

ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA

Scuola di Scienze
Corso di Laurea in Ingegneria e Scienze Informatiche

APPLICAZIONE MOBILE PER
L'ORGANIZZAZIONE AZIENDALE E IL
TIMETRACKING

Relazione finale in
PROGRAMMAZIONE DI SISTEMI MOBILE

Relatore
Dott. MIRKO RAVAIOLI

Presentata da
MATTEO VITALI

Seconda Sessione di Laurea
Anno Accademico 2014 – 2015

PAROLE CHIAVE

Organizzazione processi aziendali

Timetracking

Android

A mio padre, mia madre e mio fratello

Indice

Introduzione	ix
1 Panoramica del sistema	1
1.1 Il mondo mobile	1
1.2 Software house	1
2 Analisi	3
2.1 Analisi dei requisiti	3
2.2 Prospettiva del sistema	3
2.3 Funzionalità principali	3
2.3.1 Diagramma dei casi d'uso	4
2.3.2 Diagramma delle attività	4
2.4 Utente finale	5
2.5 Sistemi già esistenti con funzionalità equivalenti a quelle richieste	6
3 Progettazione	7
3.1 Progetto generale	7
3.2 Server	7
3.3 Schema E/R	8
3.4 Schema logico	9
3.4.1 Dati derivati	10
3.4.2 Dati ridondanti	11
3.5 Web Server	11
3.6 Mockup	11
3.6.1 Login	12
3.6.2 Navigation Drawer	12
3.6.3 Profilo	12
3.6.4 Clienti	13
3.6.5 Preventivi e Progetti	13
3.6.6 Progetto	14
3.6.7 Modulo	14
3.6.8 Attivita	15

4	Implementazione	17
4.1	Server	17
4.1.1	Database Management System e Manager	17
4.1.2	Protocollo di comunicazione	17
4.1.3	Servizi	19
4.2	Applicazione	23
4.2.1	Tipologie di app	23
4.2.2	Mobile OS	23
4.2.3	Diffusione e distribuzione	24
4.2.4	Ambiente d'uso e limitazioni	25
4.2.5	Linguaggio e piattaforma di sviluppo	25
4.2.6	Librerie	25
4.2.7	Package	30
4.2.8	Pattern utilizzati	31
4.2.9	Funzionalità implementate	33
5	Testing	35
5.1	Dispositivi in uso	35
5.2	Tempi di latenza	35
	Conclusioni	37
	Ringraziamenti	39
	Bibliografia	43

Introduzione

Scopo dell'elaborato di tesi è il comprendere l'organizzazione di un'azienda tipo al fine di ottimizzarla e renderla più efficiente.

L'organizzazione aziendale può essere vista come il processo attraverso il quale l'insieme di persone che partecipa direttamente allo svolgimento dell'attività dell'azienda viene strutturato secondo i principi di divisione del lavoro e coordinamento.

La divisione del lavoro consiste nella scomposizione dei processi aziendali in attività elementari e nel raggruppamento di queste ultime in compiti secondo un criterio logico o tecnico. L'assegnazione dei compiti avviene in base ai ruoli definiti all'interno dell'azienda. I compiti assegnati ad un impiegato costituiscono le sue mansioni. Assegnando a più persone lo stesso compito si viene a creare un gruppo di lavoro, ognuno dei quali ha un capo gruppo che ha la responsabilità dell'operato del suo team.

Il coordinamento è strettamente legato alla divisione del lavoro in quanto ha lo scopo di armonizzare le decisioni e le attività dei gruppi di lavoro assicurando la fluidità delle attività, senza interferenze o disallineamenti temporali.

In che modo è possibile semplificare ed ottimizzare un procedimento così articolato? Per realizzare questo obiettivo viene analizzato, progettato e sviluppato un sistema per dispositivi mobile. Questo sistema informatico tra i principali vantaggi renderà possibile il coordinamento e la supervisione in real-time, effettuando una reportistica del lavoro svolto dai gruppi o dai singoli impiegati.

Capitolo 1

Panoramica del sistema

In questo capitolo viene presentata una panoramica del mondo mobile e un caso concreto che illustrerà dove poter utilizzare il sistema al fine di migliorare l'organizzazione aziendale.

1.1 Il mondo mobile

Negli ultimi anni lo sviluppo dei dispositivi mobile ha avuto un'esplosione che ha portato ad un cambiamento sostanziale nella vita quotidiana. L'utilizzo degli smartphone è entrato nell'uso comune; adesso è ordinario il poter interagire col mondo digitale direttamente dal palmo della mano.

L'innovazione tecnologica degli smartphone ha portato dietro di sé il boom delle app. Per ogni esigenza esiste un'applicazione che fa al caso dell'utente.

Sundar Pichai il nuovo Ceo di Google in un post pubblicato sul blog ufficiale di BigG ha annunciato: “In pochi anni la tecnologia mobile ha rivoluzionato il nostro modo di cercare informazioni, di divertirci, di comunicare con famiglia e amici, e di occuparci di quello che abbiamo da fare. (...) Le ricerche su Google provengono più da mobile che da desktop”. Questo dimostra come l'utente medio si stia spostando dall'utilizzo del PC all'uso di dispositivi mobile.

1.2 Software house

Compito principale di una software house è: sviluppare software. Ma esattamente cosa avviene all'interno della software house? Lo sviluppo di software cosa comporta a livello organizzativo?

Lo sviluppo di un software è un'opera complessa e al fine di strutturarla al meglio viene suddivisa in più fasi.

Pianificazione Il cliente, quando ha una commissione da svolgere si rivolge ad un impiegato della software house addetto al front office al quale espone il progetto che vuole far realizzare, evidenziando gli obiettivi e i fabbisogni.

Studio di fattibilità La software house a questo punto individua possibili strategie di attuazione al fine di stilare un preventivo contenente una stima dei costi e di ore necessarie per lo sviluppo del progetto richiesto.

Quando il preventivo viene esposto al cliente c'è la possibilità o meno che avvenga una contrattazione al termine della quale viene o meno accettata la commessa. Nel caso quest'ultima venga accettata, l'azienda inizierà a realizzare il progetto.

Controllo di qualità L'azienda e il cliente si accordano e predispongono un programma di controllo di qualità per il progetto, allo scopo di garantire il rispetto delle specifiche richieste, nonché di controllare che il sistema realizzato si comporti come previsto.

Analisi dei requisiti Il progetto viene affidato ad un capo progetto che formalizza i requisiti avvalendosi di tecniche di modellazione della realtà producendo macro specifiche per la fase di progettazione.

Progettazione Il capo progetto interpreta i requisiti in una soluzione architettonica di massima. Produce specifiche indipendenti dai particolari strumenti che saranno usati per la costruzione del sistema. Suddivide la struttura in moduli e produce specifiche che possano dar luogo, attraverso il ricorso a strumenti di sviluppo opportuni, a un prodotto funzionante.

Implementazione Il progetto viene effettivamente realizzato seguendo le specifiche e la scomposizione in moduli, facilitando lo sviluppo concorrente e le successive modifiche.

Test Il software viene inizialmente collaudato internamente alla software house attraverso l'alpha test, una volta che questo viene passato, si procede al beta test cioè il test da parte del cliente. Risolti tutti i bug relativi alle specifiche, il processo di sviluppo di un software ha raggiunto la sua conclusione (se si evita di considerare la manutenzione e l'evoluzione).

Capitolo 2

Analisi

2.1 Analisi dei requisiti

L'azienda ha richiesto di progettare e sviluppare un sistema che gestisca un ordine dal momento della creazione del preventivo alla chiusura del progetto. Una volta che il preventivo viene accettato, si crea un progetto suddiviso in moduli a loro volta composti da attività. Le ore lavorative vengono impiegate per risolvere le varie attività, attraverso dei report. Il sistema memorizza e gestisce tutti i dati su un server online. La scalabilità del sistema garantisce la possibilità di poter effettuare future modifiche e permettere l'aggiunta di nuove funzionalità.

2.2 Prospettiva del sistema

Il sistema software progettato e sviluppato, è un sistema che verrà utilizzato per la gestione delle varie commissioni di un'azienda. L'azienda ha richiesto questo sistema per semplificare e ottimizzare l'organizzazione aziendale e tenere traccia del processo di realizzazione di un progetto.

2.3 Funzionalità principali

Il sistema ha quindi tre compiti principali:

- Permettere la suddivisione del lavoro all'interno dell'azienda;
- Avere un sistema di notifiche abbinato alle varie scadenze commissionate;
- Gestire il timetracking attraverso la reportistica delle ore impiegate in determinate attività svolte dagli impiegati.

2.3.1 Diagramma dei casi d'uso

Per descrivere appieno le principali funzionalità del sistema viene utilizzato il diagramma dei casi d'uso, infatti questo diagramma rappresenta gli utilizzatori del sistema (attori) e l'interazione con esso. Nella figura (fig. 2.1) è descritto il diagramma dei casi d'uso del sistema che si è sviluppato.

