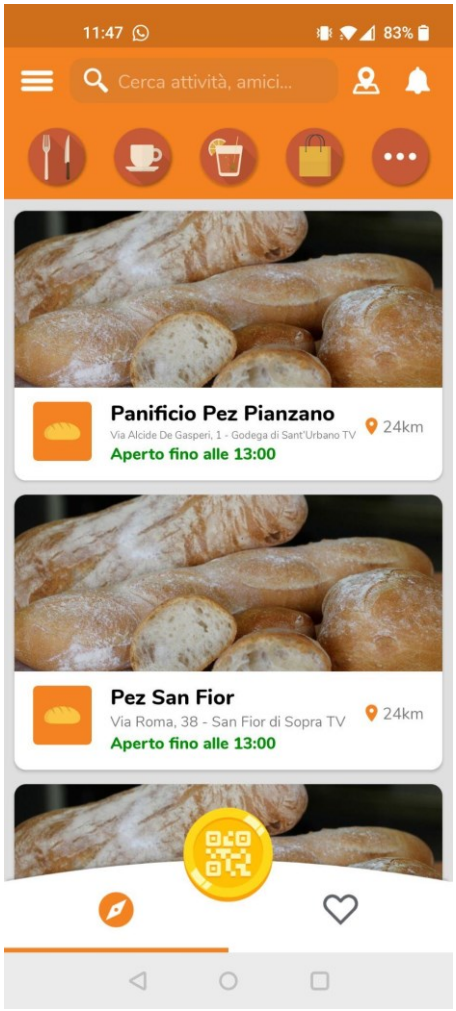

Figura 4.1.1

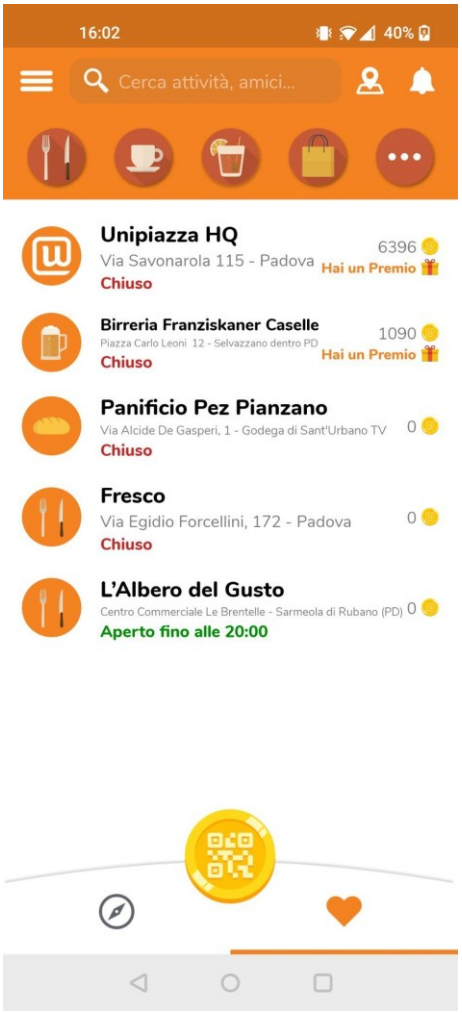

Figura 4.1.2

Tramite l'"activity" in *figura 4.1.1* è possibile visualizzare tutte le attività commerciali convenzionate con Unipiazza vicino all'utente. Le attività sono ordinate dalla più vicina alla più lontana rispetto all'utente. In alto è presente una barra di ricerca e dei bottoni tramite i quali si possono ricercare delle specifiche attività commerciali (ristoranti, bar, negozi, …).

Cliccando sull'icona a forma di cuore in basso, viene visualizzata l'"activity" in *figura 4.1.2*, tramite la quale possono essere visualizzate le attività commerciali seguite dal cliente, con il relativo indirizzo, categoria, orario di apertura, numero di gettoni raccolti e premi disponibili.
Cliccando sull'icona della mappa in alto a destra, si possono visualizzare le attività commerciali sulla mappa.
Cliccando il bottone giallo in basso, è possibile scansionare il QR code di un'attività commerciale, in modo da seguirla e cominciare a raccogliere gettoni.

Figura 4.1.3

Cliccando su un'attività, si apre l'"activity" in *figura 4.1.3,* tramite la quale possono essere visualizzati il numero di gettoni raccolti, i premi e il relativo costo in gettoni. Tramite questa activity si può decidere di seguire l'attività, in questo modo l'utente potrà raccogliere gettoni, ricevere promozioni e aggiornamenti.
Man mano che il cliente raccoglierà gettoni presso l'attività commerciale, le barre di progresso relative ai premi aumenteranno e, una volta che saranno riempite, il cliente potrà ritirare il premio recandosi presso l'attività.

