Progetto di Ingegneria del Software 2022/23
Università Ca' Foscari Venezia

Piano di progetto

2.0

0-Budget

15/01/2023

Document Informations

NomeProgetto	Bookito!	Acronimo	BKT	
Derivable	Piano di progetto			
Data di consegna	15/01/2023			
Team Leader	Giulia Cogotti			884383@stud.unive.it
Team Members	Diego Passarella Marco Chinellato Marta Cazzin Jinpeng Zhang Tudor Croitor			886711@stud.unive.it 886217@stud.unive.it 883396@stud.unive.it 886854@stud.unive.it 886005@stud.unive.it

Document History

Version	Issue Date	Stage	Changes	Contributors
1.0	17/10/2023	Draft	Prima stesura del documento	Giulia Cogotti Diego Passarella Marco Chinellato
2.0	15/01/2023	Final	Allineamento documentazione	Giulia Cogotti

Indice

1. Introduzione

1.1. Overview del progetto

Il documento in questione illustra il piano di progetto, ovvero la pianificazione del lavoro che il gruppo dovrà svolgere per portare a termine la creazione dell'applicazione Android.

Il progetto ha come obiettivo la progettazione e la seguente creazione di un'applicazione android rivolta agli utenti più fragili appartenenti a un certo quartiere, in particolare viene scelta la provincia di Venezia.

L'idea proposta è quella di creare un'applicazione che chiameremo *Bookito!* la quale ha lo scopo di favorire lo scambio e il prestito di libri usati. Il tutto con uno sguardo alle interazioni sociali e al coinvolgimento di persone economicamente fragili. Ovviamente utilizzabile anche da altri utenti non fragili i quali vorrebbero fornire un aiuto concreto verso il prossimo.

I libri potranno essere facilmente inseriti tramite la scannerizzazione o l'inserimento dell'isbn, al termine con successo dell'operazione essi saranno visibili nella propria libreria personale e gli altri utenti potranno richiederli tramite l'apposita funzione di ricerca.

Una situazione problematica è dettata dal prestito di un libro tra due utenti. Quando questo accade essi devono necessariamente concordare una data di restituzione, ovviamente salvata su database, tramite una chat creata al momento dell'accettazione della richiesta.

Nel giorno prestabilito, i due utenti devono confermare l'avvenuta restituzione del libro, dove il proprietario di quest'ultimo dovrà segnalare le condizioni in cui esso è stato restituito. Il proprietario del libro può inoltre segnalare la mancata restituzione (o il suo ritardo).

In caso di utilizzo dell'applicazione da parte di un minorenne, che per legge potrà iscriversi solo ha compiuto almeno 13 anni, si dovrà pensare allo sviluppo di un sistema di "*controllo parentale*" in modo che gli incontri avvengano tramite o insieme all'adulto responsabile del minore.

Quando si utilizza un'applicazione che comporta scambi e prestiti, il comportamento del singolo utente è fondamentale nella creazione di un clima di fiducia e sicurezza per ognuno. Lo sviluppatore del codice potrà al massimo forzare gli utenti verso quella direzione. In bookito questo verrà reso possibile grazie ad un sistema di *red flag* che segnalerà, sulla base delle recensioni ricevute, se l'utente si comporta bene o male (i.e. restituisce il libro in pessime condizioni). In questo modo il problema verrà mitigato.

1.2. Deliverables del progetto

La seguente sezione si occupa di definire i documenti che verranno consegnati con conseguente data.

ID	Titolo derivable	Data di consegna
–	Definizione dei gruppi di lavoro	26/09/2022
D0	Proposta iniziale	03/10/2022
D1	Piano di progetto	17/10/2022
D2	Documento dei requisiti	31/10/2022
D3	Piano di testing	14/11/2022
D4	Documento di progettazione	28/11/2022
D5	Versione 1.0 del codice sorgente	15/12/2022
D6	Versione 2.0 del codice e allineamento documentazione	15/01/2023

1.3. Evoluzione del progetto

Il progetto ha il fine di conseguire un esame universitario. Se, al termine di esso, il team continuasse a collaborare ampliando ulteriormente l'applicazione creata, le funzionalità che essa fornirebbe aumenterebbero. In particolare di seguito vengono proposte alcune idee:

