Università
Ca'Foscari
Venezia

Corso:

Ingegneria del Software [CT0090]
2020/2021

Team:

Junipero

Membri:

Melania Gottardo	874240
Mario Coci	876422
Monan Nasir	870374
Simone Checco	869408

PIANO DI PROGETTO

VERSIONE 1.1

INDICE

1. INTRODUZIONE

1.1. OVERVIEW DEL PROGETTO

Servendosi di un proprio dispositivo Android è richiesto lo svolgimento delle fasi di analisi, progettazione, sviluppo e testing.

Utilizzando il terminale Android come unità computazionale, opportunamente programmato con l'uso di librerie specifiche, implementeremo un'applicazione di supporto al famoso gioco tradizionale "ruba bandiera".

> *Il gioco "ruba bandiera" si forma di due squadre nemiche che si sfidano. Ogni squadra avrà un fortino, dove sarà custodita la propria bandiera. I componenti di ciascuna squadra dovranno catturare la bandiera degli avversari e proteggere la propria bandiera, affinché non venga presa. Vince la squadra che prende per prima la bandiera avversaria.*

L'app è stata pensata per essere appunto un supporto tecnologico ad un celebre gioco per bambini, ma vuole inoltre motivare i fruitori dell'app stessa, che essi siano ragazzi o adulti, a svolgere attività fisica in uno spazio aperto, necessariamente molto ampio. L'app rompe le barriere date dal fatto di non dover avere la strumentazione fisica (es bandiere, fortini), e rende più comodo il gioco grazie all'uso del solo smartphone: per esempio, quando un giocatore corre, può tranquillamente metterlo in tasca. Quest'app è un ottimo mezzo per potersi interrompere, staccare dai propri impegni e permette a chiunque di divertirsi e mettersi in moto. È stato pensato per chi ormai non gioca da parecchio tempo a ruba bandiera e gioverebbe nel riscoprire un tradizionale gioco alla portata di tutti. L'obiettivo è anche quello di riportare alla luce il gioco, modernizzandolo e renderlo più accessibile a tutti, e dare una motivazione in più a chi vorrebbe riprendere a fare esercizio fisico, ma non è sufficientemente invogliato.

Si parlerà più approfonditamente dell'applicazione nella sezione 3.1.

Al seguente link è presente la prototipazione della grafica dell'applicazione: https://xd.adobe.com/view/7ecbf62a-1706-4c0e-b293-af58545a48b7-2b98/?fullscreen&hints=off

1.2. DELIVERABLES DEL PROGETTO

Le scadenze per questo progetto sono state programmate nel seguente modo:

- Piano di progetto (18/10/2020);
- Documento di analisi e specifica (02/11/2020);
- Piano di testing (19/11/2020);
- Documento di progettazione (14/12/2020);
- Codice e messa in linea del sistema (15/01/2021).

1.3. EVOLUZIONE DEL PROGETTO

In quanto il codice sarà modulare e scalabile, si potranno utilizzare le stesse componenti, (testate e funzionanti), per creare applicazioni che le sfruttino. Ad esempio:

- **Geolocalizzazione per tracciare le posizioni dei giocatori**: è la feature principale dell'intero progetto, senza di essa non sarebbe stata possibile avere l'idea della ruba bandiera estesa al mondo Android. Permette di visionare e individuare, (con una precisione dai 2 ai 5 metri) ogni utente partecipante ad una partita;

- **Socket**: Essi sono fondamentali per connettere all'host tutti i fruitori della partita: attraverso i socket si possono inviare i dati degli utenti al fine di interpretare le strategie delle 2 squadre. A tal proposito si possono utilizzare questi dati per implementare un'intelligenza artificiale in grado di predire le mosse dei giocatori in modo da fornire una guida riguardo le varie tecniche di gioco. Un'intelligenza artificiale di questo tipo è molto efficace perché studia i comportamenti delle persone mentre stanno giocando e possono infine migliorare l'esperienza di gioco ai nuovi fruitori dell'app;