Dall'activity in *figura 4.1.3* è anche possibile ottenere il numero di telefono dell'attività commerciale, condividerla con gli amici, visitare la pagina Facebook e ottenere indicazioni stradali.

4.2. Descrizione del funzionamento delle app "business"

Per i tablet e gli smartphone lato "business" si è scelto di sviluppare le app in Android poiché la piattaforma è ricca di documentazione e offre molte librerie utili per la gestione dei componenti hardware del dispositivo (Wi-Fi, NFC, Bluetooth, …).
Inoltre, dovendo rispettare i requisiti hardware e software citati nel capitolo "2. Analisi dei Requisiti", sarà più facile trovare dei dispositivi che rispettano queste specifiche dato che esistono moltissimi dispositivi Android in commercio.

Durante lo sviluppo dell'app tablet e smartphone gestori sono state progettate le varie "activity". Di seguito vengono mostrate le "activity" dell'app tablet.

Figura 4.2.1

Figura 4.2.2

Come si può osservare, nell' "activity" in *figura 4.2.1* è presente il nome dell'attività commerciale. Da questa "activity" il cliente, scansionando la propria tessera RFID sul tablet, oppure scansionando con il proprio smartphone il codice QR code che compare premendo "Raccogli gettoni" in *figura 4.2.2*, può accedere al sistema Unipiazza.

Figura 4.2.3

Figura 4.2.4

La prima volta che il cliente scansiona una tessera RFID, dovrà associarla a sé stesso. Per farlo dovrà inserire il proprio nome, cognome e la sua e-mail nel tablet tramite l'"activity" in *figura 4.2.3*.

Una volta che la tessera è stata associata al cliente, ogni qual volta verrà scansionata su un tablet, permetterà di accedere al sistema Unipiazza, facendo comparire la schermata in *figura 4.2.4*, tramite la quale il cliente potrà visualizzare il numero di gettoni raccolti e i premi disponibili presso l'attività.

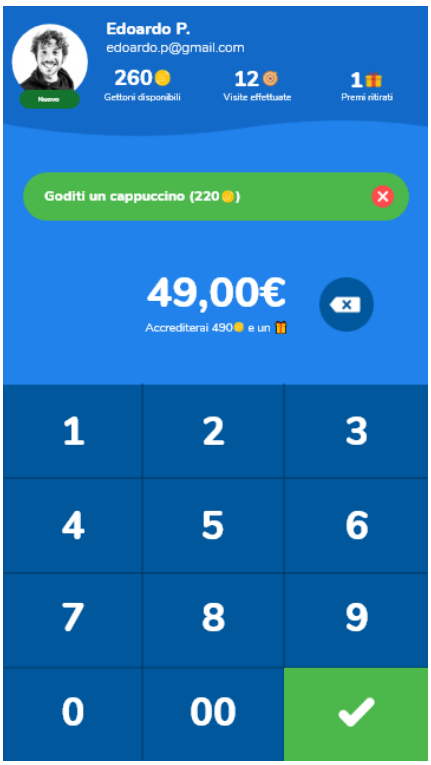

Figura 4.2.5

Nel momento in cui sul tablet è visualizzata l'"activity" in *figura 4.2.4*, sullo smartphone del gestore sarà visualizzata l'"activity" in *figura 4.2.5*, tramite la quale il gestore può visualizzare il nome del cliente e accreditargli i gettoni, inserendo i soldi che ha speso presso l'attività commerciale.
Se un cliente decide di ritirare un premio, il gestore viene informato tramite l'app smartphone del gestore e dovrà quindi consegnare il premio scelto dal cliente.

I gestori possono collegare la loro pagina Facebook a Unipiazza, in questo modo ogni cliente, se è iscritto a Facebook, può decidere di mettere "mi piace" alla pagina, guadagnando gettoni extra.