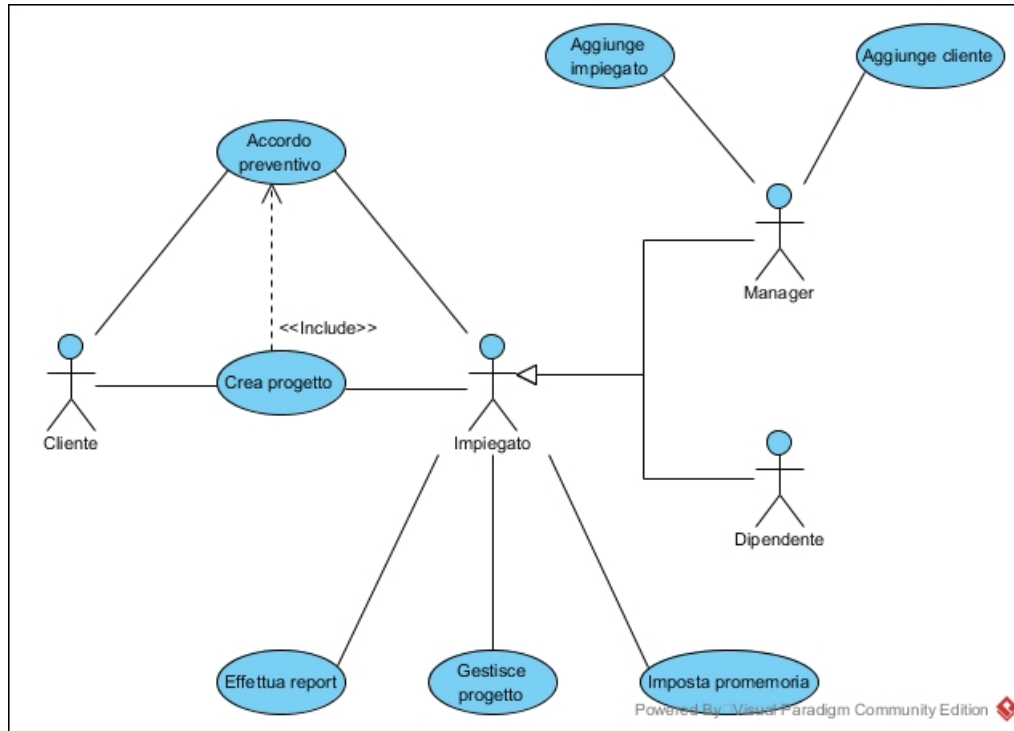


Figura 2.1: Diagramma dei casi d'uso

2.3.2 Diagramma delle attività

Al fine di capire appieno la procedura per la creazione di un progetto segue una spiegazione dettagliata e il diagramma delle attività che la rappresenta (fig. 2.2).

Il processo di creazione di un progetto inizia quando un cliente richiede un preventivo all'azienda fornendo le specifiche. Di conseguenza un impiegato dopo le necessarie valutazioni esporrà il preventivo con i costi e i tempi previsti. A questo punto il cliente valuta il preventivo e nel caso lo accetti viene concluso l'accordo concretizzando il progetto. Nel caso che il preventivo non soddisfi le esigenze del cliente, l'impiegato e quest'ultimo cercheranno un punto di incontro. Nell'eventualità che non venga trovato il punto d'incontro termina

il procedimento, nell'altro caso viene riformulato un progetto con le nuove specifiche definite.

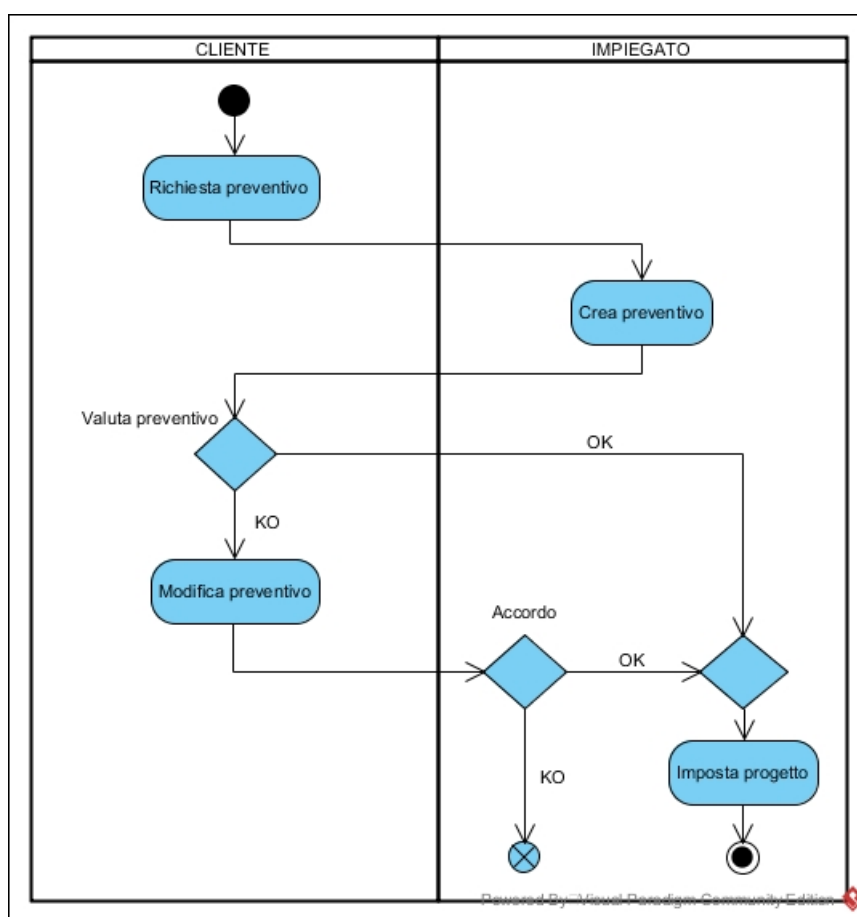


Figura 2.2: Diagramma delle attività

2.4 Utente finale

Il sistema è stato studiato affinché gli utenti finali siano gli impiegati dell'azienda al fine di ottimizzare e semplificare il loro lavoro. Per i responsabili aziendali la gestione attraverso questo sistema è resa più efficiente in quanto è possibile un coordinamento centralizzato.

2.5 Sistemi già esistenti con funzionalità equivalenti a quelle richieste

Se si effettua una ricerca di sistemi mobile già esistenti che soddisfino i requisiti richiesti, si trovano svariate applicazioni. Le migliori tra quelle analizzate sono le seguenti:

- Asana: sistema web e mobile molto diffuso che soddisfa tutti i requisiti richiesti meno che uno, perché gestisce una commissione da quando viene approvata da entrambe le parti, escludendo però la possibilità di gestire i preventivi.
- Zoho Projects: sistema web e mobile che come Asana soddisfa quasi tutti i requisiti ma non gestisce la possibilità di effettuare preventivi, inoltre non è user-friendly.

Capitolo 3

Progettazione

3.1 Progetto generale

Il progetto ha l'obiettivo di creare un sistema in real-time utilizzabile in parallelo da più utenti. Nella figura (fig. 3.1) si vede l'architettura a 3 strati di un sistema client server, come quello che dovrà essere implementato.

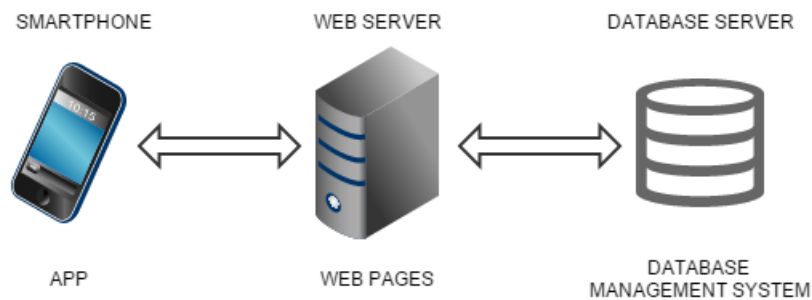


Figura 3.1: Schema logico dell'architettura a 3 strati Client - Server

3.2 Server

Un DBMS (Data Base Management System) è un sistema software in grado di gestire efficientemente le informazioni necessarie a un sistema informativo, rappresentandone i dati in forma integrata, e garantendone la persistenza. Un database (DB) è un insieme di dati e viene gestito tramite un DBMS. In quanto il sistema verrà adoperato in contemporanea da più utenti, i dati verranno esclusivamente salvati sul database all'interno del server, in modo da evitare ridondanza e semplificare la gestione della concorrenza. Una base

di dati si compone di uno schema, che ne descrive la struttura logica, e di un'istanza, che consiste nei dati memorizzati.

3.3 Schema E/R

Il modello Entity-Relationship è un modello per la rappresentazione concettuale dei dati, che descrive la realtà modellata indipendentemente da come i dati verranno logicamente e fisicamente rappresentati (fig. 3.2).