- **Altri progetti ludici**: la geolocalizzazione può essere sfruttata anche per altre applicazioni che prendono spunto da giochi tradizionali: per esempio, si pensi al gioco del nascondino, dove i giocatori che si nascondono devono evitare di farsi vedere dal giocatore che detiene la tana. Si potrebbe pensare di realizzare un'applicazione simile a quella che il gruppo sta sviluppando, dove i giocatori nascosti possono vedere il "cacciatore" attraverso l'app, aumentando la difficoltà per chi difende la tana.

1.4. MATERIALE DI RIFERIMENTO

Durante l'interezza del progetto il team si servirà dei materiali didattici presenti sulla piattaforma Moodle del corso.

Per implementare al meglio il progetto il team ha deciso di utilizzare il sito developer.android.com, che è una guida per il design visivo, dinamico e interattivo di Android.

Per la gestione del database, il team fa affidamento alla guida di Firebase.

Per la prototipazione, il team fa affidamento alla guida di Adobe XD.

1.5. DEFINIZIONI ED ABBREVIAZIONI

Qui vengono riportate le varie abbreviazioni e le definizioni si alcuni termini utilizzati all'interno dei documenti legati a questo progetto.

- **Android**: Android è un sistema operativo per dispositivi mobili sviluppato da Google Inc. e basato sul kernel Linux; non è però da considerarsi propriamente né un sistema unix-like né una distribuzione GNU/Linux, bensì una distribuzione embedded Linux, dato che la quasi totalità delle utilità GNU è sostituita da software in Java.

  Lo sviluppo di Android prosegue attraverso l'Android Open Source Project, il quale è software libero con l'esclusione di diversi firmware non-liberi inclusi per i produttori di dispositivi e delle cosiddette "Google Apps" come ad esempio Google Play. È distribuito sotto i termini della licenza libera Apache 2.0 riservandosi di non-includere software coperto da licenze copyleft.

  Fonte: https://it.wikipedia.org/wiki/Android;

- **Android Studio**: Android Studio è un ambiente di sviluppo integrato (IDE) per lo sviluppo per la piattaforma Android.

  Fonte: https://it.wikipedia.org/wiki/Android_Studio;

- **App**: applicazioni, quelle per dispositivi mobili, vengono identificate semplicemente come App.

  Fonte: https://it.wikipedia.org/wiki/Applicazione_(informatica);

- **Branch**: un branch è un ramo. Un ramo è un insieme univoco di modifiche. Ogni repository può avere più branch;

- **Entry**: tupla del database;

- **Firebase**: è una piattaforma per la creazione di applicazioni per dispositivi mobili e web sviluppata da Google. Originariamente era una compagnia indipendente fondata nel 2011. Nel 2014 Google ha acquisito la piattaforma ed è ora la sua offerta di punta per lo sviluppo di app.

  Fonte: https://it.wikipedia.org/wiki/Firebase;

- **GitHub**: GitHub è un servizio di hosting di repository Git, ma offre molte più funzionalità. GitHub fornisce un'interfaccia grafica basata sul web, il controllo dell'accesso e diverse funzionalità di collaborazione, come strumenti di base per la gestione delle attività per ogni progetto;

- **Master branch**: il master branch è il branch principale del progetto. Viene creato per la prima volta quando si invia il primo file nella propria repository;

- **Pull request**: le richieste pull consentono di comunicare agli altri le modifiche che sono state inviate a un ramo di una repository su GitHub. Una volta aperta una richiesta pull, si possono discutere e rivedere le potenziali modifiche con i collaboratori e aggiungere commit di follow-up prima che le modifiche vengano unite al master branch;

- **ES (earliest start time)**: indica il giorno minimo 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 indica il minimo giorno in cui l'attività può terminare;
- **LF(latest finish time)**: il giorno massimo in cui l'attività deve finire senza creare ritardo alle attività che dipendono da lui;
- **LS(latest start time)**: dato LF e la durata dell'attività indica il giorno massimo in cui l'attività deve iniziare senza provocare ritardi alle attività che dipendono da lui.