- In caso di molta affluenza da parte degli utenti, sarebbe utile una "*collaborazione*" con librerie o biblioteche della provincia di Venezia, in modo da aumentare i libri a disposizione e il controllo.
- Per incentivare l'utilizzo corretto dell'applicazione e aumentare la fiducia degli utenti, si potrebbe introdurre un sistema di "*punteggi/coins*", nel quale gli utenti guadagnano un certo punteggio effettuando le operazioni supportate (prestito, scambio, regalo). In particolare assumendo un corretto comportamento l'utente avrà un punteggio maggiore, in caso contrario sarà minore. Un parametro di valutazione di esempio potrebbe essere quello della restituzione del libro in buone condizioni e secondo la data prestabilita.
- In caso di interesse da parte dell'utenza si potrebbe introdurre una sezione "*Booklist*", in cui l'utente può avere una lista di tutti i suoi libri letti ed assegnar loro un voto con anche (possibilmente) una piccola recensione. La booklist potrà essere sia pubblica, che privata. In caso fosse pubblica potrebbe fornire un aiuto ad altri utenti nella scelta dei libri.
- Una funzionalità interessante sarebbe l'inserimento di annunci per richieste di libri non disponibili, in modo da trovare qualche utente disposto a inserirlo.
- Infine, si potrebbe pensare ad un eventuale ampliamento del range dell'area geografica in cui viene utilizzata l'applicazione.

1.4. Glossario

Il glossario comprende la definizione di tutti i termini tecnici utilizzati all'interno del documento, saranno elencati solamente quelli nuovi, evitando dunque ripetizioni con i deliverables precedenti.

- **Android Studio:** ambiente di sviluppo utilizzato per creare applicazioni per Android
- **APK:** *Android Application Package*. Si tratta di un formato di file compresso dove al suo interno sono memorizzati tutti i componenti necessari per l'installazione e il corretto funzionamento di un'applicazione su un dispositivo. In particolare si tratta del formato utilizzato per installare applicazioni Android
- **Backup:** In informatica si indica con la messa in sicurezza di informazioni di un sistema, da utilizzare come recupero (ripristino) in caso di danni accidentali
- **IDE**: *Integrated Development Environment* (Ambiente di sviluppo integrato) è un software che in fase di programmazione, supporta i programmatori nello sviluppo e debugging del codice sorgente di un programma
- **Discord**: Applicazione che permette la comunicazione orale e testuale fra molteplici utenti
- **Java:** Linguaggio di programmazione orientato ad oggetti
- **Kotlin:** Linguaggio di programmazione atto allo sviluppo di applicazioni android
- **LaTeX**: Linguaggio di markup per la scrittura di documenti
- **Overleaf**: Web app che supporta la scrittura di documenti in latex condivisibili con altri utenti
- **Smart-working**: è una particolare modalità di esecuzione della prestazione di lavoro subordinato introdotta al fine di incrementare la competitività e di agevolare la conciliazione dei tempi di vita e lavoro

2. Organizzazione del progetto

2.1. Modello del processo

Il modello scelto per gestire le relazioni tra le varie fasi del processo è il cosiddetto "*modello evolutivo*" il quale permette la suddivisione del lavoro partendo da una descrizione di massima di arrivare al risultato finale, attraversando le fasi intermedie. Esso dona flessibilità al progetto con il rischio accettabile di non sapere a che punto si è arrivati di preciso.

Inoltre verrà in parte utilizzato un modello di processo basato sul *riutilizzo*, il quale permette di sfruttare librerie di codice già esistenti.

Un ulteriore modello che verrà usato è quello di sviluppo agile, in particolare la sua funzionalità di *pair programming* sfruttando i vantaggi della collaborazione lavorativa.

2.2. Struttura organizzativa

La comunicazione tra i membri del gruppo avverrà nei seguenti modi:

- Discussione di gruppo nelle ore libere tra una lezione e un'altra;
- Utilizzo di un server discord provvisto di canali vocali e testuali utili nella comunicazione remota e nell'organizzazione delle informazioni;
- Utilizzo di un gruppo whatsapp per comunicazioni rapide e urgenti.

Ogni membro deve essere ben a conoscenza delle proprie competenze, in modo tale da eseguire il compito più adatto al fine di evitare un rallentamento del lavoro.

2.3. Interfacce organizzative

Per interfacce organizzative si intende le relazioni con altre entità. In particolare per quanto riguarda il docente esso si interfaccerà con il team leader che avrà l'ulteriore compito di effettuare le consegne finali. Rispetto ai potenziali utenti sarà compito di tutti i membri gestire un possibile dialogo per coinvolgerli nell'utilizzo dell'applicazione o per avere un punto di vista esterno.

2.4. Responsabilità di progetto

Potenzialmente ogni membro del gruppo può contribuire in egual modo nello sviluppo del progetto. Si è deciso tuttavia di adottare indicativamente la seguente suddivisione dei compiti, in modo tale da favorire la collaborazione e migliorare le interazioni tra i componenti.

