



POLITECNICO DI BARI

DIPARTIMENTO DI INGEGNERIA ELETTRICA E DELL'INFORMAZIONE

CORSO DI LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA

Business Report
Work Project

Advanced Software Engineering

Wear2Pay

Docente:

Prof.ssa Ing. Marina Mongiello

Referente aziendale:

Ing. Giuseppe Falagario

Project leader:

Ing. Giuseppe Rizzi

Codice Gruppo: M7

Componenti gruppo:

Sergio Abascià

Gianluca Azzollini

Alberto Carlo Maria Mancino

Sommario esecutivo

Obiettivi

Wear2Pay, sviluppato in collaborazione con Sitael SPA, nasce con l'idea di creare un sistema che consenta di acquistare prodotti da un distributore automatico, in maniera semplice ed intuitiva, utilizzando uno smartwatch WearOS.

A bordo dello smartwatch verrà installata una apposita applicazione, dal nome Wear2Pay, utile ad instaurare una connessione con la vending machine, e completare l'acquisto del prodotto selezionato.

L'utente che deciderà di iscriversi al servizio Wear2Pay risulterà provvisto di:

- un wallet virtuale, cui è possibile attingere per effettuare un acquisto;
- uno storico delle ultime 5 transazioni effettuate sulla piattaforma.

Il team di sviluppo si è occupato dello sviluppo di un'ulteriore applicazione smartphone, la cui funzione è quella di simulare il comportamento di un distributore automatico.

Smartphone (distributore), e smartwatch, potranno interagire utilizzando le applicazioni appositamente sviluppate.

In ultimo è stato implementato un Server Apache, corredato di un database relazionale, utile allo storing delle informazioni necessarie al funzionamento del sistema.

Proposta

Il seguente Business Report ha come principale obiettivo quello di fornire al lettore una overview generale sul ciclo di sviluppo del Sistema, soffermandosi in particolar modo sull'architettura dello stesso, sulla gestione del team di sviluppo e sulle problematiche e decisioni cruciali, prese durante la fase di sviluppo.

Struttura del progetto

Presa disposizione degli obiettivi finali del progetto, si sono rese necessarie le seguenti competenze ed i relativi scope.

Competenza richiesta:	Scope:
Android Studio IDE [1]	Ambiente di programmazione delle applicazioni Smartwatch & Smartphone
Java Language [2]	Linguaggio di programmazione applicazioni SW & SP
XML Language [3]	Linguaggio di programmazione, utilizzato per il layout grafico applicazioni SW & SP
Apache Web Server [4]	Server volto alla gestione del Backend, in particolare portafoglio utente e relative transazioni, in ingresso ed in uscita.
DBMS MySQL [5]	Sviluppo e gestione database Backend
HTML [6]	Linguaggio di formattazione Web, utilizzato per il Form di registrazione Utente;
CSS [7]	Linguaggio di formattazione parte grafica Web, utilizzato per il Form di registrazione Utente;
JavaScript [8]	Linguaggio di programmazione Web utilizzato per il Form di registrazione Utente
php [9]	Linguaggio di programmazione utilizzato per le chiamate ai servizi Backend
JSON [10]	Formato adatto all'interscambio di dati fra applicazioni client/server. Utilizzato per le risposte alle chiamate ai servizi backend
Bluetooth Low Energy [11]	Protocollo di comunicazione scelto per lo scambio di informazioni tra smartwatch e smartphone

Il diagramma di Gantt [12] è uno strumento di supporto alla gestione dei progetti.







Ad ogni micro obiettivo è associato il tempo necessario, stimato in ore, per completare il task, nonché il responsabile o i responsabili per la sua realizzazione.

Per avere una stima dei tempi necessari al completamento del progetto, e per facilitare la suddivisione dei compiti in relazione alle competenze necessarie e/o acquisite dai membri, il diagramma è così raffigurato:

[illegible]

[illegible]

I colori rappresentano i responsabili secondo la seguente legenda:

-  Sergio Abascià
-  Gianluca Azzollini
-  Alberto Mancino
-  Sergio Abascià e Gianluca Azzollini
-  Gianluca Azzollini e Alberto Mancino
-  Sergio Abascià, Gianluca Azzollini e Alberto Mancino

Sommario

Introduzione.....	1
Sezione I Dominio di interesse	2
Sezione II Piattaforma e Tecnologie.....	4
Sezione 3 Soluzione	7
Sezione IV.I Validazione e Test	15
4.1.1 Tipologie di test	15
4.1.2 Strumenti di test	16
4.1.3 Oggetti sottoposti al test	16
Sezione 4.2 Risultati e discussione 1.....	17
4.2.1 Test di funzione.....	17
4.2.2 Risultati test di funzione	18
4.2.3 Test multiutente.....	18
4.2.4 Risultati test multiutente	18
4.2.5 Test prestazionale.....	19
4.2.6 Risultati test prestazionale	19
4.2.7 Test di usabilità.....	20
4.2.8 Risultati del test di usabilità	21
Conclusioni e sviluppi futuri	22
Riferimenti.....	23

Introduzione

Il progetto Wear2Pay creato e sviluppato in collaborazione con Sitael SPA è volto alla creazione di un sistema, che permetta l'acquisto di un prodotto da distributore automatico, utilizzando in maniera semplice, ed intuitiva, uno smartwatch WearOS [13] dotato di applicazione Wear2Pay.

Il lavoro è il risultato delle conoscenze acquisite durante il corso di Advanced Software Engineering, facente parte delle attività didattiche del corso di Laurea Magistrale in Computer Science Engineering.

Wear2Pay è un sistema di applicazioni volto a simulare l'acquisto di prodotti, da un distributore automatico, attingendo dal credito residuo di un wallet virtuale collegato all'utente per il pagamento del prodotto.

Il Sistema è quindi composto da:

- una applicazione smartwatch volta ad effettuare acquisti da un distributore automatico;
- una applicazione smartphone utile a simulare l'attività di una vending machine;
- un servizio back-end basato su server Apache, integrante un database relazionale, utile alla memorizzazione delle informazioni degli utenti, wallet virtuale e lista delle transazioni per ciascun utente in ingresso ed in uscita.

La feature principale del sistema è rappresentata dall'applicazione smartwatch Wear2pay.

Le restanti componenti completano il sistema, rendendolo testabile e funzionante, in tutte le sue parti.

Massima attenzione è stata quindi prestata al rispetto delle linee guida di sviluppo e di design delle applicazioni per smartwatch, nonché alla scalabilità dell'applicazione stessa su modelli differenti di smartwatch.

Sezione I Dominio di interesse

I benefici derivanti dall'utilizzo, nonché i portatori di interesse della suddetta piattaforma, risultano essere molteplici.

Vari customer segments, potranno usufruire dei servizi offerti di interesse della piattaforma.

Le aziende stesse che fabbricano e commercializzano distributori automatici, potrebbero essere interessate ad incrementare i propri introiti, dotando le proprie vending machine delle tecnologie utili ad usufruire dei servizi Wear2Pay.