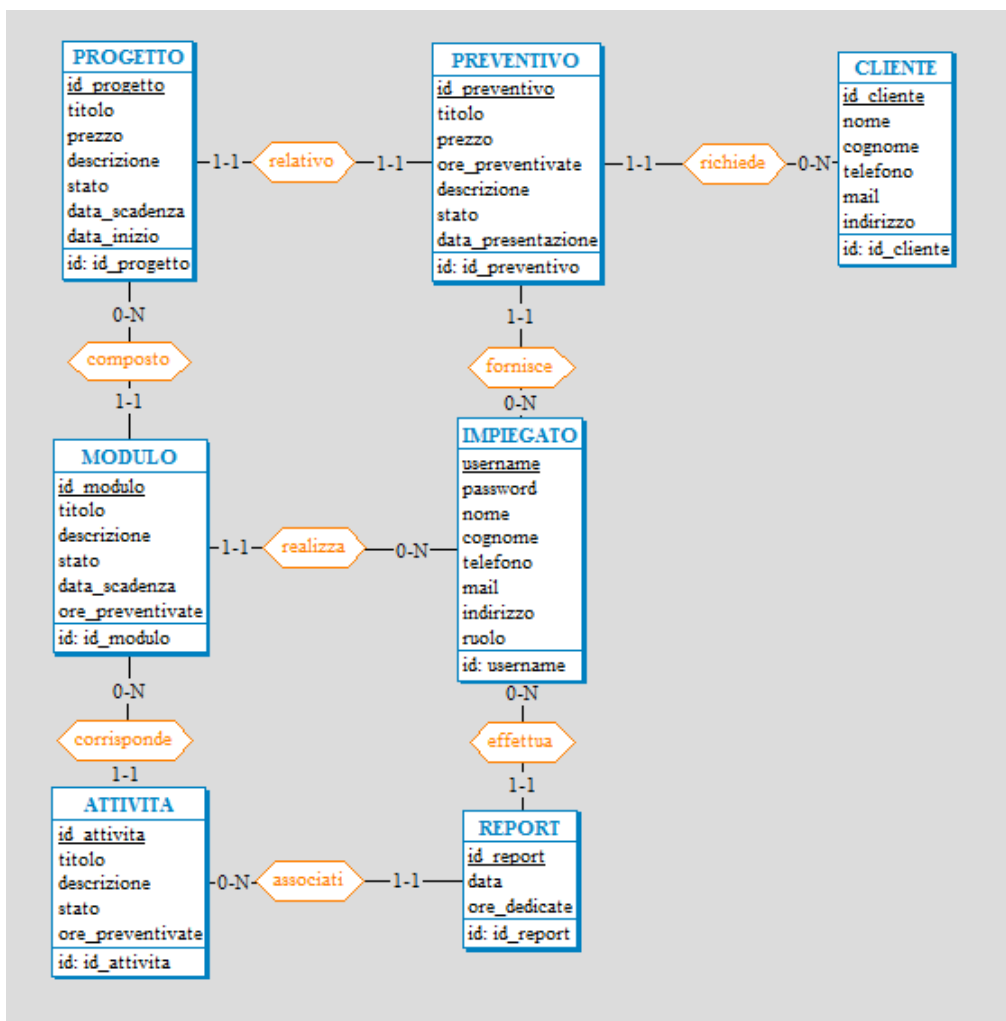


Figura 3.2: Schema E/R

Viene fornita una breve spiegazione delle varie entità e relazioni presenti nello schema E/R.

Cliente rappresenta la persona fisica che richiede un preventivo per una determinata commissione all'azienda. La sua entità è composta da una breve anagrafica.

Impiegato è un'entità che indica un'istanza di un impiegato dell'azienda, è composta da una breve anagrafica e dal ruolo che ricopre all'interno dell'azienda stessa.

L'entità preventivo denota un'istanza di un preventivo. Un preventivo viene istanziato quando un cliente contatta l'azienda per voler sviluppare un progetto esponendo una descrizione del sistema e richiedendo un preventivo di ore e di costo; tutte queste informazioni dettagliate sono inserite negli attributi insieme alla data di presentazione.

L'entità progetto indica un'istanza di progetto. Quando un preventivo viene accettato dal cliente, viene istanziato il progetto corrispondente.

Il modulo rappresenta un modulo di un progetto. Per semplificare il processo di sviluppo di un progetto, quest'ultimo viene suddiviso in moduli. Ogni modulo è affidato alla supervisione di un impiegato che dovrà controllare il processo e far rispettare i tempi di consegna.

Ogni istanza di modulo viene suddivisa in attività, ogni istanza di attività è una generica rappresentazione dell'entità attività. Attività è il compito base che può essere svolto all'interno dell'azienda. Più impiegati possono dedicare ore lavorative alla stessa attività.

L'entità report rappresenta un'istanza di report. Un report indica quante ore sono state utilizzate da un impiegato per svolgere una determinata mansione per una specifica attività nel giorno definito.

3.4 Schema logico

Lo schema logico derivato dallo schema E/R rappresenta quelle che sono le tabelle del DB e i loro collegamenti.

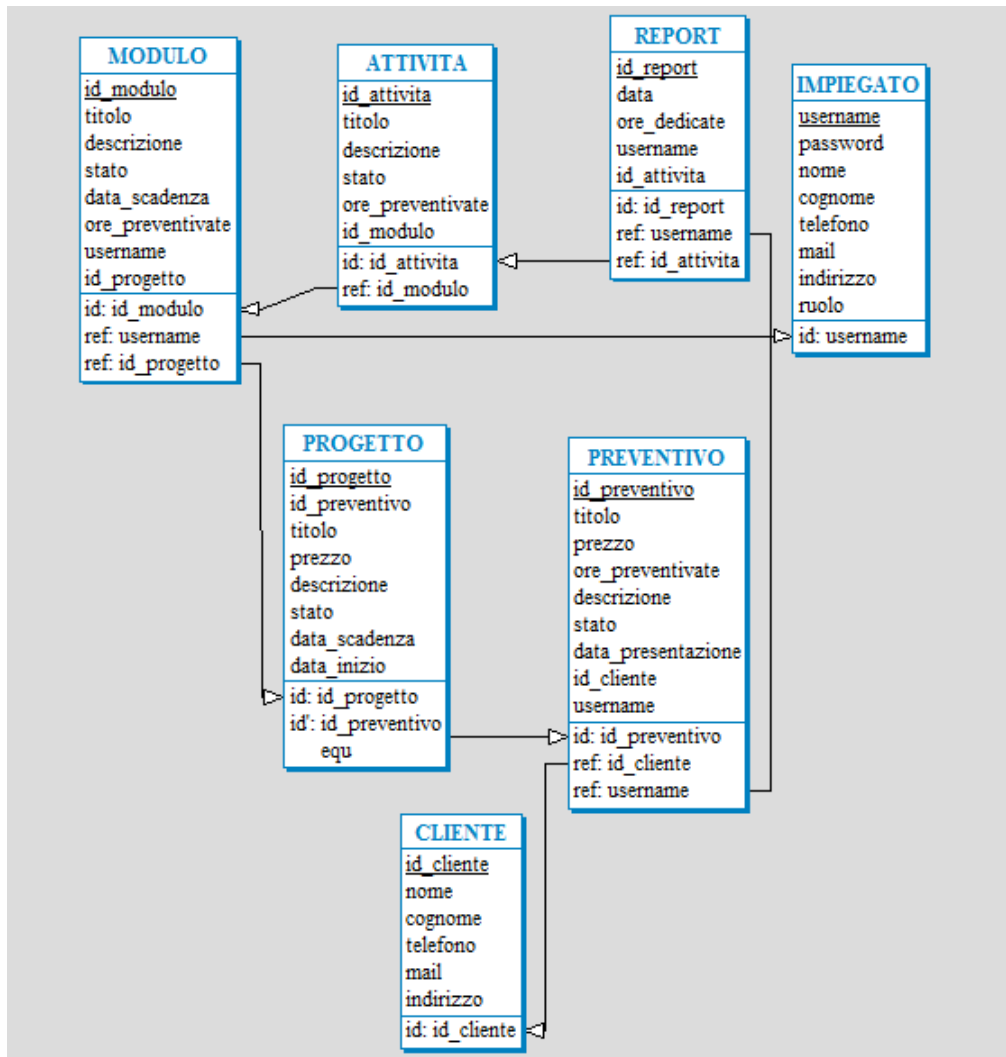


Figura 3.3: Schema logico

3.4.1 Dati derivati

Nel progetto non sono presenti dati derivati. Un possibile dato derivato poteva essere il totale ore e l'introduzione di questo avrebbe portato ad una riduzione degli accessi al DB. Gli svantaggi però sarebbero stati superiori rispetto ai vantaggi avrebbe avuto come conseguenza un aumento dell'occupazione della memoria con la necessità di effettuare operazioni aggiuntive per mantenere aggiornato il dato derivato.

3.4.2 Dati ridondanti

Nello schema logico (fig. 3.3) si vede la ridondanza dell'id del preventivo in progetto. Questa ridondanza è necessaria in quanto negli sviluppi futuri bisognerà poter associare ad un preventivo più progetti.

3.5 Web Server

La comunicazione tra client e server avviene attraverso il protocollo HTTP. Il web server è un software in esecuzione sul server in grado di gestire le richieste di pagine web di un client. Il web server interpreta la richiesta del client e gli invia una risposta. A questo punto il client è in grado di interpretare, a sua volta, la risposta ricevuta e fornirla all'utente. I principali servizi che il web server dovrà offrire sono:

- 1 Login: verifica le credenziali di accesso al sistema;
- 2 getClienti: fornisce una lista di tutti i clienti dell'azienda;
- 3 getImpiegati: fornisce una lista di tutti gli impiegati dell'azienda;
- 4 getProgettiCliente: fornisce una lista di tutti i progetti associati ad un determinato cliente;
- 5 getAttivita: fornisce una lista di tutte le attività di un determinato impiegato;
- 6 setReport: segnala il tempo che un determinato impiegato ha dedicato (n°ore) all'attività indicata;

3.6 Mockup

I mockup sono bozze delle schermate principali dell'applicazione. Sono molto importanti al fine di comprendere la struttura dell'app. Già dai mockup si può vedere come la grafica sia basata sul Material Design di Google. Il Material Design è un modo di gestire la UI (User Interface) in modo che l'utente possa sintetizzare i principi classici del buon design tramite l'innovazione della tecnologia e della scienza.

Di seguito ci sono i mockup delle schermate principali e/o la loro descrizione.

3.6.1 Login

Inserendo username e password dalla schermata di login (fig. 3.4a) si passa alla schermata principale dell'applicazione visibile nella figura (fig. 3.4b). Il login non sempre è la prima schermata, lo è solo nel caso non si sia mai fatto l'accesso oppure nel caso venga eseguito il logout.

3.6.2 Navigation Drawer

Il Navigation Drawer (fig. 3.4b) è un pannello che visualizza le principali opzioni di navigazione della app. È posizionato sul bordo sinistro dello schermo. Una volta che ne viene selezionata un'opzione viene mostrata la "pagina" ad essa collegata.