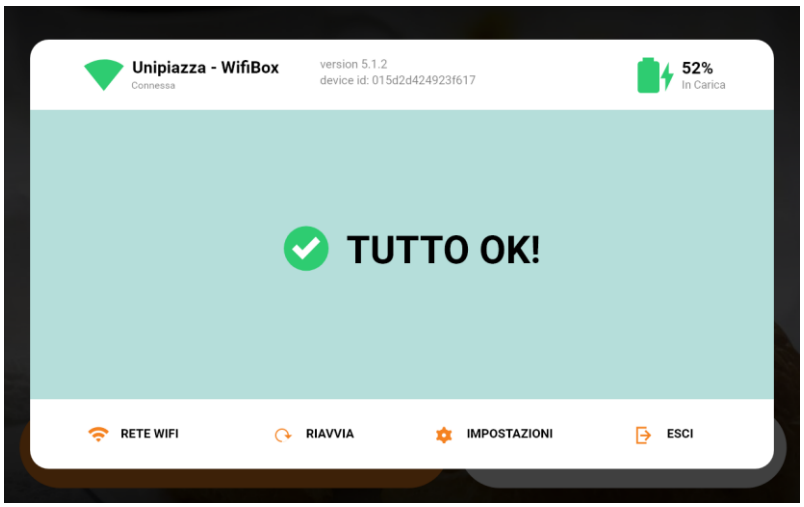

Figura 4.2.6

É presente un'"activity" che mostra lo stato del tablet: rete Wi-Fi, stato batteria, versione dell'app, ID dispositivo (*figura 4.2.6*).
Questa "activity" è utile per rilevare eventuali problemi nella comunicazione del tablet con il sistema di Unipiazza. In basso sono presenti alcuni bottoni per aprire le impostazioni del tablet o per riavviarlo.
Questa activity è accessibile tenendo premuto per 4 secondi una parte qualsiasi dello schermo.

4.3. La scelta dei dispositivi

I dispositivi da utilizzare nelle attività commerciali (tablet e smartphone) sono stati scelti in modo da rispettare i requisiti analizzati nel capitolo "2. Analisi dei Requisiti".

Dapprima, si è scelto di utilizzare il tablet Nexus 7, un tablet con display da 7 pollici basato su Android, ideato da Google e prodotto da ASUS, immesso nel mercato a partire da agosto 2013. Il tablet dispone di Wi-Fi e di uno schermo grande abbastanza per permettere un utilizzo facile. Il tablet ha un chip NFC, situato nella parte posteriore del tablet, ma utilizzabile anche se una tessera RFID viene scansionata sulla parte anteriore. Un vantaggio di questo dispositivo è che è possibile ottenere l'accesso "root", ossia l'accesso al sistema in maniera "amministratore", in pochi semplici passi.
Un altro vantaggio è che la documentazione riguardante la manutenzione o sostituzione di parti del tablet è molto ricca.

Questo dispositivo nel corso del tempo è stato però sostituito da un altro: Hipo M8 Pro.
Questo tipo di tablet è stato ritenuto migliore sotto certi aspetti rispetto al Nexus 7, per esempio, presenta il sensore NFC nella parte anteriore del dispositivo, consentendo una affidabilità maggiore nella lettura delle tessere RFID. Questo tablet ha uno schermo di 8 pollici, consentendo una leggibilità maggiore del contenuto. Anche in questo dispositivo è possibile ottenere l'accesso "root" in maniera facile. Inoltre, si tratta di un tablet molto più recente, perciò il supporto della casa madre è maggiormente garantito.

Per lo smartphone dei gestori delle attività è stato adottato l'Alcatel Pixi 4. Lo smartphone è stato scelto poiché il costo è basso, dispone di un display di almeno 3,5 pollici e ha il Wi-Fi.

4.4. Comunicazione tra tablet e smartphone gestori

Il tablet dell'attività commerciale e lo smartphone per il gestore devono scambiarsi delle informazioni. Per esempio, non appena un cliente scansiona la propria tessera RFID sul tablet, lo smartphone del gestore deve far visualizzare il nome del cliente. Inoltre, quando il gestore vuole accreditare i gettoni al cliente, lo smartphone del gestore deve comunicare con il tablet in modo che il cliente visualizzi quanti gettoni gli sono stati accreditati.
I dati da trasferire sono pochi, ma è comunque necessario che arrivino integri, in maniera affidabile e in breve tempo. Per realizzare questo tipo di comunicazione, sono state ideate e testate due soluzioni, che ora descriverò.

4.4.1. Comunicazione tramite Bluetooth

La prima soluzione è stata quella di utilizzare il Bluetooth, ossia uno standard di trasmissione dati wireless a corto raggio. Per comunicare utilizzando il Bluetooth, i due dispositivi devono eseguire un "pairing", ossia un accoppiamento da fare manualmente, obbligatorio solo la prima volta. Dopo l'accoppiamento, i dispositivi possono comunicare tra loro, con l'unico requisito che la scheda Bluetooth di entrambi sia accesa e che i dispositivi si trovino a distanza ravvicinata.
Per utilizzare il Bluetooth come canale di comunicazione, si è dovuta progettare e implementare un'interfaccia di comunicazione in modo che i dispositivi potessero scambiarsi i dati in maniera strutturata.