Distributori di questo tipo necessiteranno di modulo Bluetooth in grado di supportare la tecnologia wireless Bluetooth Low Energy, e di componenti che permettano un qualunque tipo di connessione alla rete Internet.

Ipotizzando un caso d'uso in cui un potenziale acquirente non abbia a disposizione monete o banconote di piccolo taglio, e/o il distributore automatico non sia in grado di erogare resto, poiché non sufficiente.

In tal caso l'applicazione Wear2Pay eviterebbe un qualsiasi problema di questo tipo, consentendo all'utente di acquistare in pochi e semplici passi un prodotto, sfruttando il proprio wallet virtuale.

I portafogli virtuali stanno oggi giorno acquistando popolarità via via crescente, poiché in grado di offrire agli utenti flessibilità, convenienza, sicurezza nel pagamento, nonché facilità d'utilizzo.

Evitano anche l'ingombro di portare con sé monete o banconote.

L'utilizzo di un wallet virtuale, corredato da un database relazionale, come nel nostro caso, aprirebbe le porte ad una vastissima gamma di scenari, atti alla profilazione degli utenti e dei loro trend comportamentali.

Il tutto, garantendo nella maniera più assoluta, privacy e sicurezza dell'utente.

Si renderebbero così possibili operazioni di data mining, utilizzando, sotto consenso dell'utente, le informazioni fornite durante la fase di registrazione.

Evidenziamo di seguito alcuni esempi di informazioni raccolte attraverso operazioni di data mining:

- saldo medio residuo sul wallet virtuale degli utenti;
- spesa media giornaliera / settimanale / mensile / annuale degli utenti;

- importo medio ricaricato sul wallet virtuale;
- associando il nome del prodotto alla posizione del prodotto nella macchinetta (id del prodotto nella nostra base dati) è possibile recuperare le informazioni del prodotto e tenere traccia dei trend di vendita di ciascun prodotto;
- a partire dalle informazioni ottenute al punto precedente, risulterà possibile ricavare informazioni sui prodotti più venduti, o meno venduti, in quali si è venduto più, in quale meno, quale è stato il distributore automatico da cui si è venduto di più, e molto altro;

Informazioni di questo tipo porterebbero le aziende che si occupano della gestione dei distributori automatici ad una più mirata gestione dei prodotti in vendita, delle loro quantità e della locazione dei prodotti, per ciascuna vending machine.

Sulla base delle informazioni raccolte, sarà quindi possibile inviare promozioni o sconti particolari, invogliando utenti che non acquistano prodotti da molto tempo, nonché assidui clienti, a riutilizzare il servizio messo loro a disposizione, con frequenza via via crescente.

Sezione II Piattaforma e Tecnologie

Il seguente paragrafo riporta un quadro generale relativo alle tecnologie adottate nella realizzazione del sistema Wear2Pay, spiegando e motivando le scelte legate al loro possibile utilizzo.

Si è puntato sullo sviluppo di applicazioni per AndroidOS e WearOS, utilizzando come ambiente di sviluppo Android Studio.

Tale scelta si è rivelata quasi obbligata, in relazione al notevole supporto fornito dalla documentazione Google ed alla chiara, semplice, ed intuitiva gestione del ciclo di vita delle activity.

Le activity sono la componente atomica di una qualsiasi applicazione smartphone / smartwatch basata su sistema operativo della nota compagnia californiana.

In merito all'Hardware utilizzato per lo sviluppo e per i successivi test, si è scelto come modello di Smartwatch il dispositivo Ticwatch E [14] basato su sistema operativo Android WearOS, sulla base di motivi puramente di natura economica.

Per quanto riguarda invece l'applicazione atta a simulare il comportamento di un distributore automatico, lo Sviluppo Software è avvenuto su smartphone Xiaomi Mi 9T Pro [15].

Lo smartwatch supporta connessione Bluetooth 4.2, lo smartphone connessione Bluetooth 5.0.

Entrambi i dispositivi supportano quindi BLE(Bluetooth Low Energy).

Il Bluetooth Low Energy [16] è la tecnologia wireless selezionata per consentire ai due dispositivi di comunicare.

Più specificatamente, il Bluetooth Low Energy è una tecnologia wireless personal area network progettata e commercializzata dal Bluetooth Special Interest Group (Bluetooth SIG) per nuove applicazioni, nel settore dell'assistenza sanitaria, fitness, per i beacon, per la sicurezza, per l'industria dell'intrattenimento domestico e per le industrie automobilistiche e dell'automazione.

Rispetto al Bluetooth "classico", il Bluetooth Low Energy ha lo scopo di fornire un consumo energetico nonché costi notevolmente ridotti, mantenendo un intervallo di comunicazione più o meno simile, garantendo la possibilità di regolare l'intensità del segnale.

Bluetooth SIG ha inoltre previsto che entro il 2018 oltre il 90% degli smartphone abilitati Bluetooth supporterà Bluetooth Low Energy.

Tutti i dispositivi Bluetooth Low Energy utilizzano il Profilo di Attributo Generico (GATT). Non è altro che una interfaccia software che definisce come i dispositivi possano inviare e ricevere dati, descrivendo i concetti di Servizio e Caratteristica. Il GATT sfrutta a sua volta l'ATT, ovvero l'Attribute Protocol, utilizzato per contenere i dati dei Servizi e delle

Caratteristiche che il GATT mette a disposizione all'esterno.

I dati relativi a Servizi e Caratteristiche sono memorizzati in una apposita tabella utilizzando un identificatore lungo 16 byte noto come UUID, trasmesso ogni volta che la connessione tra Client e Server viene instaurata.

Distinguiamo allora:

Client

Un dispositivo che avvia comandi e richieste GATT ed accetta risposte, ad esempio, un computer o uno smartphone, nel nostro caso uno smartwatch.

Server

Un dispositivo che riceve comandi e richieste GATT e restituisce risposte, ad esempio un sensore di temperatura.

Nel nostro caso il distributore automatico, o più specificatamente l'applicazione smartphone che simula un distributore automatico.

Characteristic

Un valore di dati trasferito tra client e server.

Service

Una raccolta di caratteristiche correlate, che cooperano eseguendo una particolare funzione. Ad esempio, il servizio Health Thermometer include le caratteristiche per un valore di misurazione della temperatura, ed un intervallo di tempo tra le misurazioni.

Descriptor

Un Descriptor fornisce ulteriori informazioni utili relative ad una determinata Characteristic.

Discuteremo in seguito in maniera più dettagliata come è possibile sfruttare la comunicazione GATT del BLE, dal servizio Wear2Pay.

Non è stato tuttavia il BLE la prima scelta: si è pensato inizialmente all'utilizzo di sensori NFC, più immediati, pratici ed addirittura meno costosi del già economico Bluetooth Low Energy.