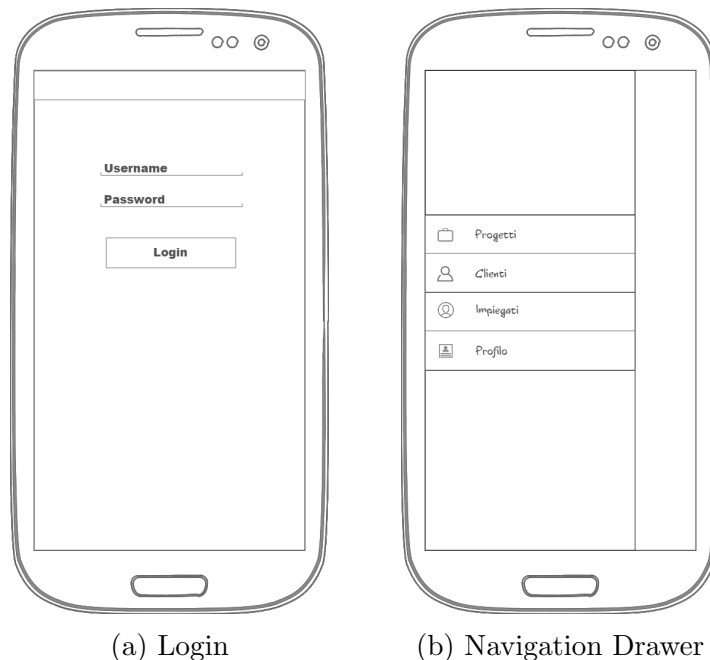


Figura 3.4: Mockup: login, Navigation Drawer

3.6.3 Profilo

Nel caso nel Navigation Drawer venga selezionata l'opzione "Profilo", apparirebbe a schermo l'anagrafica dell'impiegato loggato e un grafico. Nel grafico è rappresentata la quantità di ore lavorative svolte per giorno. Invece in alto sulla destra, nell'Action Bar c'è la possibilità di fare il log out dall'app.

3.6.4 Clienti

Selezionando nel Navigation Drawer "Clienti", visualizzeremmo una lista con l'elenco dei clienti dell'azienda. In basso a destra, se l'utente loggato come Manager c'è un Floating Action Button che se premuto avvia la procedura di aggiunta di un altro cliente.

3.6.5 Preventivi e Progetti

Selezionando un cliente il software renderizzata la schermata mockup come in figura (fig. 3.5).

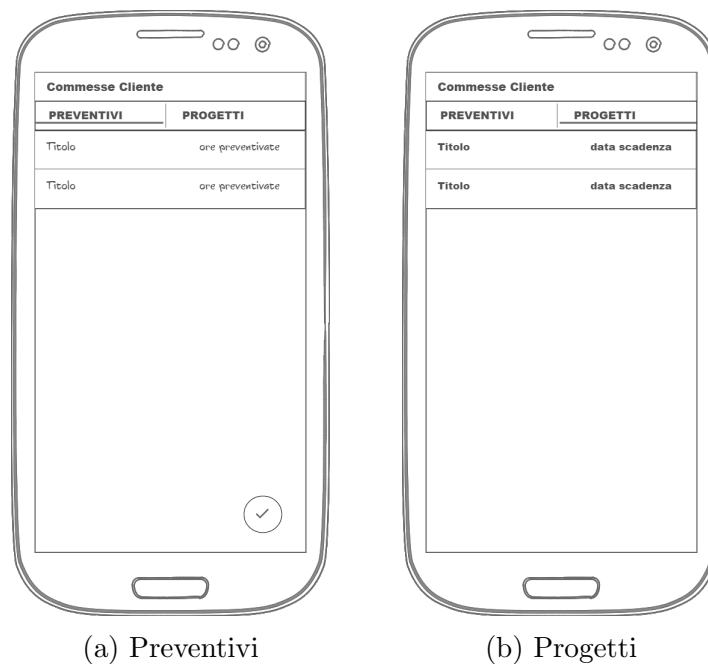


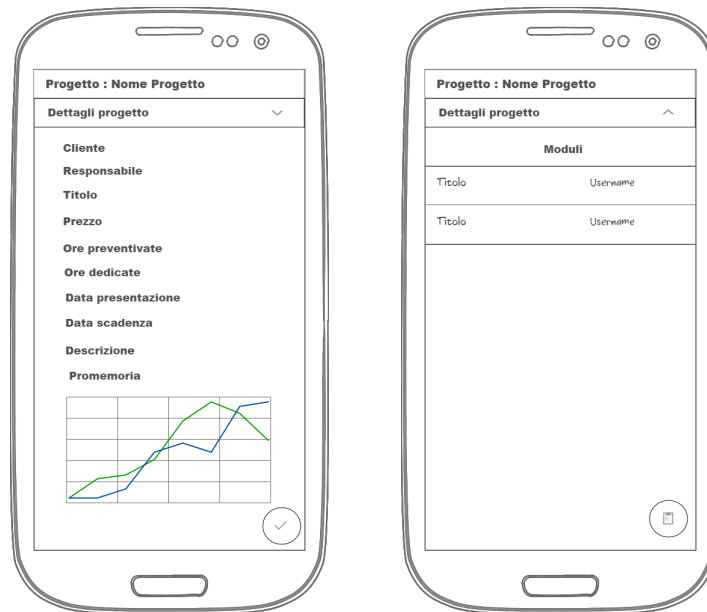
Figura 3.5: Mockup: Tabs Preventivi/Progetti

Rispettivamente nelle due figure (fig. 3.5) vengono mostrati i preventivi e i progetti associati al cliente selezionato. Per ogni preventivo è mostrato l'ammontare delle ore preventivate. Per ciascun progetto è invece resa visibile la data di scadenza.

Nell'immagine di figura (fig. 3.5a) è presente un Floating Action Button (FAB), che permette l'aggiunta di un preventivo relativo ad una commissione.

Selezionando un preventivo si avvia il procedimento di creazione del progetto, mentre selezionando un progetto viene visualizzata una schermata basata sui mockup in figura 3.6.

3.6.6 Progetto



(a) Progetto: dettagli

(b) Progetto: moduli

Figura 3.6: Mockup: dettagli e moduli progetto

Nel primo mockup (fig. 3.6a) sono resi visibili i dettagli del progetto e un FAB che permette la chiusura del progetto stesso. Nell'altro (fig. 3.6b) sono stati nascosti i dettagli in modo da rendere più facilmente visibile la lista dei moduli associati al progetto con i rispettivi responsabili. Anche qui è presente un FAB che permette l'aggiunta di moduli. Selezionando un elemento della lista si apre il corrispondente modulo.

3.6.7 Modulo

Quando un modulo viene aperto vengono visualizzate tre "pagine":

Dettagli: dove sono rappresentate le info del modulo selezionato ed il grafico delle ore dedicate allo sviluppo di esso. Il FAB presente permette la chiusura del modulo.

Report: dove viene reso visibile l'elenco di tutti i report associati al modulo.

Attività: dove si trova l'elenco delle attività correlate al modulo con l'indicatore della priorità. Il FAB permette l'aggiunta di un'altra attività al modulo. Se un'attività viene selezionata quest'ultima viene visualizzata.

3.6.8 Attività



Figura 3.7: Mockup: attività

Nel mockup di figura 3.7 vengono visualizzati i dettagli dell'attività. Premendo il FAB l'attività verrà chiusa. Premendo su "Visualizza Report" vengono mostrati tutti i report collegati all'attività con la possibilità di aggiungerne.

Capitolo 4

Implementazione

In questo capitolo sarà descritta l'implementazione dell'intero sistema, suddivisa in due parti: lato server e lato client.

4.1 Server

4.1.1 Database Management System e Manager

Sul DB tutti i dati sono formattati con l'URL Encoding in modo da poter essere codificati in modo uniforme ed avere interoperabilità verso altri sistemi.

Il DBMS utilizzato per gestire il DB è MySQL, che è un Relational Database Management System (RDBMS) rilasciato con licenza libera.

Come MySQLManager ho utilizzato phpMyAdmin, un'applicazione web scritta in PHP, ho scelto questo Manager perché è uno dei software più diffusi ed efficaci per amministrare MySQL.

4.1.2 Protocollo di comunicazione

HTTP (Hyper Text Transfer Protocol) è il protocollo che viene utilizzato per la comunicazione tra client e server sul World Wide Web. I metodi più utilizzati per passare i dati sono GET, che passa i parametri all'interno dell'indirizzo e POST che invece li passa all'interno del body della richiesta.

PHP è un linguaggio di programmazione object oriented interpretato, originariamente concepito per la programmazione di pagine web dinamiche.

In questo sistema le web pages sono pagine PHP che vengono utilizzate per interagire con il DB. Ogni pagina è relativa ad un servizio con la relativa query SQL.

SQL (Structured Query Language) è un linguaggio standardizzato per DB basati sul modello relazionale. SQL è stato progettato per:

- Creare e modificare schemi di database;
- Inserire, modificare e gestire dati memorizzati;
- Interrogare i dati memorizzati;
- Creare e gestire strumenti di controllo ed accesso ai dati.

Le comunicazioni tra lato server e lato client avvengono utilizzando un protocollo. Il protocollo di comunicazione ha 2 funzionamenti, le richieste da client a server con la sua formattazione e le risposte dal server al client con la sua formattazione.

Client-Server

Tutte le richieste da parte del client nei confronti del server verranno effettuate tramite HTTP con i dati inviati in POST.