Occupazione	Descrizione	Componenti
Sviluppo applicazione	Scrittura del codice sorgente.	Chinellato Marco Cogotti Giulia Passarella Diego Croitor Tudor

Occupazione	Descrizione	Componenti
Revisione codice	Controllo della qualità del codice.	Chinellato Marco Cogotti Giulia Passarella Diego Croitor Tudor
Testing e feedback	Testing delle funzionalità dell'applicazione, invio di feedback e segnalazioni di problemi.	Cogotti Giulia Chinellato Marco Passarella Diego Cazzin Marta Zhang Jinpeng Tudor Croitor
Stesura documentazione	Stesura dei vari documenti da consegnare descritti nella sezione 1.2 del corrente documento.	Cogotti Giulia Passarella Diego Chinellato Marco Zhang Jinpeng
Revisione documentazione	Controllo dei deliverable da consegnare.	Cogotti Giulia Tudor Croitor
Graphic UI Design	Prototipazione dell'interfaccia grafica e dell'esperienza utente.	Cazzin Marta Zhang Jinpeng

3. Processi gestionali

3.1. Obiettivi e priorità

Per concludere con successo il progetto, ovvero il completamento dell'applicazione secondo le premesse prestabilite, sono rilevanti gli aspetti che riguardano:

- *rispetto delle scadenze*: arrivare alla consegna dopo aver completato i singoli passi;
- *coinvolgimento utenti*: garantire che l'applicazione abbia uno scopo concreto nella vita reale;
- *cooperazione*: evitare che si creino meccanismi disfunzionali tra i membri del team;
- *rispetto delle funzionalità descritte*: implementare poche funzionalità correttamente.

3.2. Assunzioni, Dipendenze, Vincoli

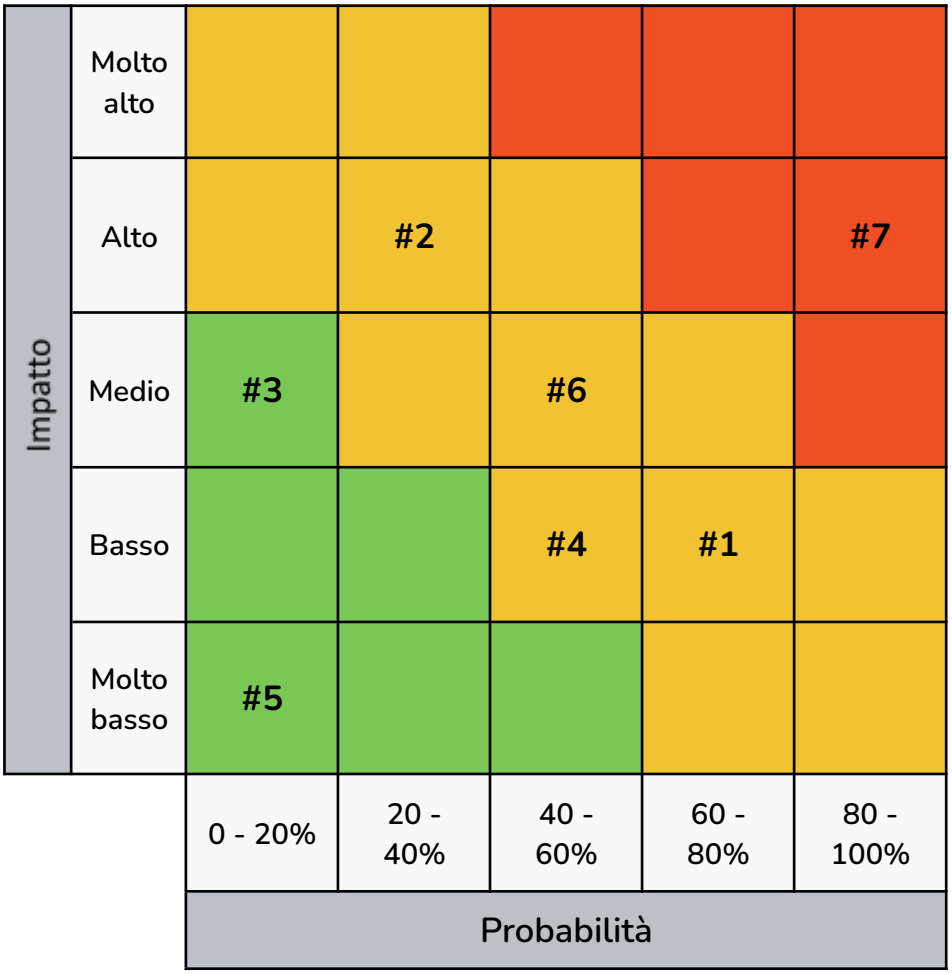
Per portare a termine del progetto si assume che tutti i membri del team siano responsabili e autonomi nel proprio lavoro per poter eseguire correttamente i propri compiti senza intralciare il lavoro complessivo. Inoltre si assume che ognuno abbia le capacità di lavorare in gruppo senza creare situazioni disfunzionali.