Questa soluzione, purtroppo, è risultata problematica, perché nel modello di tablet utilizzato dapprima nelle attività commerciali, ossia il Nexus 7, l'antenna della scheda Bluetooth e della scheda Wi-Fi è in comune. Questo causava delle interferenze nella comunicazione ogni qual volta il Bluetooth e il Wi-Fi venissero utilizzati nello stesso momento.
Il fatto che lo smartphone e il tablet, oltre a dover comunicare tra loro, devono anche comunicare con il server di Unipiazza utilizzando il Wi-Fi, è stato determinante nella scelta di una soluzione alternativa che non comprendesse l'utilizzo del Bluetooth.

4.4.2. Comunicazione tramite Ably

Ably è un PaaS (Platform as a Service) per la comunicazione real time tra più dispositivi basata sull'utilizzo del design pattern "publish/subscribe".

Specificatamente alle comunicazioni tra i dispositivi, Ably offre diverse feature:

- disaccoppiamento tra chi invia e chi riceve i messaggi;
- i messaggi vengono salvati in una cronologia persistente;
- i dispositivi possono comunicare il loro stato (online, offline, ...).

4.4.2.1. Design pattern "publish/subscribe"

Per poter utilizzare il servizio di comunicazione real time di Ably, i dispositivi devono prima autenticarsi. Una volta autenticati possono agire da "publisher" e/o da "subscriber". I "publisher" pubblicano informazioni all'interno di un canale identificato da un nome. I "subscriber" si iscrivono ad un determinato canale, ricevendo gli aggiornamenti pubblicati dai "publisher" in quel canale.

Utilizzando questo design pattern, vi è un disaccoppiamento tra chi pubblica le informazioni e chi le riceve; infatti, non è necessario che i "publisher" conoscano i "subscriber" e viceversa: è la piattaforma Ably che si occupa di far arrivare i messaggi pubblicati da un "publisher" a tutti i "subscriber" iscritti a un determinato canale.

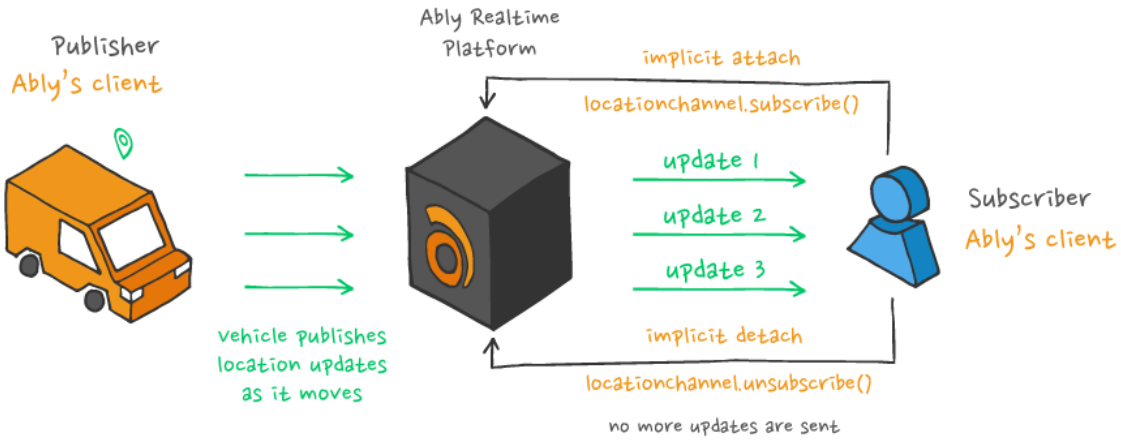

Figura 4.4.2.1.1: design pattern "publish/subscribe" di Ably

4.4.2.2. Impiego di Ably nella comunicazione tra smartphone e tablet

La prima volta che i due dispositivi vengono affidati ad una attività commerciale, deve avvenire un'associazione tra smartphone e tablet, in questo modo il tablet può conoscere quale sia lo smartphone con il quale comunicare e viceversa.

Per l'associazione tra i due dispositivi, è stata ideata una soluzione di questo tipo:

1. nel gestionale amministratore di Unipiazza, si associa al tablet l'id dello smartphone e viceversa;
2. il gestionale si occupa di informare Ably di questa associazione, in modo che lo smartphone e il tablet si iscrivano al canale di comunicazione dell'attività commerciale;
3. a questo punto, per comunicare tra loro, il tablet e lo smartphone possono pubblicare e leggere gli aggiornamenti all'interno del canale di Ably, secondo il paradigma "publish/subscribe".