Server-Client

Le web pages di risposta del server al client potevano essere strutturate con il formato JSON o con il formato XML.

```
{
  "success": true | false,
  "message": "",
  "data": null
}
```

Listato 4.1: JSON response

```
<answer>
  <success> true | false </success>
  <message> "" </message>
  <data> null </data>
</answer>
```

Listato 4.2: XML response

JSON è l'acronimo di JavaScript Object Notation, è un formato adatto allo scambio di dati tra client e server. È strutturato come una mappa, chiave - valore e questo lo rende facilmente implementabile in diversi linguaggi.

XML (EXtensible Markup Language) invece è un linguaggio marcatore che permette di creare tag personalizzati.

Le web pages sono state progettate in modo che restituiscano una risposta in formato JSON, in quanto avendo una scrittura più compatta delle informazioni si riduce l'impiego di risorse per l'invio dei dati.

Diagramma di sequenza illustra lo scenario del login, ovvero come avviene lo scambio dei dati all'interno del sistema utilizzando questo protocollo di comunicazione (fig. 4.1).

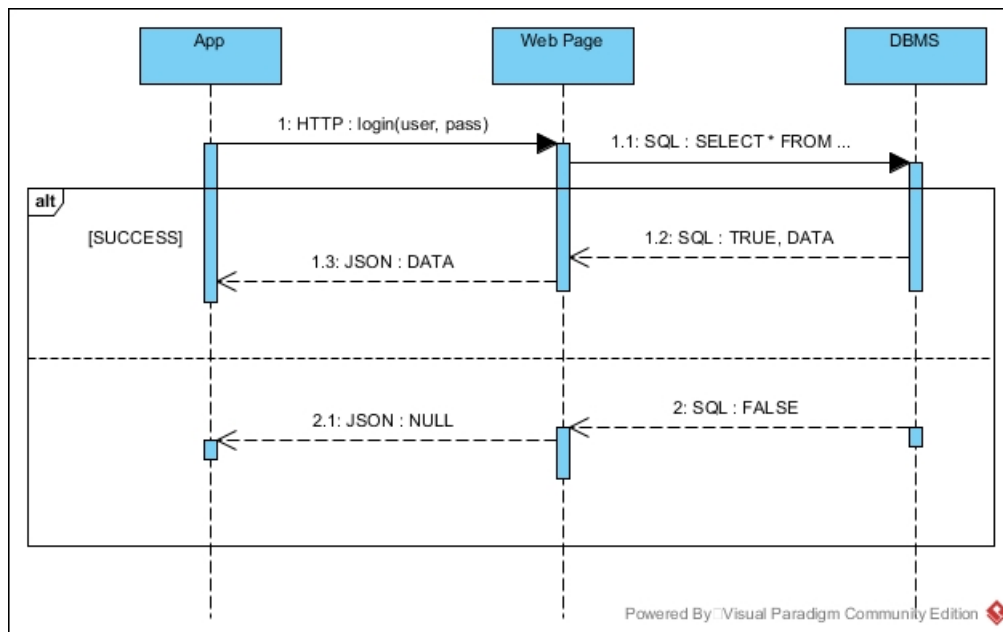


Figura 4.1: Diagramma di sequenza : login

4.1.3 Servizi

Vengono esposti di seguito i principali servizi con la loro implementazione.

login

Per effettuare il login, l'impiegato inserisce username e password poi premendo il pulsante di conferma viene richiamato il servizio a login.php.

I parametri passati in POST sono:

- user
- pass

I parametri user e pass sono rispettivamente lo username e la password dell'impiegato che vuole effettuare l'autenticazione. Di seguito un esempio di risposta alla richiesta di login al server.

```
{
  "success": false,
  "message": "username o password errati",
  "data": null
}
```

Listato 4.3: Login KO

```
{
  "success": true,
  "message": "login ok",
  "data": {
    "username": "root",
    "nome": "Matteo",
    "cognome": "Vitali",
    "telefono": "3349186920",
    "mail": "matteo.w.vitali%40gmail.com",
    "indirizzo": "Via+Brunelleschi+26",
    "ruolo": "0"
  }
}
```

Listato 4.4: Login OK

Se il login viene effettuato con successo il servizio restituisce in data un JSONObject con dentro gli attributi dell'impiegato che si è loggato. In caso di insuccesso invece data sarà avrà valore null.

getClienti

Per ottenere la lista di tutti i clienti viene utilizzato il servizio chiamato con `getClienti.php`. In POST non viene passato nessun parametro. La risposta a questo servizio è strutturata come mostrato sotto, dalla quale si può notare la differenza con il servizio precedente che in data aveva un `JSONObject`, mentre in questo c'è un `JSONArray`.

Da evidenziare che il servizio "fallisce" quando non esistono clienti nel DB.

```
{
  "success": true,
  "message": "getClienti ok",
  "data": [
    {
      "id_cliente": "3",
      "nome": "Francesco",
      "cognome": "Lelli",
      "telefono": "3339186990",
      "mail": "francesco%40gmail.com",
      "indirizzo": "Viale+Randi+11"
    }
  ]
}
```

Listato 4.5: getClienti OK

setReport

Ogni qualvolta un'impiegato dell'azienda effettua un report viene richiamato il servizio `setReport`.

Il server in POST riceve 4 dati:

- `id_attivita`
- `username`
- `data`
- `ore_dedicare`

Rispettivamente questi parametri servono per indicare su quale attività sono state svolte delle ore lavorative, in quale giorno e da quale impiegato. Nel DB

le ore lavorative, vengono gestite in minuti, visto che risultava più efficace in quanto è stato richiesto di gestire anche le mezze ore. Questo servizio oltre ad aggiungere un report imposta lo stato dell'attività, del modulo e del progetto ad esso collegato in "in corso" .

La risposta di questo servizio restituisce principalmente se ha avuto o meno successo.

getReportProgetto

Nei dettagli del progetto, per la creazione del grafico sono necessari tutti i report relativi al determinato progetto. Di conseguenza l'unico parametro passato in POST per `getReportProgetto` è l'id del progetto in modo da poterlo identificare univocamente.

Un esempio di risposta a questo servizio è visibile nel codice sottostante. Si può notare che in data c'è un `JSONArray` con tutti i report richiesti ordinati in modo decrescente. Nel caso non ci siano report collegati al progetto il servizio restituisce insuccesso con `data = "null"`.

```
{
  "success": true,
  "message": "report modulo restituite",
  "data": [
    {
      "id_attivita": "2",
      "id_report": "39",
      "data": "2015-11-02",
      "username": "root",
      "ore_dedicare": "90"
    },
    {
      "id_attivita": "1",
      "id_report": "38",
      "data": "2015-10-26",
      "username": "bell",
      "ore_dedicare": "30"
    }
  ]
}
```

Listato 4.6: `getReportProgetto`

4.2 Applicazione

4.2.1 Tipologie di app

Le applicazioni per mobile possono essere suddivise in relazione alle tecnologie su cui si basano:

- Applicazioni native: app sviluppate per una specifica piattaforma. Queste applicazioni hanno il vantaggio che possono interagire direttamente con il device e ciò porta ad un aumento delle prestazioni. Uno svantaggio è dato dal fatto che non sono direttamente portabili su altre piattaforme;
- Web app: pagine Web ottimizzate per dispositivi mobili sfruttando tecnologie Web come HTML5, JavaScript e CSS3. I vantaggi legati a questo tipo di sviluppo sono la facilità e l'accorciamento dei tempi di scrittura;
- App ibride: sono un "mix" delle due categorie precedenti, infatti sono scritte con tecnologie web e vengono eseguite localmente all'interno dell'applicazione nativa. Un vantaggio importante è dato dalla più facile portabilità su altre piattaforme. Negli svantaggi troviamo una riduzione delle prestazioni, come nel rendering grafico e nell'accesso alle risorse locali.

4.2.2 Mobile OS

Nel mondo del mobile i protagonisti incontrastati sono 3 giganti: iOS di Apple, Android di Google e Windows Phone di Microsoft.

Android è un sistema operativo open source per smartphone, tablet e dispositivi mobili in generale, sviluppato da Google. Con l'SDK (Software Developer Kit) messo a disposizione consente in modo relativamente semplice e potente di progettare e programmare applicazioni. Android dunque non è un linguaggio di programmazione ma un vero e proprio insieme di strumenti e librerie.

Andy Rubin, Rich Miner, Nick Sears e Chris White nel 2003 fondarono un'azienda, Android Inc. L'obiettivo era quello di creare un sistema operativo aperto, basato su Linux, con un'interfaccia semplice e funzionale che mettesse a disposizione degli sviluppatori strumenti efficaci per la creazione di applicazioni. Nel 2005 la svolta con l'acquisto da parte di Google e successiva fu la fondazione nel novembre dello stesso anno della Open Handset Alliance (OHA). L'Open Handset Alliance, capeggiata da Google, contava trentacinque membri fra cui alcuni operatori telefonici, produttori di dispositivi mobili, produttori di semiconduttori, compagnie di sviluppo software e di commercializzazione. Il 5 novembre 2007, l'Open Handset Alliance presenta il sistema

operativo Android. Qualche giorno dopo viene rilasciato anche il primo Software Development Kit (SDK) per gli sviluppatori che include: gli strumenti di sviluppo, le librerie, un emulatore del dispositivo, la documentazione, alcuni progetti di esempio e un tutorial.

iOS è il sistema operativo sviluppato da Apple basato su Unix. Deriva da OSX il sistema operativo per i computer Mac e viene utilizzato per gli iPhone, iPad ed iPod Touch.

Nel 2007 il sistema operativo è stato presentato ed ancora privo di nome, è entrato in commercio con il primo iPhone il 29 giugno dello stesso anno. In concomitanza con la pubblicazione della prima beta del SDK, nel 2008 il sistema operativo è stato denominato ufficialmente come "iPhone OS". Il quarto rilascio del sistema operativo nel 2010, ha aggiunto numerose funzioni e l'ora rinominato "iOS", ha unificato i vari dispositivi con una versione comune.