Tuttavia il team di sviluppo ha dovuto forzatamente scartare la soluzione NFC: i lettori NFC hanno limitazioni su smartwatch: leggere dati NFC non è possibile da un service in Android. Le funzionalità NFC come ricevere Intents, Beam ed altre, sono infatti disponibili soltanto per attività in foreground.

È possibile utilizzare soltanto Google Pay attraverso il built-in NFC chip, simulando la tecnologia Contactless delle carte di credito / debito.

La seconda scelta del team di sviluppo è ricaduta sulla tecnologia Bluetooth standard, per poi ricadere in maniera definitiva, su consiglio degli ingegneri Sitael SPA [17] (un ringraziamento

particolare va all'Ing. Rizzi), verso la scelta ultima del Bluetooth Low Energy, più economico e meno dispendioso in termini di consumi energetici.

Ci focalizziamo adesso sul servizio backend offerto dalla nostra piattaforma.

Il server scelto è il **web Server Apache** sviluppato dalla Apache Software Foundation.

È un software che realizza le principali funzioni di trasporto delle informazioni, di internetwork e di collegamento, ed ha il vantaggio di offrire funzioni di controllo per la sicurezza come quelle effettuate da un proxy.

Tale server è stato corredato di database relazionale **MySQL**.

La motivazione di questa scelta risiede nelle conoscenze pregresse del team di sviluppo, ed è basata sulla semplicità di implementazione e utilizzo del software messo a disposizione dalla Apache Software Foundation.

Inoltre, la gestione attraverso **PhpMyAdmin**, e quindi interfaccia grafica, del database relazionale, ha semplificato e ridotto i tempi di sviluppo del backend Wear2Pay.

Per interfacciare il web server alla rete, e permettere alle applicazioni di richiamare i suoi servizi, si è scelto di utilizzare il servizio di tunnelling **Ngrok** [18].

Questo associa alla macchina su cui viene avviato un URI e si mette in ascolto reindirizzando le richieste su una data porta, nel nostro caso la porta 80.

Le iscrizioni dei clienti al sistema Wear2Pay sono gestite tramite richieste automatiche a servizi Php lato server. Tali richieste sono formattate ed inviate tramite specifico sito Web, da noi appositamente sviluppato.

Questo presenta un classico form di iscrizione in cui viene richiesto all'utente di inserire:

- Nome
- Cognome
- Indirizzo e-mail
- Username
- Password

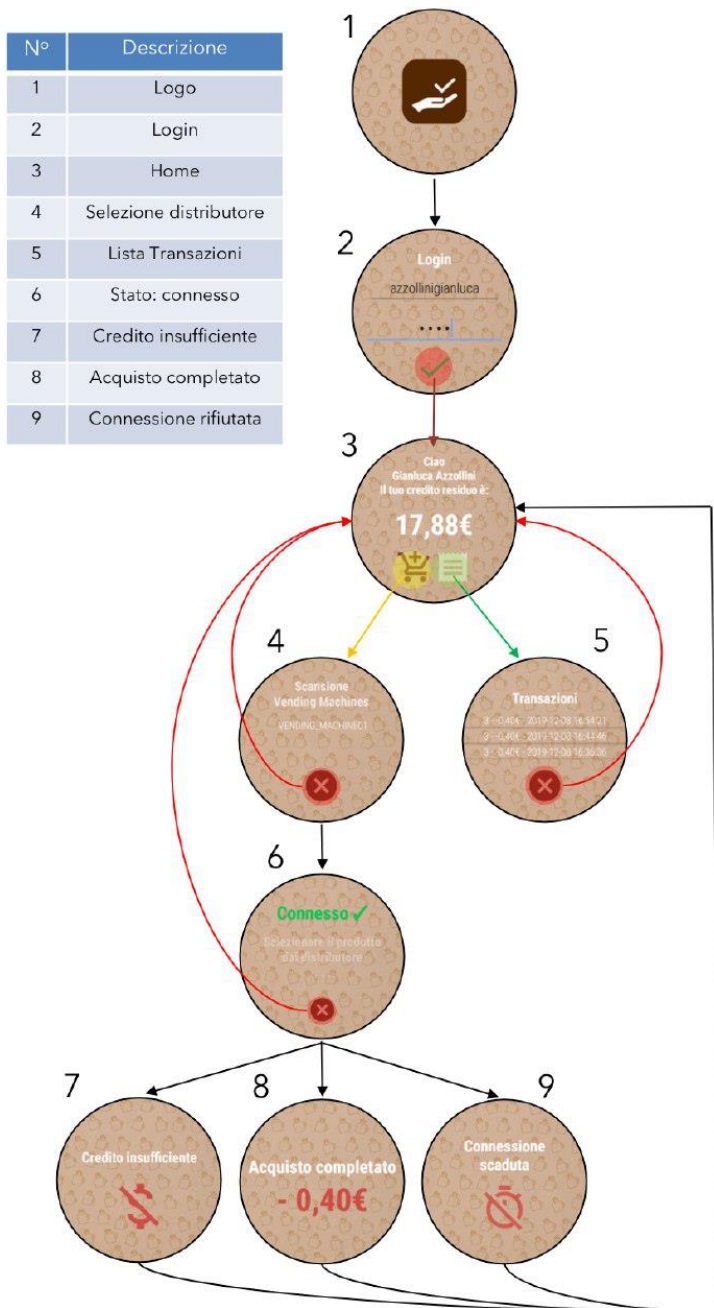
Inviata la richiesta, il server verifica che non siano già iscritti utenti aventi username e/o indirizzo e-mail coincidenti con quelli appena inseriti.