---

2. ORGANIZZAZIONE DEL PROGETTO

2.1. MODELLO DEL PROCESSO

Il modello del processo che si è deciso di adottare è di tipo **Agile**. Si vuole porre il focus sull'usabilità dell'applicazione e sulla ricezione da parte dei fruitori. Il team vuole essere più flessibile possibile per adattarsi ai cambiamenti in corso d'opera, senza rimanere ancorato ad un progetto che potrebbe bloccarsi e non portare più ai risultati desiderati. Inoltre, si vuole dare una libertà controllata ai membri del team, lasciandoli lavorare in autonomia, ma a delle task prestabilite dall'intero team.

Per ottimizzare il lavoro, il team si organizza la settimana con meeting per identificare i compiti da svolgere quella settimana, la suddivisione dei compiti e fare una ricapitolazione di ciò che è stato fatto in quella settimana. Verranno schedulate le attività considerando un margine per gli imprevisti e un ordine da rispettare nella sequenza di operazioni da svolgere con eventuali sovrapposizioni, anticipazioni o posticipazioni. Gli incontri settimanali avverranno secondo la seguente programmazione.

Lunedì	Martedì	Giovedì	Venerdì
Mattino	Pomeriggio	Giornata intera	Pomeriggio
Incontro per la ricapitolazione del lavoro svolto nella settimana passata, l'organizzazione della settimana appena cominciata.	Lavoro al progetto in presenza per confronto diretto con il team per dubbi e perplessità sul lavoro da svolgere.	Lavoro al progetto in presenza per confronto diretto con il team per dubbi e perplessità sul lavoro da svolgere.	Termine per la consegna del lavoro settimanale, punto della situazione del progetto con eventuale rischedulazione delle attività o delle tempistiche.

Per supportare l'organizzazione delle attività settimanali si utilizzerà una scrum board semplificata identificata con tre colonne: "to do", "doing", "done".

TO DO	DOING	DONE
➔ attività 1 ➔ attività 5	➔ attività 3 ➔ attività 4	➔ attività 2

Nella colonna "to do" verranno riportate le attività ancora da svolgere programmate per quella settimana; nella colonna "doing" verranno riportate le attività già cominciate che devono essere ancora terminate (non è detto che le attività in questione saranno completate in quella determinata settimana); nella colonna "done" verranno inserite le attività completate fino a quel momento dall'inizio della pianificazione del progetto.

Le attività saranno identificate e assegnate direttamente ad un componente del team con le attività presenti nella sezione 5.1.

Le fasi principali del progetto sono:

- Analisi e definizione del progetto;
- Prototipazione dell'interfaccia grafica;
- Progettazione del sistema e del software:
    - Componenti di geolocalizzazione;
    - Componenti di connessione;
    - Comunicazione delle componenti di geolocalizzazione e di connessione;
- Creazione di una demo dell'app;
- Implementazione e test delle singole componenti;
- Integrazione e test del sistema;
- Sviluppo della versione completa dell'app;
- Sviluppo interfaccia grafica e migliorie;
- Installazione e mantenimento.

Ogni fase avrà una durata con un certo margine di errore e verrà relazionata alle altre per la schedulazione con le varie sovrapposizioni, anticipazioni e posticipazioni come spiegato nella sezione 5.1.