Windows Phone è la famiglia di sistemi operativi sviluppati da Microsoft per gli smartphone.

4.2.3 Diffusione e distribuzione

Nella figura sottostante possiamo vedere la diffusione dei dispositivi mobile per sistema operativo.

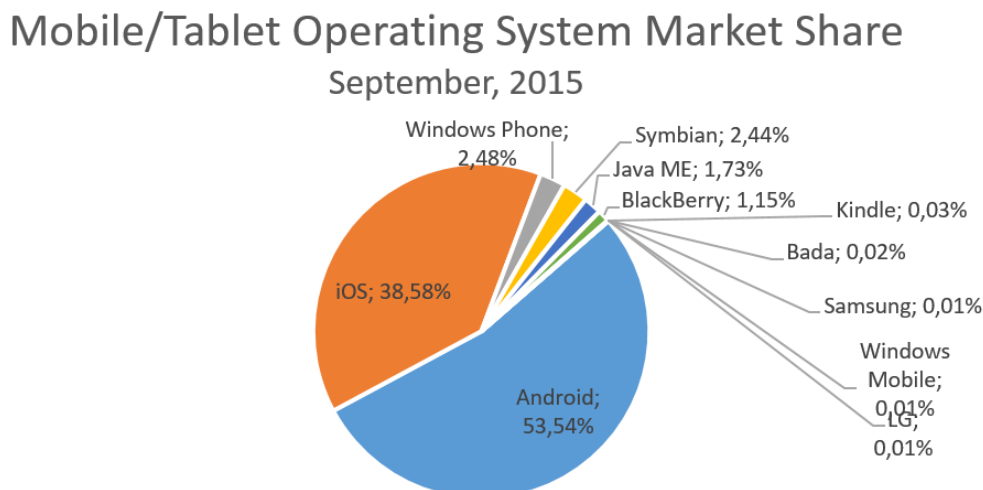


Figura 4.2: Quota di mercato Mobile/Tablet per O.S.

4.2.4 Ambiente d'uso e limitazioni

Il sistema dovrà essere utilizzabile da dispositivi mobile. La scelta tra le diverse tipologie di applicazioni è ricaduta sulle app native in quanto più efficienti nella gestione delle risorse e del rendering grafico. Lo svantaggio correlato a questa scelta è la necessaria implementazione di tante applicazioni quanti gli OS.

Dopo aver deciso la tipologia dell'app è stato scelto quale OS sviluppare per primo. La decisione è ricaduta su Android per la sua più ampia diffusione rispetto agli altri. L'SDK minimo impostato è il 16 che corrisponde alla versione 4.1 di android cioè la JELLY BEAN in modo da coprire il 92.7% dei dispositivi android.

4.2.5 Linguaggio e piattaforma di sviluppo

L'IDE (integrated development environment) ufficiale per lo sviluppo di applicazioni di Android prodotto da Google è: Android Studio; che è basato su IntelliJ IDEA, uno dei migliori IDE per Java.

4.2.6 Librerie

Attorno ad Android è nata una comunità di sviluppatori che produce una vasta gamma di librerie. Al giorno d'oggi non è necessario reinventare la ruota ed allo stesso modo non è indispensabile scrivere del codice se esiste già una libreria open source che produce l'effetto desiderato.

Per comprendere al meglio l'utilizzo delle librerie verranno utilizzati degli screenshot ed introdotti due concetti fondamentali di android.

Activity è una classe Java che, insieme ad un file di layout XML che rappresenta e gestisce una View dell'applicazione; occupandosi quindi sia della View sia del Controller.

Fragment è un componente che viene eseguito sempre nel contesto di una Activity ma rimane indipendente per quanto riguarda il ciclo di vita e l'interfaccia utente; hanno un proprio ciclo di vita e una loro interfaccia utente.

Floating action button

Le librerie prese dal web sono state inserite nel progetto dell'applicazione aggiungendo delle dipendenze al gradle, come in questo caso.

```
dependencies {  
    compile 'com.melnykov:floatingactionbutton:1.3.0'  
}
```

Listato 4.7: library

Il FAB è un bottone che segue le linee guida del material design. Quando si scorre la pagina, alla quale il FAB è associato, questo si "nasconde", per poi ricomparire quando la pagina torna nella posizione iniziale. Ogni FAB è personalizzabile, cambiando il colore, l'icona, la grandezza; permettendo di avere un'applicazione custom anche utilizzando una libreria.

Network

L'azienda ha fornito una libreria, contenente due classi `ServerResponse` e `NetworkSendDataItem` che vengono utilizzate per effettuare le richieste HTTP e ricevere la risposta. Al fine di non bloccare il thread principale, la richiesta al server viene effettuata in un `Async Task`, che permette di effettuare l'operazione in un thread diverso dal main thread; questo è possibile grazie ad un'altra classe della libreria, `NetworkSendAsyncTask`.

Tutti i servizi vengono gestiti tramite queste classi. Per una spiegazione più esaustiva della libreria utilizziamo come esempio il servizio del login come si può vedere nel codice sottostante.

```
private static final String URL_LOGIN = URL_BASE + "login.php";  
  
public static NetworkSendDataItem getImpiegato(String user, String  
    pass) {  
    NetworkSendDataItem item = new NetworkSendDataItem(URL_LOGIN,  
        NetworkItem.NetworkMethod.POST);  
    try {  
        item.putItem("username", URLEncoder.encode(user, UTF_8));  
        item.putItem("password", URLEncoder.encode(pass, UTF_8));  
    } catch (UnsupportedEncodingException e) {  
        e.printStackTrace();  
    }  
    return item;  
}
```

Listato 4.8: NetworkSendDataItem

Nel codice sopra viene creata la richiesta al servizio del login e vengono inseriti username e password in POST.

```
if (NetworkUtils.isOnline(LoginActivity.this)) {
    new NetworkSendAsyncTask(getImpiegato(user, pass),
        new ServerResponse<JSONObject>() {
            @Override
            public void onError(String message) {
                /*
                 la richiesta non e' andata a buon fine
                 stampa l'errore
                */
            }

            @Override
            public void onSuccess(JSONObject data, String message) {
                /*
                 la richiesta e' andata a buon fine
                 controlla la risposta contenuta nel JSONObject
                */
            }
        }).execute();
} else {
    /*il dispositivo non e' online*/
}
```

Listato 4.9: NetworkSendAsyncTask

Nella codice soprastante si può vedere la chiamata al servizio e la gestione della risposta da parte del client.

Material dialogs e Material edit text

Anche il testo e la sua formattazione con l'avvento del material design ha subito qualche miglioria.

Dialog sono piccole finestre modali che richiedono all'utente di prendere una decisione o inserire informazioni aggiuntive. Nell'applicazione vengono utilizzate ad esempio quando vengono chiusi i progetti (fig. 4.3a).

Edit text sono dei campi dove viene inserito del testo. I campi di testo possono avere diversi tipi di input, come il numero, la data, la password o l'indirizzo email (fig. 4.3b).

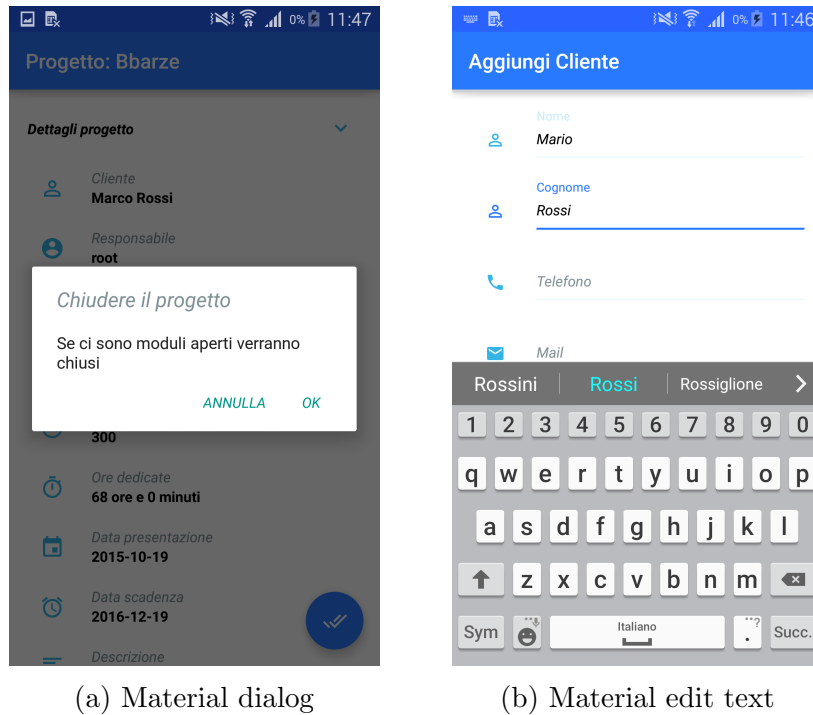


Figura 4.3: Screenshot: Material dialog e Material edit text

Chart

Per disegnare i grafici delle ore lavorative è stata utilizzata una libreria presa da internet. Questa libreria permette di disegnare svariati tipi di grafico. Per questa applicazione sono stati usati i grafici a linee in modo da far visualizzare le ore lavorative svolte giorno per giorno (fig. 4.4).