4.4.3. I vantaggi di Ably

L'utilizzo di Ably rispetto al Bluetooth per la comunicazione tra smartphone e tablet porta ai seguenti vantaggi:

- sistema di comunicazione più semplice da implementare e personalizzare;
- alta disponibilità del servizio (99.999% di "uptime");
- comunicazioni criptate e a latenza inferiore;
- nessuna interferenza di rete all'interno del tablet poiché il Bluetooth non è utilizzato;

- la gestione delle associazioni tra smartphone e tablet può essere fatta da remoto, utilizzando il gestionale amministratore di Unipiazza.

Dato il numero consistente di vantaggi, l'impiego di Ably è stata la soluzione che tutt'oggi viene utilizzata per la comunicazione tra smartphone e tablet all'interno dell'attività commerciale.

4.5. Aggiornamento remoto delle app "business"

Una volta forniti i tablet e gli smartphone alle attività commerciali, bisogna implementare un sistema che consenta di aggiornare da remoto l'app di Unipiazza.
L'aggiornamento deve essere automatico, in modo che il gestore non debba preoccuparsene.

Quando viene sviluppata una nuova versione dell'app per tablet e smartphone lato gestori, il relativo file "apk" (pacchetto di installazione per le app Android) viene caricato sul server Amazon di Unipiazza, ad un URL specifico, al quale i client possono accedere solamente facendo una richiesta API autenticata.
Il tablet e lo smartphone devono quindi collegarsi a questo URL, scaricare il file "apk" dell'aggiornamento e installarlo al posto della precedente versione dell'app.

Sono state ideate e testate due soluzioni per svolgere questa operazione all'interno del tablet e dello smartphone:

1. Eseguire questa operazione esclusivamente tramite l'app principale di Unipiazza
2. Servirsi di un'app esterna, che si occupi di installare gli aggiornamenti per l'app principale di Unipiazza

La prima soluzione implementata è stata quella di aggiornare l'app senza servirsi di app esterne. Di seguito viene descritta questa soluzione, illustrata in *figura 4.5.1*.

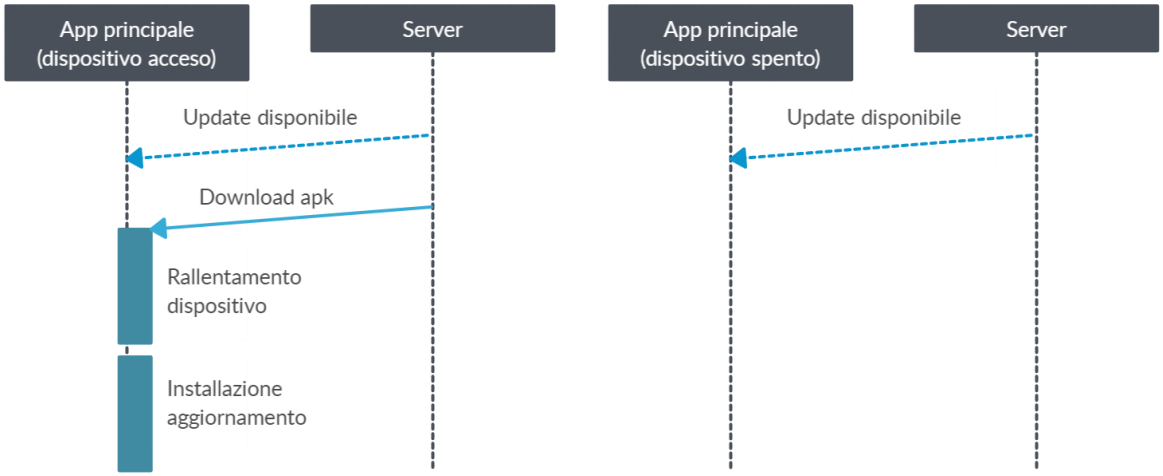

Figura 4.5.1: aggiornamento remoto tramite la sola app principale di Unipiazza

Quando è presente un nuovo aggiornamento, il server invia un messaggio al dispositivo con l'URL dell'"apk" da scaricare. Il dispositivo scarica immediatamente l'"apk", causando un possibile rallentamento dell'app principale della durata di 2-4 minuti (talvolta anche più lungo: dipende dalla connessione internet).
A scaricamento completato, il dispositivo installa subito la nuova versione, causando un disservizio temporaneo del sistema di Fidelizzazione dell'attività commerciale (qualche decina di secondi). Poiché durante il processo di aggiornamento l'app non può essere utilizzata, se il processo di aggiornamento avviene durante gli orari di servizio dell'attività commerciale, la situazione diventa critica.