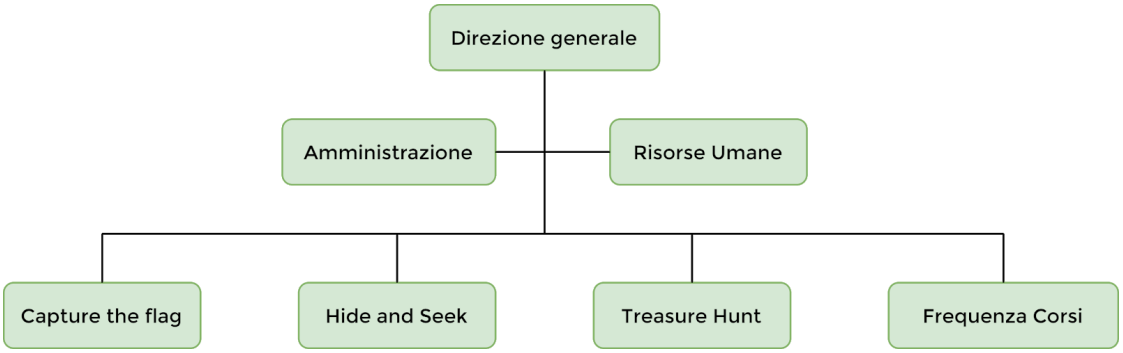
2.2. STRUTTURA ORGANIZZATIVA

Qui di seguito verranno esplicitate le scelte per quanto riguarda la struttura organizzativa e la tipologia di team.

- **Struttura organizzativa**: la tipologia di organizzazione che si è deciso di adottare per questo progetto è l'**Organizzazione per Progetto**.

  Si è deciso di optare per questa struttura organizzativa per perseguire i seguenti obiettivi:

  - Definire con chiarezza i ruoli e le responsabilità delle persone assegnate al progetto, nonché definire l'interfacciamento con persone esterne in relazione al progetto;
  - Assegnare formalmente le responsabilità del project manager e degli altri ruoli di coordinamento;
  - Approvare gli obiettivi ed i piani da parte di tutta della struttura impattata dal progetto, in particolar modo dai vertici.



- **Tipologia di team**: per quanto riguarda la tipologia di team, si è optato per un **Controllo Decentralizzato**. Qui i leaders coordinano chi sta sotto di loro nella gerarchia, si riferiscono a chi sta sopra di loro nella gerarchia per essere coordinato e comunicano tra di loro se stanno nello stesso livello della gerarchia. Si ottiene una comunicazione duplice:

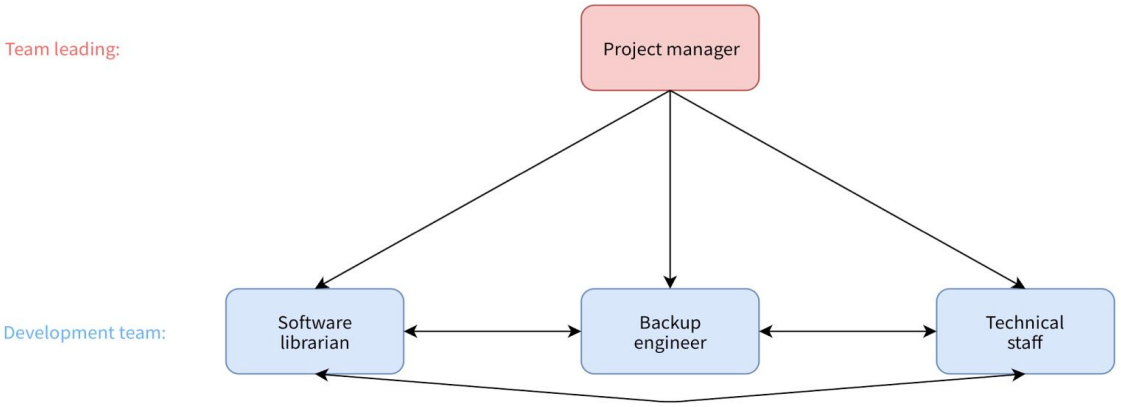
  - **Verticale**: tra persone che appartengono a livelli diversi e che, quindi, vengono coordinate o coordinano;
  - **Orizzontale**: tra persone che appartengono ad uno stesso livello.