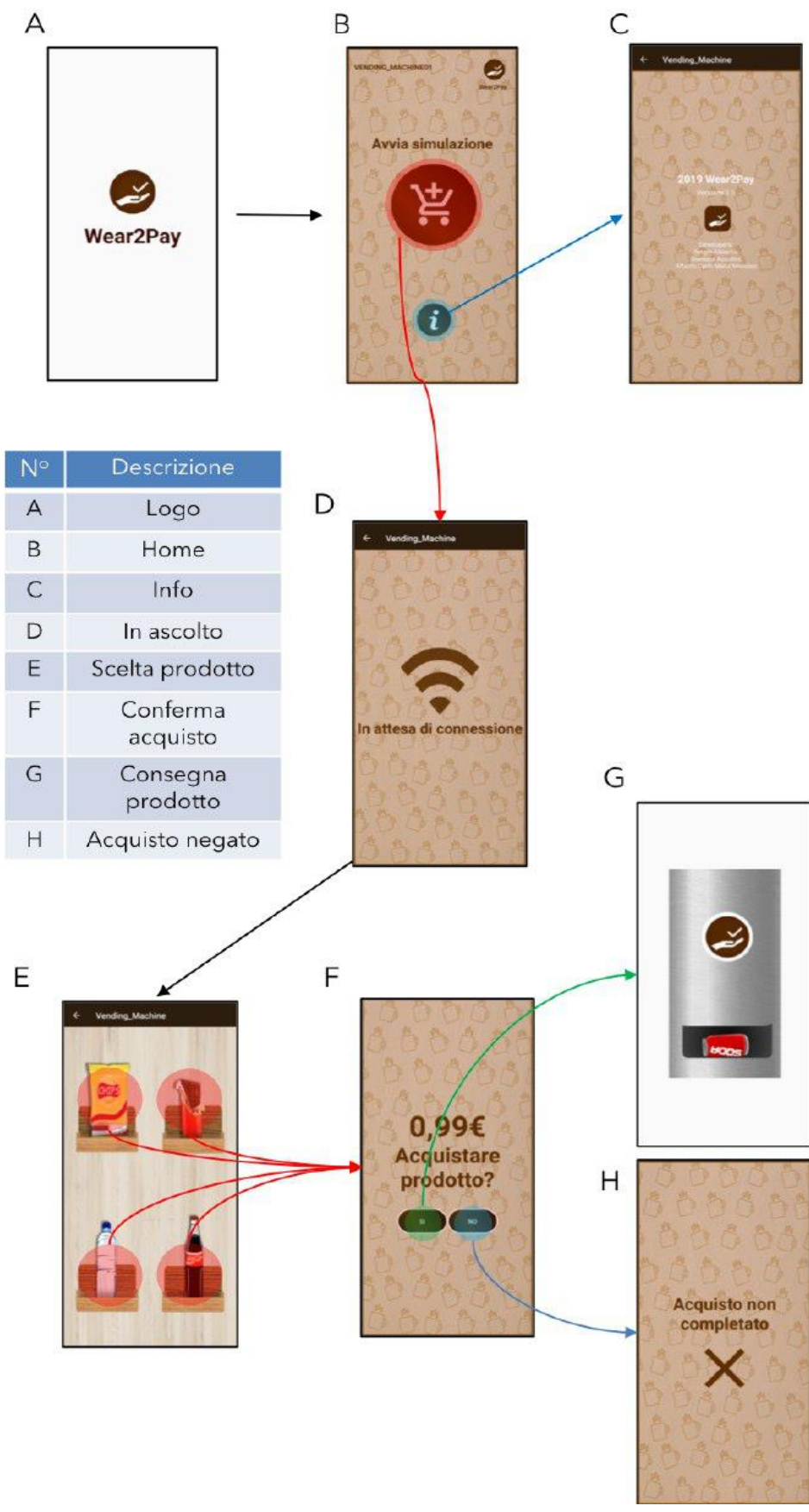
Se la verifica da esito positivo, l'iscrizione avviene con successo.

Sezione 3 Soluzione

Nel seguente paragrafo attraversiamo il ciclo di vita delle applicazioni che compongono il sistema.

Di seguito riportiamo il workflow di utilizzo dell'applicazione Wear2Pay e dell'applicazione simulante il funzionamento di un distributore automatico.





Ipotizzando una registrazione [19] di un nuovo utente attraverso l'apposito Web Form, è necessario specificare:

- Nome;
- Cognome;
- Login name;
- Password;
- Indirizzo email;

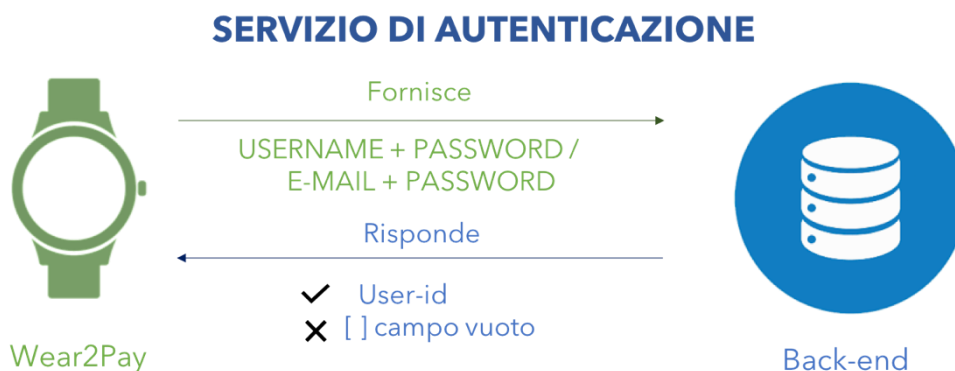


Una volta effettuata la registrazione possiamo esaminare il flow dell'applicazione Wear2Pay.

Aperta l'applicazione ed effettuato il caricamento del logo [1], verrà aperta l'activity del Login [2].

L'activity terrà in memoria l'ultimo login effettuato e consentirà, con la pressione del bottone rappresentato da una spunta verde, di effettuare il login.

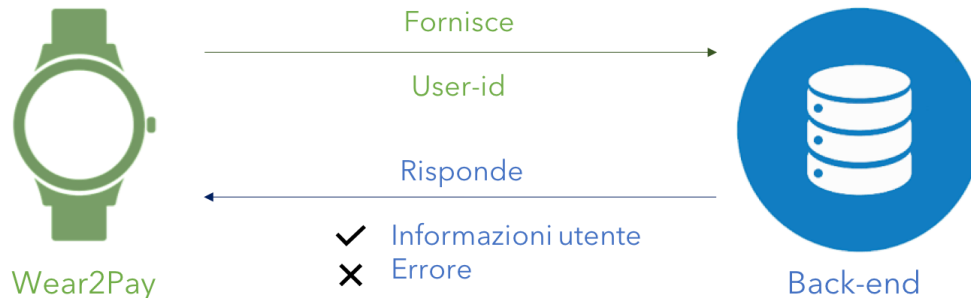
In alternativa sarà possibile modificare l'user da tastiera, modificando le credenziali di accesso.



Effettuato il login con successo avviene il passaggio all'activity [3].

All'apertura della activity [3] verrà effettuata la seguente richiesta al servizio di backend da parte dell'applicazione Wear2Pay.

SERVIZIO DI RECUPERO DELLE INFORMAZIONI UTENTE



Viene così recuperato e mostrato a video il nome dell'utente, il suo cognome, nonché il credito residuo su wallet virtuale.

Cliccando sul bottone avente come background un foglio bianco si aprirà l'activity [5] con conseguente chiamata al servizio backend da parte dell'applicazione Wear2Pay.

SERVIZIO DI RECUPERO DELLE TRANSAZIONI



Verranno quindi mostrate le ultime cinque transazioni effettuate (in ordine cronologico) dall'utente e sarà possibile tornare al menu principale dell'applicazione (Activity [3]) cliccando sulla croce rossa posta in basso.

Cliccando alternativamente sul carrello verde, dal menu principale [3], verrà aperta l'activity [4] .

Si suppone in questa fase che si sia resa disponibile la vending machine sulla Activity [D] dell'applicazione smartphone, per la simulazione del distributore automatico stesso.

All'apertura della activity [4] dunque, viene avviato il processo di scansione del Bluetooth Low Energy, lato GATT Client (smartwath).

Una volta terminata la scansione, della durata di 10 secondi, comparirà a video la lista dei distributori automatici disponibili alla connessione.