Riguardo le conoscenze tecniche, si assume che ogni membro sappia utilizzare gli strumenti, elencati al punto 4.1, necessari a svolgere i compiti assegnati al punto 2.4. In caso non fosse vero, ogni studente dovrà informarsi ed esercitarsi nella comprensione e utilizzo di tali software applicativi.

Altre assunzioni riguardano le ore di lavoro. Considerando che la consegna finale andrà effettuata a gennaio, si presuppone che nel periodo natalizio il lavoro eseguito sarà minore, sia per il tempo speso con la propria famiglia, sia per il tempo necessario allo studio di altri corsi. Di conseguenza sarà responsabilità di ognuno anticipare il lavoro per evitare un sovraccarico nel periodo indicato.

3.3. Gestione dei rischi

La tabella seguente visualizza la probabilità e i rischi relativi allo svolgimento del progetto. I rischi vengono identificati tramite una scala di colori che varia dal verde (basso impatto e bassa probabilità) al rosso (alto impatto e alta probabilità). Al di sotto vengono definiti i rischi con possibili misure per risolvere il problema. Ovviamente uno sforzo maggiore verrà fatto per prevenirli, le misure saranno adottate solo in casi estremi.



La seguente tabella illustra i rischi con la relativa probabilità e impatto, inoltre viene fornita una possibile soluzione ai problemi nel caso in cui essi si dovessero verificare:

N°	Rischio	Probabilità	Impatto	Soluzione
#1	Malattia di uno o più componenti.	75%	Basso	Lavoro in *smart-working*. In caso di gravi condizioni, suddivisione del lavoro tra i restanti membri del team.
#2	Un membro non porta a termine i suoi incarichi.	40%	Alto	Richiamo ed eventuale esclusione dal gruppo con conseguente ridistribuzione del lavoro.
#3	Un componente abbandona spontaneamente il gruppo.	10%	Medio	Ripartizione del lavoro fra i restanti membri del team.
#4	Competenze richieste troppo elevate con una complessità	55%	Basso	Studio approfondito degli strumenti e

N°	Rischio	Probabilità	Impatto	Soluzione
	del progetto troppo elevata.			ridimensionamento delle funzionalità.
#5	Perdita dei dati.	8%	Molto basso	Ripristino da backup precedenti.
#6	Impegni personali di un membro per un certo periodo prestabilito.	60%	Medio	Gestione del lavoro in data precedente, in modo da alleggerire il carico durante il periodo di assenza.
#7	Concorrenza con altri corsi.	95%	Alto	Provare a distribuire il tempo, in caso di fallimento dare minore priorità ad altri corsi.

3.4. Meccanismi di monitoraggio e di controllo

Ogni documento seguirà un format specifico fornito dal docente, in particolare dovrà essere presente il logo dell'università Ca' Foscari nell'intestazione e titolo del corso con nome del docente nel piè di pagina. La prima pagina funge da copertina, la seconda contiene una tabella riguardante le informazioni relative al documento e la terza contiene l'indice.

I documenti e il codice stesso partono dalla versione 1.0, in seguito quando verranno apportate importanti modifiche essi passeranno alla versione 1.X, fino al raggiungimento della versione finale 2.0.

Per quanto riguarda il monitoraggio dei rischi elencati nella gestione dei rischi, essi si possono prevedere e controllare tramite una comunicazione attiva tra i membri del team.

3.5. Pianificazione dello staff

Tutti i componenti del team devono collaborare e comunicare tra di loro, esponendo problemi e/o idee relativi al progetto stesso.

Il progetto richiede competenze nello sviluppo di applicazioni android, al momento nessun componente del gruppo ha queste capacità. Per risolvere il problema sarà necessario seguire attentamente le lezioni presentate in aula e documentarsi personalmente tramite internet o altre fonti informative.

Inoltre è fortemente richiesta una solida base di programmazione e capacità di problem-solving. Qualora ci fosse qualche componente con lacune in questo, esso dovrà approfondire la conoscenza tramite risorse a sua disposizione evitando di rallentare lo svolgimento del progetto.