  Nel caso di questo progetto ci saranno solo due livelli:

  - **Team leading**: è il primo livello, composto esclusivamente da una persona che coordina le persone che appartengono al livello due;
  - **Development team**: è il secondo livello che fa testo al primo livello; chi fa parte di questo livello ha una comunicazione orizzontale con gli altri membri che appartengono al livello.

  Ogni persona di ogni livello avrà poi un ruolo specifico a cui attenersi con dei compiti specifici da svolgere (vedi sezione 2.4).

Qui viene riportato il grafico con le specifiche dei livelli e dei soggetti che fanno parte del team di sviluppo del progetto.



2.3. INTERFACCE ORGANIZZATIVE

Relazioni con altre entità: durante la realizzazione del progetto, per il team assegnato allo svolgimento del progetto sarà necessario interfacciarsi con persone esterne al team. In particolare, le entità con cui il team avrà necessità di interfacciarsi sono:

- **Monitoraggio**: per quanto riguarda il monitoraggio ed i feedback sul progetto, il team si interfaccerà con il Professor A. Cortesi periodicamente. In particolare, il membro del team che si occuperà di questa entità sarà il project manager.
- **Consultazione**: per quanto riguarda la consultazione durante le fasi di sviluppo del progetto, il team si interfaccerà con il Professor A. Spanò e con il tutor del corso G. Zausa. In particolare, i membri del team che si occuperanno principalmente di questa entità saranno il software librarian e il technical staff.
- **Stakeholder**: per quanto riguarda l'approcciarsi con chi potenzialmente potrebbe essere influenzato da una decisione, attività o risultati del progetto (vedi sezione 1.1), se ne occuperà il membro designato come backup engineer.

2.4. RESPONSABILITÀ DEL PROGETTO

Qui sono elencati i vari ruoli assegnati ai membri del team con relativi compiti e responsabilità. Questi sono compiti specifici che caratterizzano maggiormente ciascun membro del team, ma ognuno sarà in grado di sostituire chiunque altro membro nel caso egli sia impossibilitato ad adempiere il proprio ruolo, per sviluppare l'applicativo e per stilare i documenti relativi al progetto.

- **Project manager**:
    - Membro: Melania Gottardo;
    - Responsabilità: pianifica, coordina e supervisiona le attività del team per il progetto corrente durante le varie settimane, accertandosi che ogni membro del gruppo svolga i compiti a lui/lei assegnati; in più, si assicura che i deliverables vengano consegnati entro le date previste e siano completi. Come ruolo aggiuntivo, si occupa di gestire le dinamiche del team qualora vi siano screzi o si incorra in uno o più rischi (sezione 3.3);
- **Technical staff**:
    - Membro: Simone Checco;
    - Responsabilità: è a capo dell'analisi e dello sviluppo dell'applicativo assieme al software librarian, il quale gli fornirà le direttive per procedere nello sviluppo dell'applicativo. Inoltre si occuperà di monitorare lo sviluppo del codice di tutti gli altri membri del team;
- **Backup engineer**:
    - Membro: Mario Coci;
    - Responsabilità: supporta il project manager nella gestione del team e lo sostituisce qualora sia impossibilitato ad adempiere al suo ruolo. È responsabile della validazione del software, ovvero sarà colui che darà il via libera per l'approvazione (interna la team) del codice scritto, infatti sarà lui che validerà le pull request (vedi sezione 4.3) verso il ramo principale sul quale si sta lavorando e sul ramo master di GitHub;
- **Software librarian**:
    - Membro: Monan Nasir;
    - Responsabilità: espone le problematiche software da risolvere all'intero team e si occupa di ideare delle possibili soluzioni a tali problemi lavorando a fianco al Technical staff durante lo sviluppo dell'applicativo, inoltre mantiene la documentazione aggiornata in caso di cambiamenti riguardanti lo sviluppo lato codice del progetto.