Un altro problema è che non tutti i dispositivi sono accesi quando viene inviato il messaggio dal server, perciò, è possibile che alcuni dispositivi non ricevano l'aggiornamento.
Inoltre, per aggiornamenti "massivi", centinaia di dispositivi si ritrovano a scaricare la stessa risorsa nel medesimo istante, causando un traffico elevato verso il server e allungando quindi i tempi di download.

La seconda soluzione è stata quella di sviluppare un'ulteriore app chiamata "Unipiazza Updater", che sarebbe stata installata sui tablet e gli smartphone forniti alle attività commerciali. Di seguito viene descritta questa soluzione, illustrata in *figura 4.5.2*.

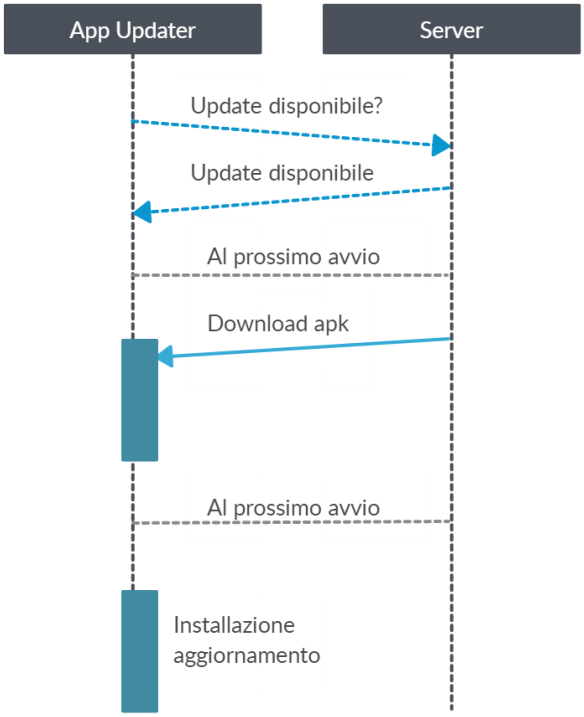

Figura 4.5.2: aggiornamento remoto tramite l'app "Unipiazza Updater"

Ad ogni avvio del dispositivo, l'app "updater" interroga il server sulla disponibilità di aggiornamenti, in caso positivo scarica l'"apk" e confronta l'MD5 con quello fornito dal server per verificare l'integrità del file di installazione.
All'avvio successivo del dispositivo, se l'app "updater" trova un file "apk" scaricato, allora installa l'aggiornamento dell'app principale e cancella il file di installazione.

Questa soluzione ha diversi vantaggi. Innanzitutto, i dispositivi vengono a conoscenza della presenza di aggiornamenti anche se sono spenti, inoltre, il download e l'installazione avvengono entrambi quando il dispositivo viene acceso: un istante in cui ci si può aspettare un piccolo momento di disservizio.
Un altro vantaggio è che, se sono necessari aggiornamenti "massivi" di molti dispositivi, questi scaricano il file "apk" solamente al loro avvio, consentendo di distribuire il traffico nel tempo, diminuendo eventuali rallentamenti dovuti ad accessi concorrenti alla stessa risorsa.

Quest'ultima soluzione è stata decretata la migliore, poiché diminuisce il tempo di disservizio e permette l'aggiornamento dei dispositivi, anche se sono momentaneamente spenti. Perciò, l'app ausiliare coinvolta in questa soluzione, chiamata "Unipiazza Updater", è attualmente installata in tutti gli smartphone e i tablet delle attività commerciali.

4.6. Comunicazione tra dispositivi lato “business” e server: algoritmo di “exponential backoff”

I dispositivi lato “business”, ossia smartphone e tablet delle attività commerciali, comunicano con il server di Unipiazza. Può succedere che, a causa dell’inaffidabilità dei collegamenti a Internet presenti all’interno delle attività commerciali, l’invio di un messaggio dai dispositivi al server, o viceversa, fallisca.

Per questo motivo, quando il dispositivo invia una richiesta al server, esegue un processo in background che si occupa della gestione della risposta del server. Se la risposta non arriva entro un timeout di X secondi, il processo invia nuovamente la richiesta.

La problematica di questa soluzione è la decisione della durata del timeout tra una richiesta e quella successiva.
Tramite l’esempio seguente è facile intuire l’impatto che ha questa scelta:

- ipotizzando che il server sia irraggiungibile per 20 minuti, se si utilizza un timeout troppo corto, verranno generate moltissime richieste che non andranno a buon fine, inutilmente;
- ipotizzando che il server sia irraggiungibile per soli 3 secondi, se si utilizza un timeout troppo lungo, il dispositivo attenderà più di quanto è realmente necessario prima di ricevere una risposta.