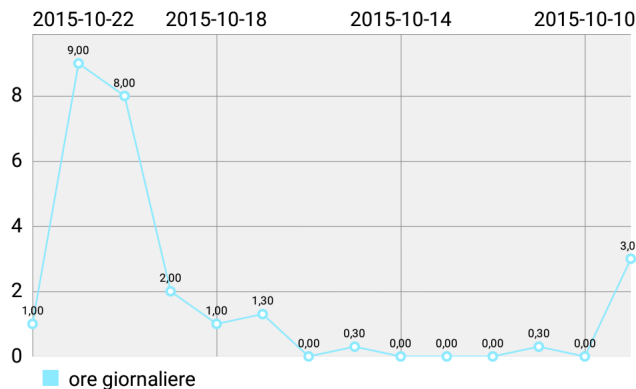


Figura 4.4: Screenshot: Chart

Tabs

Le sliding tabs (fig. 4.5) permettono la visualizzazione di più pagine affiancate l'una all'altra. All'interno del progetto vengono utilizzate in due casi differenti: sia visualizzare i preventivi e progetti di un determinato cliente sia i dettagli di un modulo con le relative attività ed il rispettivo report.

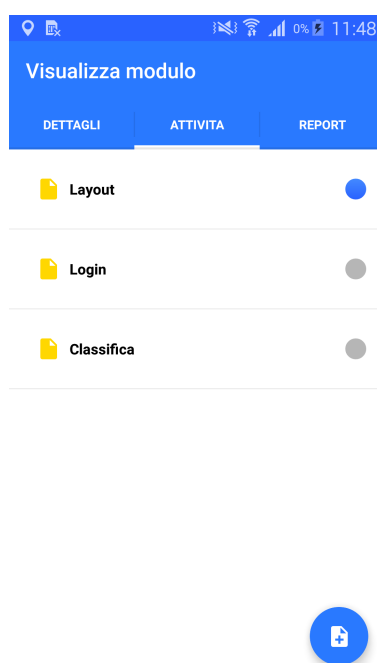


Figura 4.5: Screenshot: Sliding tabs

Navigation drawer

La libreria che gestisce il navigation drawer (fig. 4.6) semplifica e rende più comprensibile il codice. La libreria utilizzata infatti permette la gestione di tutte le opzioni del menu singolarmente. Per ogni possibilità si può impostare la grafica della voce del menu e l'azione collegata nel caso venga selezionata.

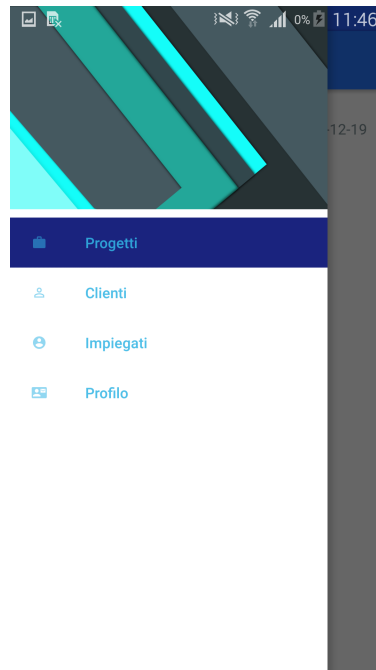


Figura 4.6: Screenshot: Navigation drawer

4.2.7 Package

Il progetto per una migliore organizzazione del codice è stato suddiviso in package come in figura 4.7. All'interno di ogni package ci sono tutte le activity e i fragment relativi ad una determinata entità; fatta eccezione per Utils, che contiene classi di utilità e Tabs che gestisce lo sliding tab.











- ▶  Attivita
- ▶  Cliente
- ▶  Impiegato
- ▶  Modulo
- ▶  Preventivo
- ▶  Progetto
- ▶  Report
- ▶  Tabs
- ▶  Utente
- ▶  Utils

Figura 4.7: Package

4.2.8 Pattern utilizzati

Template method

Definisce lo scheletro di un algoritmo, lasciando l'indicazione di alcuni suoi aspetti alle sottoclassi.

Strategy

Definisce una famiglia di algoritmi, e li rende intercambiabili, ossia usabili in modo trasparente dai loro clienti.

MVC

Consiste nella separazione dei compiti fra i componenti software che interpretano tre ruoli:

- il model contiene i dati;
- la view visualizza i dati contenuti nel model e si occupa dell'interazione con gli utenti;
- il controller riceve i comandi dell'utente attraverso la view e li attua modificando lo stato degli altri due componenti.

Singleton

Garantisce che una classe abbia una unica istanza, accessibile globalmente.

Diagramma delle classi: Tab

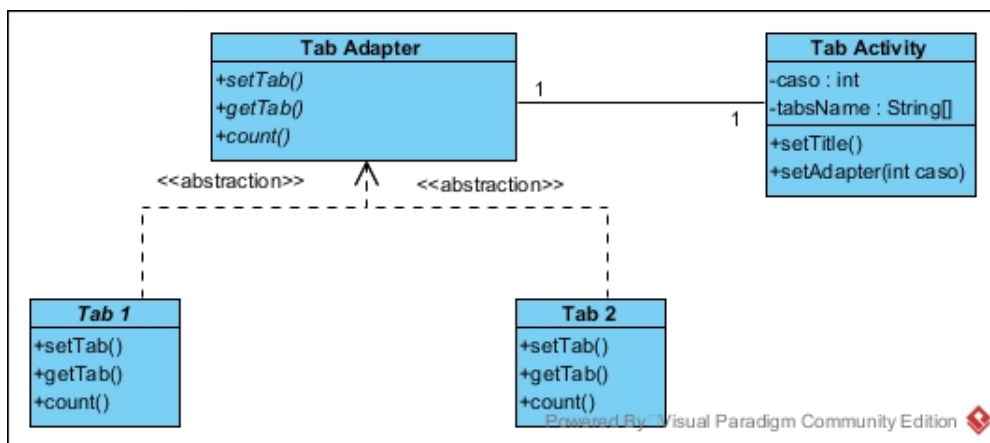


Figura 4.8: Diagramma delle classi: Tab

Dal diagramma soprastante (fig. 4.8) si evince l'utilizzo del pattern Template Method, infatti Tab Adapter è una classe astratta e viene estesa sia da Tab1 sia da Tab2. Tab Activity in base al caso definito utilizzerà un Tab Adapter, cioè o Tab1 o Tab2. L'utilizzo di questo pattern evita la riscrittura di codice e rende facile la scalabilità.

Diagramma delle classi: Progetto

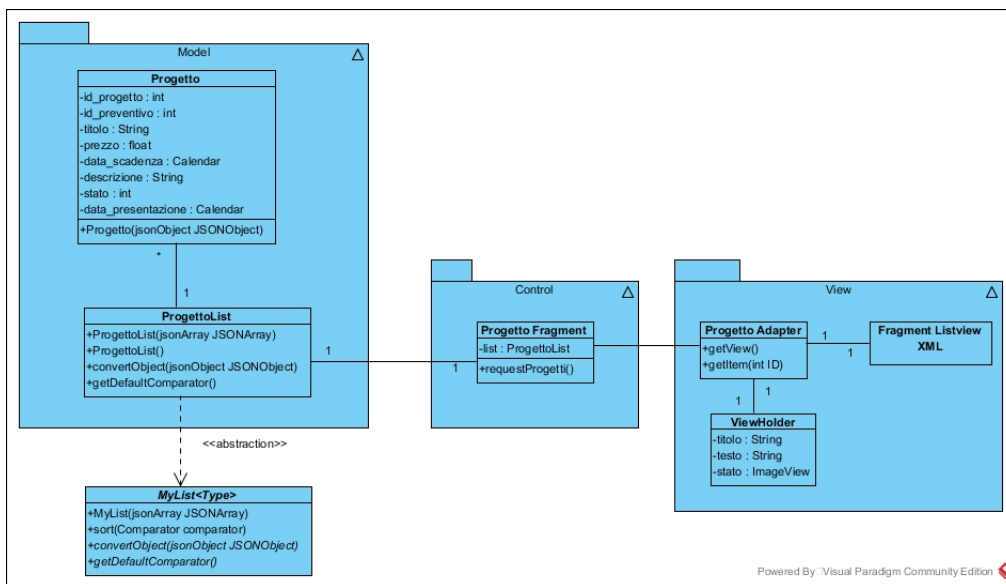


Figura 4.9: Diagramma delle classi: Progetto

In questo diagramma (fig. 4.9), che rappresenta parte delle classi del package Progetto, è possibile intravedere l'utilizzo di più pattern:

- MVC organizza al meglio il codice suddividendolo in base allo scopo. Il model gestisce i dati scaricati del server e la lista a questi corrispondenti, la view compone tutta l'interfaccia con l'utente e il controller coordina sia il model sia la view.
- Template method è stato pensato ed utilizzato perché in molti package è presente l'utilizzo di una lista di elementi, quindi è stata creata **MyList**, una classe astratta che viene poi estesa da **ProgettoList**.
- Strategy permette di implementare l'ordinamento della lista già in **MyList**. Il processo verrà poi effettivamente delegato alla lista "specializzata", in questo caso **ProgettoList**.

Diagramma delle classi: Promemoria

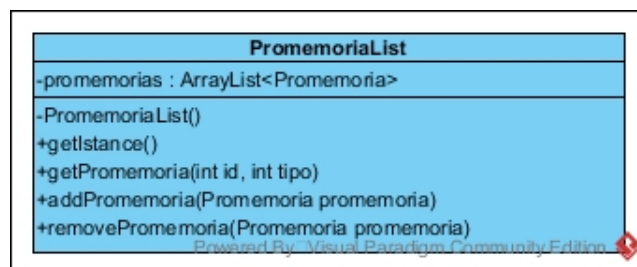


Figura 4.10: Diagramma delle classi: Promemoria

In questo diagramma (fig. 4.10) si nota l'utilizzo del pattern Singleton che permette la creazione di una sola istanza della classe PromemoriaList, in questo modo si evita la concorrenza per il recupero e salvataggio dei dati.