4. Processi tecnici

4.1. Metodi, strumenti e tecniche

Nello sviluppo del progetto, per quanto riguarda il lato puramente software, verranno utilizzati i seguenti strumenti:

- ***Android Studio*** come ambiente di sviluppo per Java
- ***Java/XML*** per il FrontEnd, ***Firebase & Javascript*** per il Back-end
- ***API e Framework***: Firebase per gestire autenticazione degli utenti, Google Books API per avere una lista di tutti i libri indicizzati da google.
- ***Git/GitHub*** come strumento di controllo di versione del codice e piattaforma di backup decentralizzato.
- ***Figma*** come strumento di prototipazione dell'interfaccia ed esperienza utente.

La documentazione relativa al progetto viene fornita utilizzando i seguenti strumenti, ognuno di essi permette la scrittura simultanea di più utenti rendendone più semplice la stesura:

- LaTeX
- OverLeaf
- GoogleDocs

Di seguito sono indicate le linee guida a cui ci si farà riferimento, con i relativi collegamenti alle pagine web:

- Documentazione Java
  (https://docs.oracle.com/javase/7/docs/api/overview-summary.html)
- Documentazione Android Studio
  (https://developer.android.com/docs)
- Linee guida material design per Android
  (https://material.io/)
- Documentazione Firebase:
  (https://firebase.google.com/docs)
- Documentazione Javascript:
  (https://developer.mozilla.org/en-US/docs/Web/JavaScript)

4.2. Documentazione del software

In linea teorica è prevista la stesura di un manuale d'utilizzo dell'applicazione da fornire agli utenti. Questo documento verrà creato soltanto nel caso in cui tutte le attività del progetto finissero nei tempi previsti dall'EF (*earliest finish time)*, rappresentato al punto 5.2.

4.3. Funzionalità di supporto al progetto

Per le funzionalità di supporto viene fatto riferimento allo standard *ISO/IEC 25010* che descrive la qualità del software sulla base di otto attributi: idoneità funzionale, efficienze/performance, compatibilità, usabilità, affidabilità, sicurezza, manutenibilità e portabilità.

- **idoneità funzionale**: il sistema supporta tutte le funzionalità indicate
- **efficienza/performance**: il sistema riesce ad effettuare tutte le azioni senza un impiego di tempo eccessivo
- **compatibilità**: il sistema scambia correttamente e in pochi secondi informazioni tra applicazione e API (durante l'operazione di inserimento libri) e tra applicazione e database (ogni qualvolta sia necessario prelevare o inserire dei dati, i.e. login)
- **usabilità**: l'applicazione può essere facilmente utilizzata da utenti inesperti senza commettere errori dopo 30 minuti di uso intenso
- **affidabilità**: il sistema riesce ad eseguire correttamente tutte le funzionalità in un periodo limite
- **sicurezza**: i dati sensibili degli utenti sono protetti tramite i meccanismi di sicurezza di Firebase (database utilizzato), inoltre essi sono trattati con le modalità descritte nell'informativa sulla privacy
- **manutenibilità**: l'applicazione può essere modificata dagli sviluppatori in qualsiasi momento, senza perdita di dati da parte degli utenti e rilasciando una nuova versione
- **portabilità**: tramite l'apk dell'applicazione, scaricabile dall'apposita pagina su GitHub, il software può essere installato in qualsiasi dispositivo Android che rispetti le specifiche descritte nel documento riguardante l'*analisi dei requisiti*.

5. Pianificazione del lavoro, delle risorse umane e del budget

5.1. WBS

La seguente sezione mostra il WBS, ovvero il *Work Breakdown Structure*, il quale è un diagramma che suddivide il progetto principale in diverse attività, suddivise a loro volta in task. Inizialmente è presente una descrizione testuale del WBS e in seguito il vero e proprio diagramma.

Per convenzione si è deciso che le lettere greche indicano le funzioni, le lettere maiuscole le attività e le lettere minuscole i task. I numeri che seguono le lettere aiutano ad individuare la gerarchia tra attività, sotto attività e task.

Le dipendenze tra le varie componenti vengono mostrate nella sezione successiva in modo da non sovraccaricare graficamente il WBS.

Funzioni:

Le funzioni qua descritte prevedono un affiancamento *costante* allo sviluppo del progetto.