---

3. DESCRIZIONE DEI PROCESSI GESTIONALI

3.1. OBIETTIVI E PRIORITÀ

- **Obiettivo:** l'obiettivo principale consiste nel creare una applicazione peer2peer utilizzabile dalla maggior parte dei dispositivi Android. L'applicazione deve essere in grado di scambiare (inviare e ricevere) diversi tipi di dati con altri dispositivi Android, in modo da poter raggiungere un obiettivo non realizzabile con un unico telefono.

  L'applicazione da noi ideata consiste in una rivisitazione digitale del gioco tradizionale "ruba bandiera", dove due squadre si sfidano per cercare di catturare la bandiera della squadra nemica. È necessario quindi, all'inizio di ogni partita, stabilire chi saranno i cacciatori, che dovranno difendere la propria bandiera e cercare contemporaneamente di catturare quella nemica, e chi sarà, invece, la base che avrà lo scopo di tenere la bandiera al sicuro dai cacciatori nemici. La partita finisce nel momento in cui una delle due squadre cattura la bandiera della squadra nemica.

  L'applicazione, quindi, avrà bisogno di utilizzare alcuni sensori all'interno del telefono: in base al ruolo che si andrà a ricoprire, infatti, si utilizzeranno sensori diversi. Chi sarà cacciatore avrà bisogno di utilizzare il localizzatore gps (per tener conto della distanza tra lui e la bandiera da catturare), l'accelerometro e magnetometro (per capire invece la direzione da prendere, altrimenti si potrebbe pensare di trovare la direzione verso la bandiera "avversaria" calcolandola attraverso la posizione della bandiera stessa e la posizione di se stessi); mentre, chi interpreterà la base avrà sempre il segnalatore gps attivo in modo da dover comunicare in ogni momento la sua posizione agli altri.

  Il primo obiettivo quindi sarà quello di realizzare una demo, dove le funzioni del gioco saranno limitate (possibilità di creare una sola partita, controllo del calcolo della distanza). Una volta completata la demo (numerosi test superati con successo) verrà sviluppata la versione finale, implementando la possibilità di creare più partite contemporaneamente, oltre ad una serie di altre cose, come la grafica per l'applicazione e la gestione di un database dinamico e tutti i sensori che saranno d'aiuto per il funzionamento dell'applicazione. Verrà anche implementato un sistema base di messaggistica istantanea.

- **Priorità:** a livello tecnico, il team dovrà necessariamente gestire le eccezioni ed evitare malfunzionamenti tecnici che porterebbero ad un fallimento dell'applicazione e della prova stessa. Per questo motivo è necessario tenere conto delle seguenti priorità:
  - i codici identificativi dei giocatori devono essere univoci;
  - le stanze create dall'host devono essere univoche e devono avere un codice che verrà comunicata a chi deve entrare in stanza;
  - sono ammessi al massimo 10 giocatori a partita;
  - dovrà essere presente solamente una bandiera per squadra;

- non si può in nessun caso spegnere la geolocalizzazione o la connessione ad internet;
- se sono ad una distanza tale (dalla persona con la bandiera della squadra opposta) da poter vincere la partita (5 metri ca.) il risultato della partita deve essere immediato e comunicato a tutti i giocatori, e la partita viene conclusa;
- non ci sono, invece, restrizioni per quanto riguarda la grandezza massima del campo da gioco.