4.2.9 Funzionalità implementate

Di seguito vengono spiegate alcune delle principali funzionalità implementate.

Sicurezza

Per la crittografia vengono utilizzate funzioni di hash, cioè funzioni che mappano un messaggio arbitrariamente lungo in una stringa di lunghezza prefissata, in modo da non poter risalire al messaggio che l'ha generata. La lunghezza della stringa finale è direttamente correlata con la sicurezza della funzione di hash, perché più una stringa è lunga, minore sarà la probabilità di trovare due messaggi con la stessa stringa finale. Per la crittografia delle password in questo sistema sono stati presi in considerazione due tipi di codifica SHA-2 e MD5, che restituiscono rispettivamente 256 bit il primo e 128 bit il secondo. La scelta è ricaduta su SHA-2 in quanto più robusto.

Sessione

Quando si esegue il login con successo, lo username viene salvato in locale attraverso le Shared Preferences. Successivamente ogni qualvolta l'applicazione verrà avviata non sarà più necessario autenticarsi con username e password, perché il dispositivo dopo un controllo sul DB accederà automaticamente. Il logout dall'applicazione va a cancellare lo username dalle Shared Preferences.

Trasformazione dati grafico

I dati rappresentati nei grafici sono estrapolati dai report. In profilo ad esempio viene visualizzato il grafico di tutte le ore lavorative effettuate dall'impiegato loggato. Il servizio apposito recupera l'elenco di tutti i report associati allo username, in seguito l'elenco verrà compattato unendo tutti i report effettuati lo stesso giorno, per poi aggiungere i giorni in cui le ore lavorative sono pari a zero; a questo punto i dati vengono dati in gestione alla libreria che gestisce il grafico e quest'ultimo viene renderizzato.

Promemoria e notifiche

I promemoria sono personali, ogni impiegato può impostare un promemoria per ogni progetto e uno per ogni modulo. Per realizzare questa funzionalità, i promemoria vengono salvati sul dispositivo, in modo da non occupare spazio sul DB, perché nel caso contrario sarebbe stato necessario salvare tutti i promemoria di tutti gli impiegati. Nel dispositivo viene quindi tenuto un file ordinato per scadenza contenente tutti i promemoria associati dallo username ai vari moduli e progetti. Ogni minuto l'applicazione, attraverso l'Alarm Manager controlla all'interno del file se c'è un promemoria da notificare, e nel qual caso ci fosse lo notifica al sistema per poi cancellarlo dal file.

Capitolo 5

Testing

In questo capitolo viene mostrato il test effettuato con i dispositivi. I test sono eseguiti variando il carico di lavoro.

5.1 Dispositivi in uso

In base ai dispositivi utilizzati ci si aspetta di vedere differenti risultati, causate dalle differenti prestazioni degli smartphone. Il primo utilizzato per effettuare i test è un Samsung Galaxy S5 con la versione di Android 5.0 mentre il secondo è un HUAWEI Y300-0100 con la versione di Android 4.1.1.

5.2 Tempi di latenza

In quanto la maggior parte dei dati viene salvata online, per ogni operazione viene introdotto un ritardo causato dal tempo che il dispositivo impiega a richiedere al Server un certo servizio e dal tempo che impiega poi quest'ultimo a rispondere.

Questo test è stato effettuato sul servizio `getModuli`, che si occupa di richiedere al server tutti i moduli collegati ad un determinato progetto. Sono stati impostati due scenari:

Nel primo scenario sono stati associati ad un unico progetto 2500 moduli, ovviamente è uno scenario surreale ma permette di vedere le seguenti differenze. La latenza media del primo device è di 1 secondo e mezzo, mentre per il secondo smartphone si aggirava sui 3 secondi.

In uno scenario quotidiano, con ad esempio una decina di moduli associati ad un unico progetto, la latenza dei due dispositivi è pressoché uguale ed è poco inferiore a cinque decimi di secondo.

Conclusioni

Il progetto di tesi conseguito è stato estremamente istruttivo, perché mi ha permesso di continuare un percorso in azienda iniziato col tirocinio formativo addentrandomi più a fondo nell'ambiente lavorativo.

Il sistema è stato sviluppato secondo i requisiti richiesti, sviluppando un'applicazione in Android, rimane da effettuare lo sviluppo dell'applicazione equivalente per IOS e Windows Phone.

Le conoscenze acquisite durante le lezioni universitarie hanno svolto un ruolo importante per lo sviluppo del sistema, in particolare Programmazione di Sistemi Mobile per ovvie ragioni, Basi di Dati per la gestione del Server e Programmazione a Oggetti per il paradigma a oggetti e l'utilizzo dei vari design pattern.

Sviluppi Futuri

Come sviluppi futuri si potrebbe pensare di unire il sistema da me progettato e implementato con il sistema creato da Anna Giulia, una collega in azienda. Il suo sistema gestisce tutta la procedura di comunicazione tra un'azienda fornitrice di servizi e i suoi clienti, tramite un'applicazione mobile. Questa unione sarebbe molto produttiva perché permetterebbe di controllare il progetto in seguito al beta testing.

Tra le funzionalità che si possono inserire ci sono:

- la creazione di gruppi di lavoro;
- la possibilità di creare più progetti da un unico preventivo;
- la possibilità di associare più promemoria ai moduli e ai progetti, personalizzabili dal singolo impiegato.

Al giorno d'oggi i sistemi informatici sono in continua evoluzione e l'unico limite è l'immaginazione, quindi i possibili sviluppi futuri sono infiniti.

Ringraziamenti

Innanzitutto desidero ringraziare il Dott. Mirko Ravaioli per avermi permesso di svolgere la tesi in azienda, dove con tutti gli impiegati si è instaurato un bel rapporto costruttivo e collaborativo.

Doverosi ed altrettanto sentiti sono i ringraziamenti che rivolgo agli amici che ho conosciuto in questo percorso iniziato tre anni fa: Bajram, Anna Giulia, Samantha, Francesco, Andrea e Francesco. Con loro ho passato momenti di studio, "matto e disperatissimo" per citare Leopardi, alternati a momenti di svago, come quando al termine delle lezioni ci recavamo a giocare a frisbee. Grazie a voi, che avete reso le lezioni e gli interminabili viaggi in pullman sempre nuovi ed interessanti.

Ringrazio anche Raffaele (detto Lello) che mi ha spinto a frequentare l'università quando già cercavo lavoro, senza di te questo cammino non sarebbe neanche iniziato.

Infine, per ultimi, ma non per importanza ringrazio la mia famiglia e tutti i miei amici che mi sono sempre stati accanto, sostenendomi e riempiendo di allegria le mie giornate.

Elenco delle figure

2.1	Diagramma dei casi d'uso	4
2.2	Diagramma delle attività	5
3.1	Schema logico dell'architettura a 3 strati Client - Server	7
3.2	Schema E/R	8
3.3	Schema logico	10
3.4	Mockup: login, Navigation Drawer	12
3.5	Mockup: Tabs Preventivi/Progetti	13
3.6	Mockup: dettagli e moduli progetto	14
3.7	Mockup: attivita	15
4.1	Diagramma di sequenza : login	19
4.2	Quota di mercato Mobile/Tablet per O.S.	24
4.3	Screenshot: Material dialog e Material edit text	28
4.4	Screenshot: Chart	28
4.5	Screenshot: Sliding tabs	29
4.6	Screenshot: Navigation drawer	30
4.7	Package	30
4.8	Diagramma delle classi: Tab	31
4.9	Diagramma delle classi: Progetto	32
4.10	Diagramma delle classi: Promemoria	33

Bibliografia

- [1] Wallace Jackson, *Learn Android App Development*, Apress, 2014-2015.
- [2] Mirko Ravaioli, *Dispense programmazione di sistemi mobile*, 2014-2015.
- [3] Paola Salomoni, Silvia Mirri, Catia Prandi, *Dispense sistemi multimediali*, 2014-2015.
- [4] Stefano Rizzi, *Dispense ingegneria del software*, 2014-2015.
- [5] Mirko Viroli, Andrea Santi, Danilo Pianini *Dispense programmazione ad oggetti*, 2013-2014.
- [6] Dario Maio, Annalisa Franco, *Dispense basi di dati*, 2013-2014.
- [7] Wikiversità, *Funzioni di hash*, https://it.wikiversity.org/w/index.php?title=Funzioni_di_hash&oldid=93636, 26 maggio 2015 14:54 UTC.
- [8] Android Open Source Project, *Dashboards*, <https://developer.android.com/about/dashboards/index.html>, 20 novembre 2015 09:10 UTC.
- [9] Wikipedia, *Organizzazione aziendale*, https://it.wikipedia.org/w/index.php?title=Organizzazione_aziendale&oldid=75837274, 20 ottobre 2015 12:47 UTC.
- [10] Simone D'Amico, *Guida PHP*, <http://www.html.it/guide/guida-php-di-base/>, 20 novembre 2015 09:10 UTC.
- [11] Net Market Share, *Market Share Statistics for Internet Technologies*, <https://netmarketshare.com/>, 20 novembre 2015 09:10 UTC.
- [12] Evoluzioni Web Società Cooperativa, *App: Native, Ibride o Web?*, http://www.evoluzionieweb.it/IT/Mobile_App_Native_Ibride_Web, 20 novembre 2015 09:10 UTC.

- [13] Dott. Bruno Salvatore Belluccia, *Introduzione allo sviluppo Android*, <http://www.sbengine.com/edu.php>, 20 novembre 2015 09:10 UTC.
- [14] Dott. Bruno Salvatore Belluccia, *Introduzione allo sviluppo iOS*, <http://www.sbengine.com/edu.php>, 20 novembre 2015 09:10 UTC.