- **α** ***Documentazione***, composta dagli elaborati descritti nei *deliverables di progetto*.
- **β** ***Gestione progetto***.
- **γ** ***Controllo qualità*** del codice e della documentazione.
- **δ** ***Studio delle tecnologie*** in modo da adeguarsi agli standard esistenti

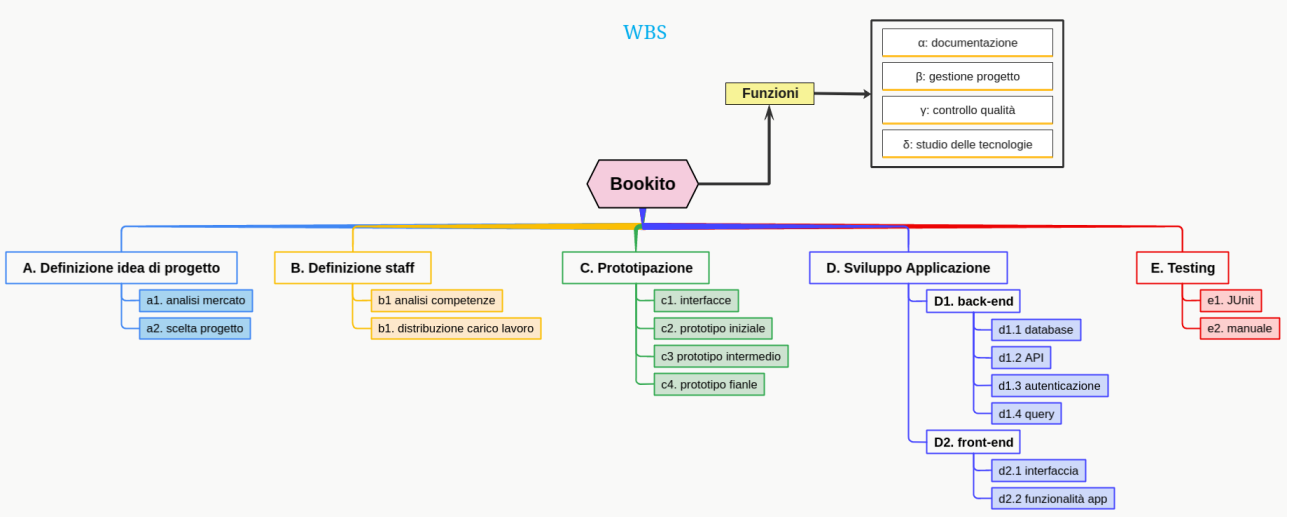
Attività:

- **A** **Definizione idea di progetto:**
  - *a1* *Analisi del mercato* riguardante le applicazioni già esistenti
  - *a2* *Scelta del progetto*, ovvero una chiara definizione riguardo alle funzionalità e lo scopo dell'applicazione scelta
- **B** **Definizione dello staff:**
  - *b1* *Analisi competenze dei membri* tramite un attento sguardo al curriculum presentato in fase di creazione team
  - *b2* *Distribuzione del carico di lavoro* come mostrato nelle *responsabilità di progetto*
- **C** **Prototipazione interfaccia grafica:**
  - *c1* *Definizione interfacce*
  - *c2* *Disegno del primo prototipo*
  - *c3* *Analisi e miglioramento del prototipo*
  - *c4* *Implementazione del prototipo finale*
- **D** **Sviluppo applicazione:**
  - **D1** **Configurazione backend:**

- *d1.1* *Configurazione database*
- *d1.2* *Configurazione API*
- *d1.3* *Configurazione autorizzazione e autenticazione utenti*
- *d1.4* *Generazione delle query* per operazioni sui dati dell'applicazione

- **D2** **Configurazione frontend:**
  - *d2.1* Implementazione dell'interfaccia
  - *d2.2* Sviluppo delle funzionalità dell'applicazione

- **E** **Testing:**
  - *e1* *Testing automatici con JUnit*
  - *e2* *Testing manuali* installando l'applicazione in un dispositivo Android.



5.2. Dipendenze

La tabella mostra i dati relativi all'WBS descritto nel punto precedente, aggiungendo le informazioni riguardanti le attività e i task. In particolare vengono mostrate le dipendenze, la durata espressa in giorni e i dati del diagramma di PERT (ES, EF, LS, LF). In particolare le sigle hanno i seguenti significati:

- **ES** (*earliest start time*): il minimo giorno di inizio dell'attività, a partire dal minimo tempo necessario per le attività che precedono.
- **EF** (*earliest finish time*): dato ES e la durata dell'attività, esso rappresenta il minimo giorno in cui l'attività può terminare.
- **LF** (*latest finish time*): il massimo giorno in cui quel task deve finire senza che si crei ritardo per i task che dipendono da lui.
- **LS** (*latest start time*): dato LF e la durata del task, esso rappresenta il massimo giorno in cui quel task deve iniziare senza provocare ritardo per i job che dipendono da lui.