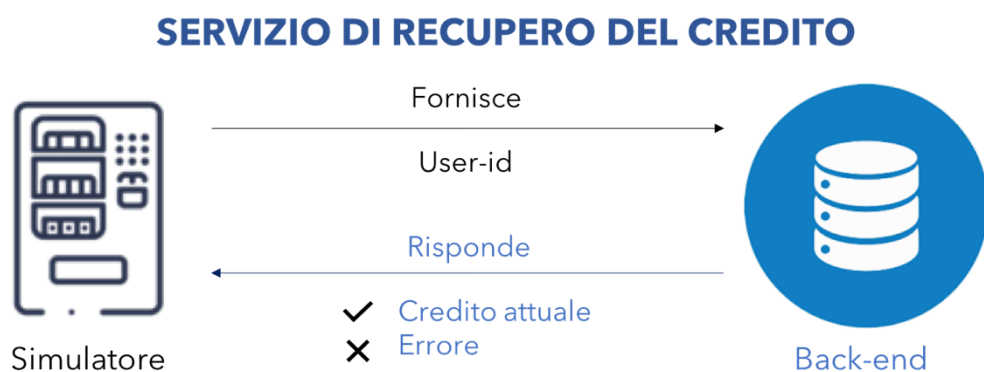
Per distributori automatici disponibili alla connessione si intendono quei distributori vicini allo smartwatch in termini di segnale BLE, nonché tutti quei distributori non impegnati in altre connessioni, o interazioni manuali e “classiche” con qualsiasi tipo di utente.

Cliccando sul nome del distributore, sarà allora possibile effettuare la connessione.

A questo punto il GATT Client si preoccupa di scrivere un'unica caratteristica, esposta da un unico servizio presente sul GATT Server (distributore automatico), come riportato in figura.

SCHEMA OROLOGIO -> MACCHINETTA (USERID)

Una volta recuperato l'User Id dall'applicazione Wear2Pay, il distributore automatico recupererà il wallet virtuale dell'utente, come evidenziato in figura, con la seguente chiamata al servizio backend.



Il prossimo step del workflow sarà l'apertura dell'activity [6] dell'applicazione Wear2Pay e l'utente sarà in grado di selezionare, come suggerito a schermo, un prodotto dal distributore automatico.

E' bene sottolineare come a partire da questo momento non si renderà più necessaria alcuna interazione dell'utente con l'applicazione smartwatch Wear2Pay.

Si evince che, con il login effettuato, sia possibile connettersi ed acquistare un prodotto da distributore automatico con due semplici tap.

Questo punto è stato oggetto di particolare attenzione da parte del team di sviluppo: di notevole importanza è stato infatti il rispetto delle linee guida che rappresentano lo stato dell'arte per lo sviluppo di app per wearable devices, in particolar modo app smartwatch.

È fondamentale che un'applicazione come Wear2Pay, destinata a questa tipologia di Hardware necessiti di un numero minimo di interazioni quanto più basso possibile, e che queste interazioni siano basilari su smartwatch. È fondamentale che l'utilizzo dell'app da parte dell'utente, sia quanto più chiaro semplice ed immediato.

È a questo punto possibile scegliere il prodotto dal distributore automatico ed effettuare l'acquisto (Activity [E] e [F]) .

Una volta selezionato il prodotto, l'applicazione che simula il distributore automatico verificherà che il wallet dell'utente sia sufficiente a completare la transazione, ed in caso di esito positivo, contatterà il backend, come segue in figura, registrando l'effettivo avvenimento della transazione.



Una volta registrata o meno la transazione sulla base dati, ne verrà salvato l'esito su Servizio, e sulla Caratteristica esposta del GATT Server, presente su distributore automatico.

Dopo un timeout di trenta secondi, l'applicazione Wear2Pay potrà leggere la caratteristica attraverso il suo GATT Client e aprire l'activity successiva che riporterà l'esito, positivo o negativo della transazione.

SCHEMA OROLOGIO <- MACCHINETTA (esito transazione)

Activity	Valore letto dalla caratteristica	Esito transazione
[7]	"-1.0"	Credito insufficiente
[8]	prezzo del prodotto acquistato	Successo
[9]	"-2.0"	Errore di connessione

Una volta aperta l'activity che mostra all'utente l'esito della transazione, dopo un ulteriore timeout di sette secondi, l'applicazione Wear2Pay tornerà al menu iniziale (Activity [3]) mostrando a schermo il credito aggiornato sul wallet virtuale dell'utente.

E' di rilevante importanza il workaround implementato, reso necessario a seguito di alcune problematiche riscontrate nell'implementazione della comunicazione Bluetooth Low Energy.

Dopo numerosi test eseguiti è stato accertato che effettuare due connessioni BLE tra due dispositivi in un breve intervallo temporale comporta in maniera frequente uno "Status code error 133.

Come evidenziato di sviluppo Android di Google, è buona norma in tal caso tenere aperta la connessione tra i due dispositivi GATT Server e GATT Client.

Il workaround adottato per la seguente problematica consiste nell'effettuare una seconda scansione da parte del GATT Client (smartwatch) atta a rilevare il GATT Server con cui il Client si era precedentemente connesso.

Una volta rilevato la scansione viene interrotta, il GATT Client si collega al Server e si prepara a leggere la caratteristica mostrata dal servizio sul GATT Server.

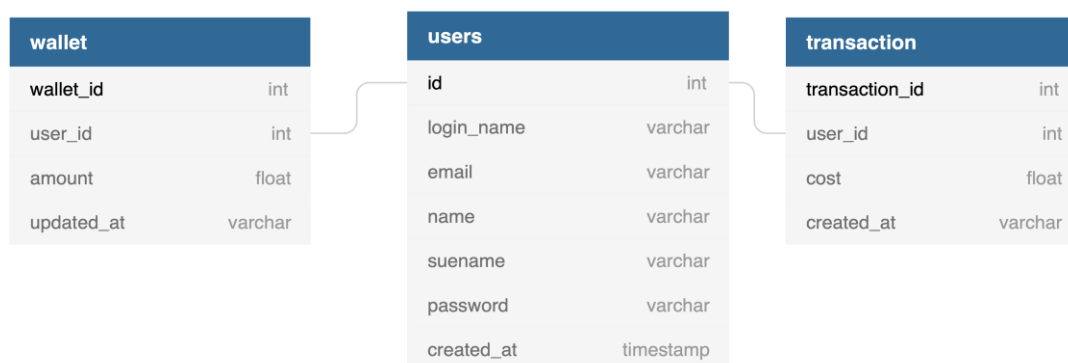
La base dati costruita per completare il sistema è una base dati semplice e puramente dimostrativa.