Una soluzione migliore comprende l’impiego dell’algoritmo di “exponential backoff”. L’idea che sta dietro a questo algoritmo è quella di utilizzare dei timeout progressivamente più lunghi al crescere del numero di tentativi effettuati. La crescita della durata del timeout segue l’andamento di una funzione esponenziale (*figura 4.6.1*).

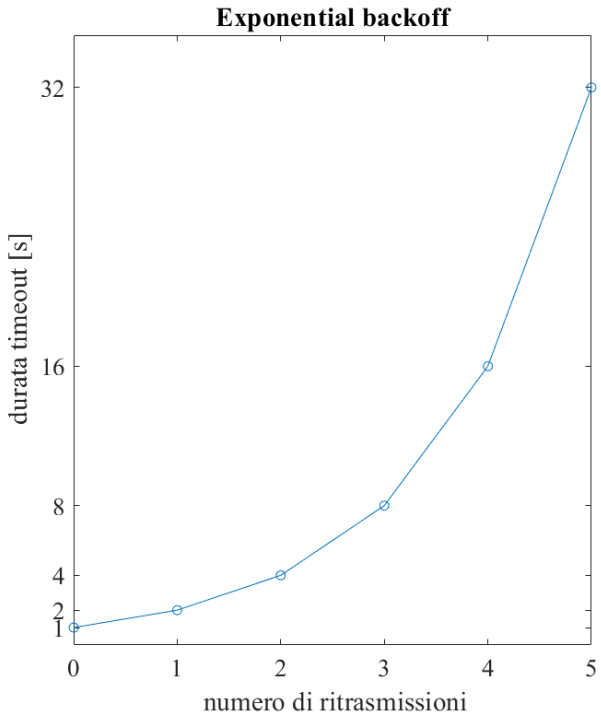

Figura 4.6.1: algoritmo di "exponential backoff"

Ipotizziamo il caso in cui il server sia in grado di gestire al massimo N richieste inserendole in una coda. Il server estrae di volta in volta una richiesta dalla coda, la processa e invia la risposta al client.
Nel caso in cui il server riceva molte richieste in un breve intervallo di tempo, poiché il tempo per processare una risposta non è trascurabile, è possibile che la coda si riempia completamente. Quando la coda è piena, tutte le successive richieste in arrivo verranno ignorate senza inviare una risposta.

Sotto queste ipotesi, l' algoritmo di "exponential backoff" è migliore rispetto a quello a timeout costante poiché lascia al server un tempo sempre maggiore prima di effettuare una nuova richiesta, diminuendo la probabilità che la coda del server si riempia.

Questo algoritmo può essere personalizzato in base al contesto d'utilizzo, aggiungendo per esempio un "jitter" nel calcolo di ogni timeout, ossia una componente di tempo casuale che diminuisce la probabilità di collisioni tra richieste provenienti da diversi client verso il server.
É consigliabile, inoltre, impostare un numero di ritrasmissioni massimo (oppure un timeout massimo) che, una volta superato, secondo la nostra logica dell'app, permetta di visualizzare un errore. In questo modo possiamo accorgerci dell'esistenza di qualche effettivo problema, non momentaneo, nella comunicazione tra il dispositivo e il server.

5. Conclusioni

In questa tesi è stato analizzato il processo di creazione di un servizio per la fidelizzazione clienti, implementato tramite delle app per smartphone e tablet.

Nel capitolo “2. Analisi dei requisiti” è stato importante determinare in maniera puntuale i requisiti alla base del servizio, in modo da effettuare le scelte implementative migliori per la realizzazione di un servizio facile e intuitivo.

In seguito, all’interno del capitolo “3. Overview dell’architettura informatica”, è stato fatto un confronto tra l’approccio orientato ai microservizi e quello monolitico, spiegando le motivazioni della scelta finale. È stata descritto, poi, com’è stata effettivamente implementata l’architettura informatica di Unipiazza, spiegando il funzionamento delle varie componenti.

Nel capitolo “4. Realizzazione delle app”, sono state descritte le interfacce utente e la logica presenti nelle app per smartphone e tablet.
In fase di realizzazione, sono emerse alcune problematiche tecniche, non prevedibili in fase di analisi. Grazie al metodo di sviluppo descritto nella sezione “2.7 Metodo di sviluppo”, è stato possibile ideare soluzioni alternative in un breve tempo. Queste soluzioni sono state discusse nelle sezioni: “4.3. La scelta dei dispositivi”, “4.4. Comunicazione tra tablet e smartphone gestori”, “4.5. Aggiornamento remoto delle app “business” e “4.6. Comunicazione tra dispositivi lato “business” e server: algoritmo di “exponential backoff””.