ID	Dipendenze	Durata	ES	EF	LS	LF
A	-	3gg	19/09/2022	22/09/2022	23/09/2022	26/09/2022
a1	-	1gg	19/09/2022	20/09/2022	23/09/2022	24/09/2022
a2	*a1*	2gg	20/09/2022	22/09/2022	24/09/2022	26/09/2022
B	*A*	2gg	26/09/2022	28/09/2022	01/10/2022	03/10/2022
b1	*A*	1gg	28/09/2022	29/09/2022	01/10/2022	02/10/2022
b2	*b1*	1gg	29/09/2022	30/09/2022	02/10/2022	03/10/2022
C	*B*	15gg	03/10/2022	18/10/2022	08/12/2022	23/11/2022
c1	*B*	3gg	03/10/2022	06/10/2022	08/12/2022	11/12/2022
c2	*c1*	4gg	06/10/2022	10/10/2022	11/12/2022	15/11/2022
c3	*c2*	3gg	10/10/2022	13/10/2022	15/11/2022	18/11/2022
c4	*c3*	5gg	13/10/2022	18/10/2022	18/11/2022	23/11/2022
D	*C*	47gg	18/10/2022	4/12/2022	23/11/2022	15/01/2023
D1	*C*	14gg	18/10/2022	29/10/2022	23/11/2022	06/12/2022
d1.1	*C*	1gg	18/10/2022	19/10/2022	20/11/2022	21/11/2022
d1.2	*d1.1*	2gg	19/10/2022	20/10/2022	21/11/2022	23/11/2022
d1.3	*d1.2*	2gg	20/10/2022	22/10/2022	23/11/2022	25/11/2022
d1.4	*d1.3*	10gg	01/11/2022	11/11/2022	25/11/2022	05/12/2022
D2	*C, D1*	47gg	18/10/2022	05/12/2022	23/11/2022	15/01/2023
d2.1	*C*	7gg	18/10/2022	25/10/2022	23/11/2022	30/11/2022
d2.2	*D1*	40gg	29/10/2022	05/12/2022	06/12/2022	15/01/2023
E	*d2.1*	40gg	25/10/2022	05/12/2022	06/12/2022	15/01/2023
e1	*d2.1*	40gg	25/10/2022	05/12/2022	06/12/2022	15/01/2023
e2	*d2.1*	40gg	25/10/2022	05/12/2022	06/12/2022	15/01/2023

5.3. Risorse necessarie

Lo svolgimento del progetto richiede un certo quantitativo di risorse, in particolare risorse temporali. Ogni membro del team dovrà prendersi la responsabilità di garantire il suo contributo.

Al momento della stesura del presente documento, il team svolge almeno due incontri settimanali in presenza della durata di 5 ore ciascuno, per fare il punto sulla situazione e collaborare. In misura non quantificabile c'è una frequente comunicazione tramite discord e whatsapp per condividere idee e pareri sullo svolgimento del progetto.

Quando inizierà la scrittura del codice si presuppone che le ore dedicate al progetto aumentino.

5.4. Allocazione del budget e delle risorse

Tutte le API utilizzate prevedono un piano gratuito, quindi non sono stati spesi soldi da nessun componente del gruppo per il loro utilizzo.

Secondo varie fonti esaminate, lo stipendio di un Junior Software Engineer corrisponde a circa €13-14 l'ora, per semplicità questo valore verrà arrotondato a €13. Mediamente vengono svolti 2 incontri settimanali tra i componenti del gruppo in presenza, di circa 10 ore in tutto, per un totale di €780 a settimana. La consegna del progetto verrà effettuata in 13 settimane, dunque in totale la forza lavoro avrà un costo complessivo di €10.140.

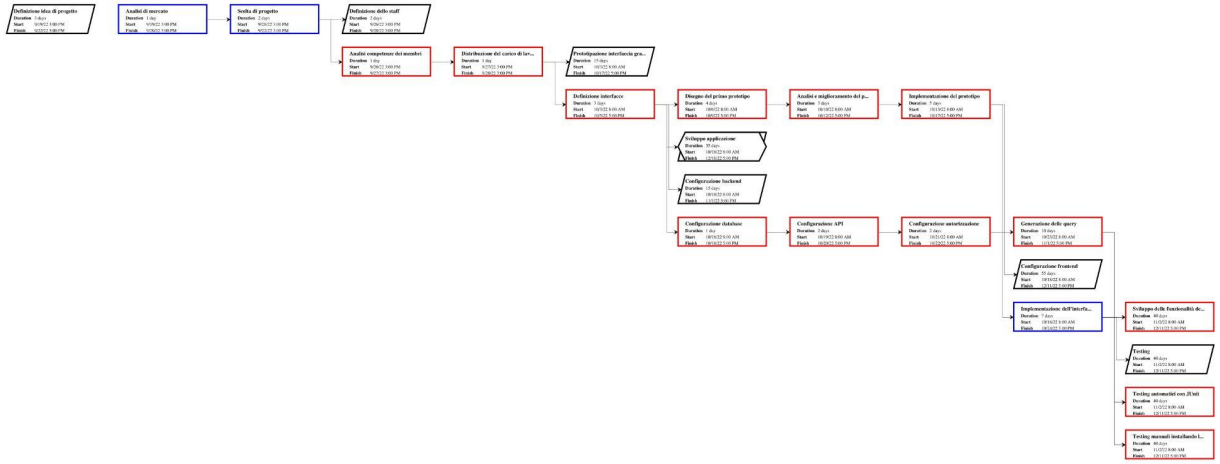
Qui di seguito viene riportata una tabella rappresentante una stima approssimativa del costo totale di progetto comprendente il costo di implementazione delle varie funzionalità e dei servizi offerti.