3.2. ASSUNZIONI, DIPENDENZE, VINCOLI

- **Assunzioni:** bisogna assumere che ogni membro del team sia responsabile, diligente e che operi nel bene comune del team che sia autonomo nella gestione del suo tempo per il completamento degli obiettivi settimanali, e che sia disposto a mettere a disposizione le proprie abilità per il completamento del progetto. Assumiamo inoltre che i mezzi tecnici, i materiali e la strumentazione per portare a termine la varie parti del progetto (e successivamente il progetto intero) siano a disposizione sin dall'inizio dei lavori in modo da non causare ritardi per mancanza di materiale.
- **Dipendenze:** per la riuscita del progetto è necessaria la conoscenza del linguaggio Java, della struttura dei database NoSQL, dell'ambiente di sviluppo Android Studio e le conoscenze base per poter utilizzare GitHub. Nel caso di Android Studio, esso verrà appreso attraverso le lezioni frontali e la documentazione sulla piattaforma Moodle. Un discorso diverso vale per Java, dove è necessaria una conoscenza pregressa per linguaggio ad oggetti, e NoSQL, dove bisognerà effettuare una preparazione come autodidatti.
- **Vincoli:** bisogna assolutamente evitare di consegnare oltre il giorno di scadenza di consegna dei lavori, in modo da evitare ritardi a cascata su consegne successive; e non consegnare documenti non controllati, superficiali, e non intestati in maniera corretta. Cercare (possibilmente tutti gli elementi del team o almeno una persona) di essere sempre a lezione in modo da rimanere sempre aggiornati sulle ultime comunicazioni e per poter chiarire eventuali dubbi riguardo lo sviluppo.

3.3. GESTIONE DEI RISCHI

- **Strategia preventiva per la gestione dei rischi - Riduzione del rischio:** La tecnica adottata dal team è denominata Riduzione del rischio, ed è incentrata su azioni che hanno la possibilità di ridurre in maniera sostanziosa le probabilità che un rischio si verifichi e di conseguenza anche il suo impatto. La motivazione che ha portato il team a questa scelta è, che non avendo a disposizione tempi molto lunghi per un ricalcolo di tutte le tempistiche, si preferisce prevenire il rischio, visto che, nonostante monitorare e controllare richiede una spesa elevata in risorse, in termini puramente tempistici, ci permette di lavorare in maniera più rapida e fluida rispetto al ricalcolo delle tempistiche e delle mansioni lavorative.

- **Identificazione:** per evitare ritardi o addirittura la cancellazione del progetto è necessario identificare i rischi, ecco un elenco di eventuali rischi da dover evitare. Il piano è comunque da rivedere in modo periodico.
    - **Mancanza di responsabilità:** può capitare che uno o più membri del team sottovalutano, rimandino o addirittura abbandonino la propria mansione. Ciò potrebbe causare eventuali ritardi, bloccare il processo di realizzazione o sovraccaricare di lavoro i restanti membri del team.
    - **Incapacità collaborativa:** possono verificarsi scontri o incomprensioni fra i membri del team per visioni di sviluppo differenti. Questo caso si può verificare solo nel caso di un sistema disorganizzato e immaturo. Per ovviare questo possibile problema, è necessario o un sistema di voto democratico o un leader che prenda decisioni in caso di incertezza.
    - **Disgregazione del team o dei suoi elementi:** se le controversie all'interno del team si accentuano in maniera troppo grave, si rischia la disgregazione del team e di non riuscire a portare più a termine il progetto. Per questo motivo, è importante chiarire sin da subito qualsiasi diverbio tra i membri in modo da lavorare in maniera più coesa possibile.
    - **Problemi di salute:** uno o più membri del team potrebbero essere inattivi a causa di malattie che potrebbero tenerli a riposo forzato per periodi più o meno lunghi; in questo periodo, non bisogna nemmeno sottovalutare il rischio di contagio che obbligherebbe ad un periodo di stop dei lavori da parte di tutto il team.
    - **Perdita dati:** se non vengono effettuati dei backup del lavoro svolto, si rischia di perdere tutto il lavoro e di dover ricominciare da capo o dall'ultimo salvataggio eseguito con tutte le conseguenze tempistiche del caso. Meglio adottare alcune contromisure di auto-salvataggio (meglio se su cloud), oppure condivisione del lavoro svolto su più dispositivi in modo da avere più copie.
    - **Malfunzionamenti macchine da lavoro:** la possibile rottura delle macchine dove lavoriamo potrebbe oltre a portare una parziale perdita di dati (come già detto), anche rallentamenti dovuti alla diminuzione delle capacità produttive. È necessario quindi cercare di curare in ogni dettaglio il dispositivo sul quale si lavora.
    - **Malfunzionamenti macchine di test:** essendo una applicazione da testare con più dispositivi l'eventuale rottura di uno di essi potrebbe compromettere determinati test procedendo quindi alla cieca o subendo gravi ritardi. Sarà necessario quindi, avere cura dei propri dispositivi di test o, al massimo, in caso di rottura, averne qualcuno di scorta in modo da sostituire quello non disponibile.
    - **Scarsa conoscenza tecnica:** se non si ha esperienza con i linguaggi, gli ambienti di sviluppo e le strumentazioni da utilizzare, ci saranno sicuramente difficoltà nella stesura del codice e rallentamenti nelle sequenze di lavoro. Questa conoscenza permette anche di scoprire e risolvere l'eventuale presenza di bug.