Grazie all’approccio utilizzato, in cui il riscontro dell’utente finale è la priorità, il servizio sarà progressivamente arricchito con funzionalità utili.
Avendo utilizzato un approccio altamente scalabile per l’architettura informatica, sarà facile ospitare un numero sempre maggiore di utenti in futuro.

5.1. Sviluppi futuri

5.1.1. Modalità lite del servizio

In questo momento, è in corso la realizzazione di una modalità "lite" del servizio di fidelizzazione Unipiazza. Questa modalità consiste nel proporre tutto il servizio in maniera gratuita ai gestori, per consentir loro di testarlo nella propria attività commerciale. Dopo un certo periodo di tempo, i gestori potranno valutare, anche tramite le statistiche fornite da Unipiazza, se il servizio ha portato dei benefici all'attività commerciale, permettendo loro di scegliere se passare alla versione a pagamento o meno.

5.1.2. Passaggio al framework "React"

Nello sviluppo delle app smartphone, è in corso il passaggio da Android e iOS nativi a "React Native", un framework "Javascript" per la programmazione di app native per Android e iOS.
Questo framework è sempre più presente nello sviluppo delle app mobile, poiché presenta una serie di vantaggi. Quello principale è consentire la realizzazione di app in maniera "cross-platform", ossia realizzare solo una versione del codice per poter eseguire l'app su sistemi operativi differenti (come, ad esempio, Android e iOS).

Anche il gestionale lato "business" e admin sta subendo questo cambiamento. In particolare, trattandosi di sviluppo web, il framework che si sta adottando è "React", ossia la versione web di "React Native".
Inoltre, utilizzare lo stesso framework per lo sviluppo mobile e quello web, ha il vantaggio di poter riutilizzare stesse parti di codice per entrambe le piattaforme. In questo modo lo sviluppo è ancora più veloce.

5.2. Il mio contributo

Il mio tirocinio presso la startup Unipiazza si è svolto tra il 07/09/2020 e il 06/11/2020, per una durata totale di 225 ore.

Durante questo periodo ho contribuito in varie aree che riguardano il sistema di fidelizzazione e gli altri servizi forniti da Unipiazza. Ho lavorato nell'area informatica, occupandomi sia del "front end " che del "back end", sviluppando nuove funzionalità o risolvendo eventuali problematiche.

Di seguito un elenco delle principali attività a cui mi sono dedicato:

- Miglioramento interfaccia grafica delle app Android per clienti e gestori.
- Realizzazione delle nuove app in "React Native" per lo smartphone lato clienti.
- Realizzazione di componenti grafiche in "React" per l'e-commerce.
- Realizzazione di un metodo standardizzato per il "backup" e "restore" dei dati nei dispositivi Android lato "business"
- Realizzazione di uno script per l'aggiornamento remoto dell'app gestori per Windows
- Implementazione della funzionalità che permette ai gestori di collegare la propria pagina Facebook a Unipiazza, utilizzando le API di Facebook.
- Risoluzione di eventuali problemi.

6. Bibliografia

Amazon (2021) *Microservizi*:
https://aws.amazon.com/it/microservices (Acceduto il: 29/08/2021)

Ably (2021) *Ably*:
https://ably.com (Acceduto il: 31/08/2021)

Amazon (2021) *Ripetizione dei tentativi in caso di errore e backoff esponenziale in AWS:*
https://docs.aws.amazon.com/it_it/general/latest/gr/api-retries.html
(Acceduto il: 02/09/2021)

Vito Lavecchia (2021) *Differenza e vantaggi tra Architettura monolitica e Architettura di microservizi in informatica:*
https://vitolavecchia.altervista.org/differenza-e-vantaggi-tra-architettura-monolitica-e-architettura-di-microservizi-in-informatica (Acceduto il: 07/09/2021)

Vito Lavecchia (2021) *Caratteristiche, funzionamento e vantaggi dell'Architettura Monolitica in informatica:*
https://vitolavecchia.altervista.org/caratteristiche-funzionamento-e-vantaggi-architettura-monolitica-in-informatica (Acceduto il: 07/09/2021)

Node.js (2021) *Informazioni su Node.js:*
https://nodejs.org/it/about (Acceduto il: 09/07/2021)

MongoDB (2021) *MongoDB:*
https://www.mongodb.com/it-it (Acceduto il: 09/07/2021)

Amazon (2021) *Cloud computing con AWS:*
https://aws.amazon.com/it/what-is-aws (Acceduto il: 09/07/2021)

Heroku (2021) *Heroku:*
https://www.heroku.com/what (Acceduto il: 09/07/2021)