Gruppo	Tipologia	Prezzo	Totale parziale
Tempo di lavoro	Ore lavorative	€780/settimana	€10.140
Documenti	Piano di Progetto	€350	€1480
	Documento di analisi e Specifica	€370	
	Piano di Testing	€390	
	Documento di Progettazione	€370	
Funzionalità	Chat fra utenti	€500	€4.700
	Creazione database	€1000	
	Autenticazione utente	€700	
	Funzionalità esclusive	€2500	
		Totale (IVA ESCLUSA)	€16.320

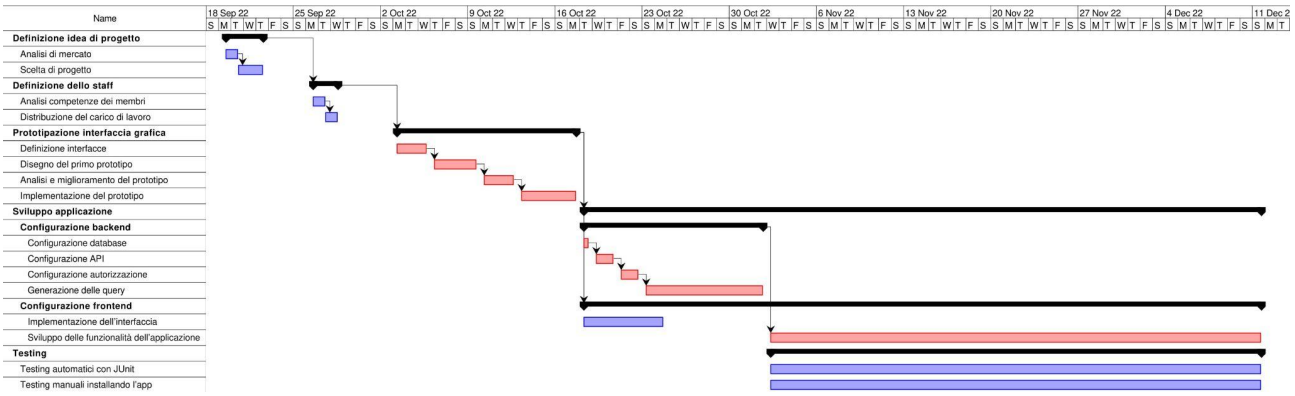
5.5. Pianificazione

In questa sezione verrà mostrato il diagramma di PERT e il diagramma di Gantt strettamente correlati alle attività e task descritti nell' WBS con le relative informazioni specificate nella tabella delle dipendenze.

Inizialmente viene mostrato il grafo delle attività, anche detto **diagramma di PERT**. Esso esprime le dipendenze e il cammino critico, cioè la sequenza di mansioni che porta ad arrivare alla consegna del progetto senza tollerare ritardi.



Infine viene mostrato il cosiddetto **diagramma di Gantt** che mostra i task del WBS esprimendo la temporizzazione. In questo modo si nota quali compiti possono essere sovrapposti e quali devono attendere la fine di un task precedente.



6. Riferimenti e materiali

I materiali e i testi di riferimento presi in considerazione per la stesura del presente documento sono indicati di seguito:

- Slides fornite dal professore durante lo svolgimento del corso.
- Progetti degli anni passati forniti sulla piattaforma moodle.
- Fonti online per stimare il calcolo del budget:
    - https://it.indeed.com/career/programmatore-java-junior/salaries
    - https://www.jobbydoo.it/stipendio/programmatore-informatico
- Software con funzionalità gratuite utilizzati per la progettazione dei diagrammi:
    - https://www.projectlibre.com/
    - https://www.edrawsoft.com/it/edraw-max/