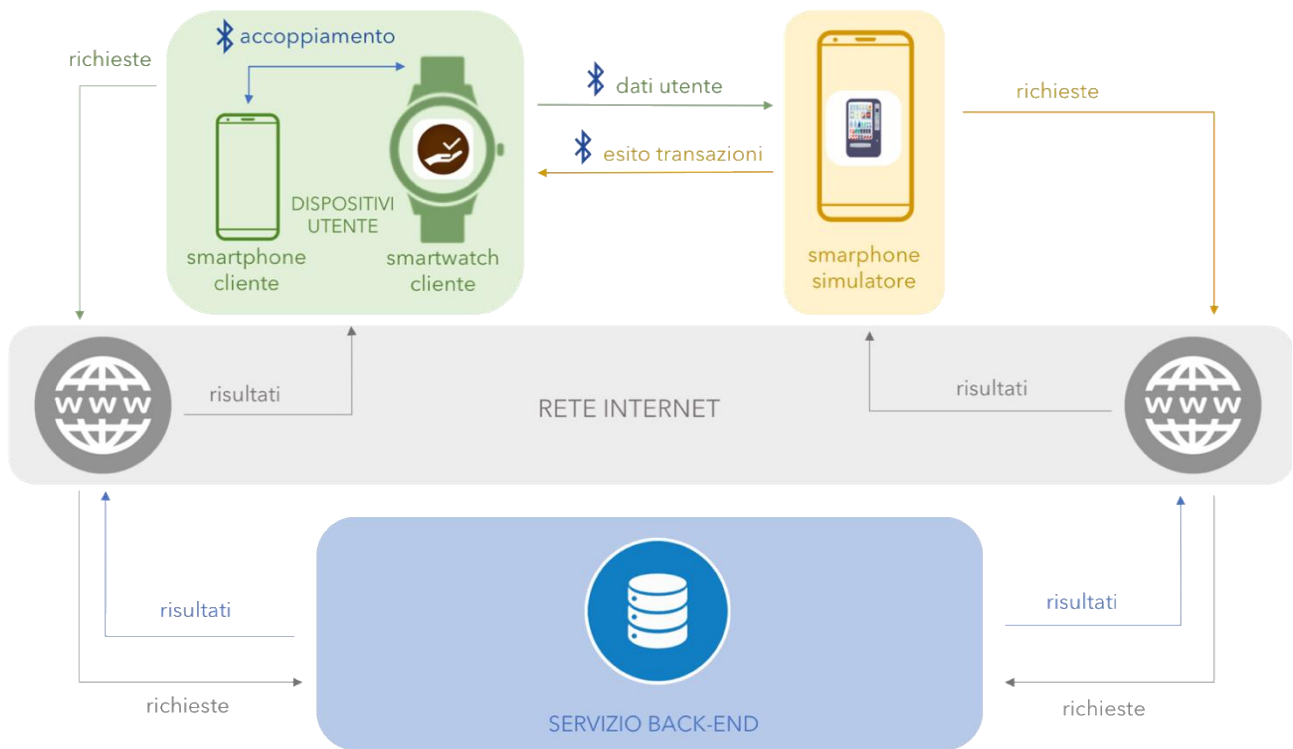
Essa è composta come segue:

- una tabella "users" contenente i principali dati di profilazione dell'utente;
- la tabella "wallet" contenente il portafoglio utente aggiornato;
- la tabella "transaction" contenente lo storico delle n transazioni di ogni utente, in cui evidenziamo il "product_id", associato allo slot di prodotto presente nel distributore automatico.

Il database è inoltre provvisto di un trigger mySQL utile ad effettuare l'aggiornamento del wallet dopo l'inserimento di una transazione, sia essa positiva o negativa.

A seguire lo schema architettura completo della nostra soluzione.





Sezione IV.I Validazione e Test

Ciascuna fase di test [20] è stata preventivamente analizzata al fine di garantire:

- oggettività
- replicabilità
- documentabilità

È stato dunque definito un insieme di valori registrabili con le relative unità di misura.

Ciascuno di questi valori viene memorizzato durante la specifica fase di test, e formattato per adattarsi alla documentazione proposta.

Sono state selezionate solo le tecniche di test ritenute consistenti rispetto alla strumentazione a nostra disposizione.

Ciascun test è stato svolto per fini didattici.

Gli strumenti per il test sono poco diversificati e condizionati nelle scelte da vincoli di budget.

I test condotti vanno ritenuti come una approssimazione dei test ideali e va tenuto conto che non escludono la possibilità che si presentino errori di misurazione e/o di valutazione.

4.1.1 Tipologie di test

All'interno della seguente tabella riportiamo le tipologie di test selezionate. Per ciascuna di esse viene specificato il raggruppamento, scelto tra test funzionale, test non funzionale e test di integrazione, e l'obiettivo che la data tipologia di test si pone di raggiungere.

TIPOLOGIA DI TEST	RAGGRUP- PAMENTO	OBIETTIVO
Test di funzione	TF	Verifica che le funzionalità attese del sistema siano rispettate, verificando che sia rispettato il loro funzionamento atteso.
Test multiutente	TF	Verifica il corretto funzionamento con utenti concorrenti, in scenari simili
Test prestazionale	TNF	Verifica che il sistema rispetti i requisiti prestazionali richiesti
Test di usabilità	TNF	Verifica che il sistema soddisfi i requisiti di usabilità richiesti, come la facilità di comprensione delle interfacce e delle interazioni.

4.1.2 Strumenti di test

Per la realizzazione dei suddetti test si è reso necessario l'utilizzo di differenti strumenti, a seconda della natura del test.

TIPOLOGIA DI TEST	STRUMENTI
Test di funzione	Smartwatch Ticwatch E, Smartphone Xiaomi Mi 9T Pro, Laptop Personal Computer HP Omen 15-ax000nl, Smartphone Xiaomi Mi 9T, connessione a rete mobile Vodafone
Test multiutente	Laptop Personal Computer HP Omen 15-ax000nl [21], MacBook Pro 13" Early 2011, MacBook Pro 13" Early 2015
Test prestazionale	Smartwatch Ticwatch E, Smartphone Xiaomi Mi 9T Pro, Laptop Personal Computer HP Omen 15-ax000nl, Smartphone Xiaomi Mi 9T, connessione a rete mobile Vodafone
Test di usabilità	Smartwatch Ticwatch E, Smartphone Xiaomi Mi 9T Pro, Laptop Personal Computer HP Omen 15-ax000nl, Smartphone Xiaomi Mi 9T, connessione a rete mobile Vodafone.

Gli strumenti si sono resi necessari, al fine di quantificare e/o verificare le grandezze e le caratteristiche misurate all'interno dei test prestazionali e qualitativi.

- Test multiutente: fa utilizzo dell'analisi dei tempi di risposta forniti dal browser Firefox.
- Test prestazionale: fa utilizzo di librerie Java per il calcolo del tempo trascorso, utili per calcolare la durata delle activity.
- Test di usabilità: fa utilizzo di un form a scelta multipla per la raccolta dei pareri di un insieme di utenti campione, selezionati randomicamente.

4.1.3 Oggetti sottoposti al test

Richiamo di seguito tutti gli oggetti che sono stati sottoposti ai test suddetti:

- Codice applicazione Wear2Pay: le activity e i layout che compongono questa applicazione Android;
- Codice applicazione Simulatore: le activity e i layout che compongono questa applicazione WearOs;
- Codice servizi back-end: i file PHP che realizzano i servizi del back-end;
- Infrastruttura: WebServer Apache, servizio tunneling Ngrok, RDBMS MySQL, servizio di gestione PhpMyAdmin

Sezione 4.2 Risultati e discussione 1

4.2.1 Test di funzione

Attraverso il seguente test si verifica che le funzionalità attese del sistema siano realizzate in maniera corretta. Tali funzionalità sono indicate nel documento "Specifiche dei requisiti", cui viene fatto riferimento al paragrafo 1.2 di questo lavoro.

Le funzionalità del sistema che abbiamo scelto di testare sono riportate di seguito:

FUNZIONALITÀ	DESCRIZIONE
Connessione Bluetooth tra Wear2Pay e il simulatore.	<p>Il test verifica che nella fase di scelta del distributore si crei il canale Bluetooth tra le applicazioni simulatore e Wear2Pay.</p> <p>Tale verifica avviene tramite controllo dei messaggi di log all'interno della console di Android Studio.</p> <p>Il canale si deve venire a creare quando viene premuto il campo relativo al nome del simulatore, all'interno della lista che mostra i Device Name dei dispositivi BLE nelle vicinanze.</p>
Iscrizione di utente	<p>Il test verifica che una volta compilati tutti i campi presenti all'interno del sito web, l'iscrizione sia presente o meno all'interno del database. Il test controlla inoltre che in caso di username già iscritto, indirizzo e-mail già iscritto o in caso di campi mancanti, venga visualizzato un messaggio di errore, e venga data la possibilità di effettuare un nuovo tentativo.</p>
Visualizzazione delle transazioni	<p>Il test verifica che un utente loggato sull'applicazione Wear2Pay possa visualizzare lo storico delle proprie transazioni. Viene verificato che gli elementi che compongono tale lista siano coerenti rispetto a quelli presenti sul database.</p>
Segnalazione credito insufficiente	<p>Il test verifica che a seguito della selezione di un prodotto durante la fase di acquisto, il cui prezzo superi il credito residuo del cliente sul proprio portafoglio online, venga visualizzato un messaggio di errore, non venga registrata una nuova transazione, e non venga simulata la consegna del prodotto.</p>
Transazione avvenuta con successo	<p>Il test verifica che una transazione venga registrata con successo sul database quando viene selezionato un prodotto durante la fase di selezione dei prodotti, e che il credito residuo sul portafoglio online del cliente sia maggiore o uguale al costo del prodotto selezionato.</p>

Ciascuna funzionalità è testata dagli sviluppatori e viene ritenuta consona o meno.

4.2.2 Risultati test di funzione

Ciascuna funzionalità è stata verificata con successo.

4.2.3 Test multiutente

Attraverso il test multiutente si analizza il comportamento del sistema in presenza di al più 3 utenti, che ne fanno uso simultaneamente.

Il test si è focalizzato sulla componente del sito web, dedicata alla registrazione degli utenti al servizio.

Il test è suddiviso nei due seguenti sotto-test:

1. iscrizione simultanea di utenti con dati di registrazione differenti (TEST 1)
2. iscrizione simultanea di utenti con gli stessi dati di registrazione (TEST 2)

Il test cerca di simulare una situazione reale, in cui più utenti tentano di iscriversi al sistema attraverso il sito web. Per tale motivo non vengono utilizzati tool per l'invio di richieste HTTP simultanee, ma con l'utilizzo di 3 personal computer, inviando da ciascuno di essi, simultaneamente, una richiesta di iscrizione.

4.2.4 Risultati test multiutente

Per ogni test sono stati raccolti i seguenti dati:

- ID: identificativo del test: 1 indica il TEST 1, 2 indica il TEST 2;
- N° richieste: numero di richieste di iscrizione inviate contemporaneamente;
- Macchine: numero di computer utilizzati per il test;
- Tentativo: numero che indica di quale tentativo di test si tratta;
- Collegamento: può essere WLAN, LAN, MN (Mobile Network). Indica su che tipo di rete si trovano il client, che invia la registrazione, e il server, che la riceve;
- Esito: l'esito si ritiene positivo se non vengono iscritti due utenti con stesso username o stessa password, e la pagina di risposta giunge dal server in meno di 2000ms;

I risultati del test sono raccolti nella tabella che segue.

ID	No RICHIESTE	MACCHINE	TENTATIVO	COLLEGAMENTO	ESITO
1	2	2	1	WLAN	positivo
1	2	2	2	WLAN	positivo
1	2	2	3	WLAN	positivo
1	3	3	1	WLAN	positivo
1	3	3	2	WLAN	positivo
1	3	3	3	WLAN	positivo
2	2	2	1	WLAN	positivo

2	2	2	2	WLAN	positivo
2	2	2	3	WLAN	positivo
2	3	3	1	WLAN	positivo

4.2.5 Test prestazionale

Il test verifica che vengano rispettati i requisiti prestazionali specificati nel documento di analisi dei requisiti (vedi paragrafo 1.2).

Per la misurazione dei tempi è stata utilizzata la libreria TimingLogger. inserendo opportunamente nel codice le istruzioni, in corrispondenza dell'inizio e del termine dei requisiti prestazionali.

Per ciascun sotto-test si svolgono 20 tentativi, e per ciascun requisito viene riportato il valor medio dei tempi ottenuti.

4.2.6 Risultati test prestazionale

La tabella riporta i seguenti campi:

- ID: identificativo del requisito non funzionale, così come specificato nel documento di analisi dei requisiti;
- Descrizione: breve descrizione degli eventi misurati;
- Tempo: media dei tempi ottenuti in 20 misurazioni differenti;
- Limite: tempo massimo come dichiarato nell'analisi dei requisiti non funzionali;

ID	DESCRIZIONE	TEMPO	LIMITE
1	Apertura dell'applicazione Wear2Pay e la fine della schermata con logo	4,91 s	7,00 s
1	Apertura del simulatore la fine della schermata con logo	4,02 s	7,00 s
1	Viene premuto il pulsante di login e si attende la risposta dal server per la richiesta inviata	1,26 s	4,00 s
1	Viene premuto il pulsante della lista transazioni e le transazioni vengono mostrate a video	1,33 s	2,00 s
1	Viene premuto il tasto di ricerca dei distributori e la ricerca viene terminata mostrando i distributori.	5,39 s	10,00 s
1	Viene premuto il campo con l'identificativo del distributore e viene stabilito il collegamento	2,03 s	4,00 s
2	Viene premuto il pulsante che riporta "si" per la conferma di acquisto e viene mostrata l'animazione di consegna prodotto	5,57 s	10,00 s

4.2.7 Test di usabilità

Il test prevede altri elementi oltre all'utilizzo degli strumenti specificati nel paragrafo 4.1.2.

In particolare, si ricorre a:

- utenti volontari che non hanno mai utilizzato il sistema Wear2Pay
- un questionario per la raccolta delle opinioni

Per lo svolgimento del test viene selezionato un volontario per volta e gli viene chiesto di:

- utilizzare il sito web per iscriversi al servizio;
- utilizzare l'applicazione Wear2Pay per simulare l'acquisto di un prodotto dal simulatore;
- compilare un form di domande;

Le persone coinvolte nel test devono rispettare i seguenti requisiti:

- non devono aver mai utilizzato il servizio Wear2Pay in nessuna delle sue componenti;
- non devono conoscere il funzionamento delle interazioni a schermo;
- non devono aver assistito ad un altro test di questo specifico tipo;

A ciascun volontario viene riferito qual è lo scopo del servizio, e vengono i volontari stessi seguiti durante la fase di test, senza mai dar consigli riguardo il funzionamento specifico di alcuna componente.

Al volontario viene fornito, per la durata del test, un personal computer, uno smartwatch ed uno smartphone, collegato allo smartwatch e provvisto di connessione alla rete Internet.

Alla fine dell'esperienza viene chiesto di compilare un form come quello presente sul documento "Wear2Pay - Test di usabilità"[4].

I risultati ottenuti dall'esperienza di test e le risposte fornite dai volontari vengono valutati dagli sviluppatori.

4.2.8 Risultati del test di usabilità

Il test è stato svolto su un campione di 6 individui. Come si può notare dal grafico A, nessuna di queste aveva mai utilizzato l'applicazione Wear2Pay.

Avevi già utilizzato l'applicazione?

6 risposte

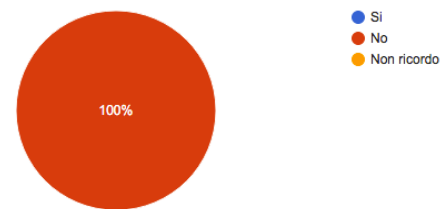


Figura A - Nessuno dei volontari aveva mai utilizzato l'applicazione Wear2Pay

Il sito web è stato ritenuto di semplice utilizzo, con una valutazione di 4,6 su una scala da 1 a 5, dove 5 indica una grande semplicità d'uso.

I risultati sono riportati nel grafico B.

SITO WEB: da 1 a 5 quanto ritieni semplice il suo utilizzo? (1 molto difficile, 5 molto facile)

6 risposte

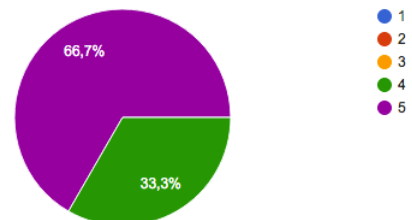


Figura B - La maggior parte dei volontari ha ritenuto molto semplice l'utilizzo del sito web per l'iscrizione al servizio Wear2Pay

Conclusioni e sviluppi futuri

L'ultima release di Wear2Pay risulta essere una prima soluzione completa e funzionante. Possibili scenari e sviluppi futuri verranno ipotizzati, elencati, e descritti di seguito.

La sicurezza delle comunicazioni, sia BLE che HTTP, è uno dei primi aspetti che andrebbero sviluppati e migliorati.

I dati trasmessi attraverso BLE potrebbero essere criptati, a discapito della attuale implementazione.

Il protocollo HTTP può essere sostituito con il protocollo HTTPS garantendo la sicurezza dei dati personali di ciascun utente, che vengono inviati.

Il protocollo Bluetooth Low Energy permette di settare il parametro della potenza del segnale trasmesso dal GATT Server. Attraverso test specifici si potrebbero analizzare numerosi possibili scenari, ottimizzando la fase di scansione dei distributori automatici nei dintorni. Inoltre, miglioramenti delle prestazioni si potrebbero ottenere riprogettando l'implementazione delle comunicazioni BLE tra GATT Server e GATT Client.

Un'altra possibile miglioria futura, potrebbe essere l'aggiunta della feature di ricarica wallet virtuale attraverso una carta di credito registrata al servizio backend, nonché la definizione completa di un sito web completamente dedicato alla gestione del servizio Wear2Pay.

Un altro caso d'uso implementabile è la possibilità di poter ricaricare il wallet virtuale dell'utente direttamente da distributore: l'utente, una volta collegato, potrebbe effettuare una ricarica tramite contanti.

A seguito dei test effettuati abbiamo denotato il bisogno, da parte di alcuni utenti, di avere una interfaccia più esplicativa: per alcuni dei nostri testers è risultato alquanto fuorviante la fase di selezione di un distributore. Questa potrebbe essere migliorata tramite suggerimenti a schermo, quali ad esempio i TOAST java.

Riferimenti

- [1] «Android Studio» [Online]. Available: https://it.wikipedia.org/wiki/Android_Studio.
- [2] «Java» [Online]. Available: https://www.java.com/it/download/faq/whatis_java.xml.
- [3] «XML» [Online]. Available: <https://en.wikipedia.org/wiki/XML>.
- [4] «Apache Web Server» [Online]. Available: <https://httpd.apache.org/>.
- [5] «DBMS MySql» [Online]. Available: <https://it.wikipedia.org/wiki/MySQL>.
- [6] «HTML» [Online]. Available: <https://www.html.it/guide/guida-html/>.
- [7] «CSS» [Online]. Available: <https://www.w3schools.com/css/>.
- [8] «Javascript» [Online]. Available: <https://www.html.it/guide/guida-javascript-di-base/>.
- [9] «PHP» [Online]. Available: <https://www.php.net/>.
- [10] «JSON» [Online]. Available: <https://www.json.org/json-it.html>.
- [11] «Bluetooth Low Energy» [Online]. Available: https://it.wikipedia.org/wiki/Bluetooth_Low_Energy.
- [12] «Diagramma di Gantt» [Online]. Available: https://it.wikipedia.org/wiki/Diagramma_di_Gantt.
- [13] «Wear OS» [Online]. Available: https://it.wikipedia.org/wiki/Wear_OS.
- [14] «Ticwatch E» [Online]. Available: <https://ticwatch.it/smartwatch-android-wear/9-47-ticwatch-express.html>.
- [15] «Xiaomi Mi 9T Pro» [Online]. Available: <https://www.mi.com/it/mi-9-t-pro/>.
- [16] «BLE Android» [Online]. Available:
<https://developer.android.com/guide/topics/connectivity/bluetooth-le>.
- [17] «Sitael SPA» [Online]. Available: <http://www.sitael.com/>.
- [18] «Ngrok» [Online]. Available: <https://ngrok.com/>.
- [19] «Documento dei Requisiti».
- [20] «Documento dei Test».
- [21] «HP Omen» [Online]. Available: <https://support.hp.com/it-it/document/c05228618>.
- [22] «Diagramma di Gantt» [Online]. Available: <https://blog.teamleadercrm.it/diagramma-di-gantt>.