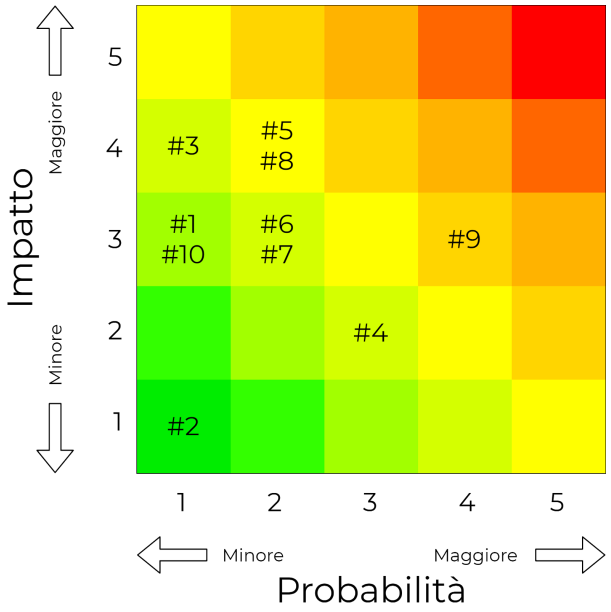
- **Previsione sbagliata:** è possibile sottovalutare o certe fasi di progetto o sviluppare parti non necessarie al fine del completamento della versione finale. È necessario quindi prevedere con cura le tempistiche e la direzione di sviluppo da prendere per evitare lavoro inutile e ritardi.
- **Chiusura spazi di lavoro:** viste le circostanze del periodo attuale, potrebbe essere possibile non lavorare più in presenza, questo potrebbe creare disagi e incomprensioni durante i lavori di gruppo, pertanto è necessario adottare misure di comunicazione real-time per gestire in modo efficiente il lavoro.

Qui si seguito, sono riportate la tabella dei rischi sopra elencati e la corrispondente classificazione degli stessi.

Tabella dei rischi

ID	Rischio	Probabilità	Impatto	Azione
1	Mancanza di responsabilità	1	3	Richiamo ed eventuale monitoraggio del lavoro
2	Incapacità collaborativa	1	1	Richiamo e voto democratico, eventuale espulsione del membro non collaborativo
3	Disgregazione del team o dei suoi elementi	1	4	Patto di termine lavori o rifondazione del team con elementi rimasti
4	Problemi di salute	3	2	Ripartizione temporanea del lavoro
5	Perdita dati	2	4	Backup frequenti e condivisione salvataggi su più dispositivi
6	Malfunzionamento macchine da lavoro	2	3	Manutenzione o, se possibile, eventuale sostituzione temporanea
7	Malfunzionamento macchine di test	2	3	Sostituzione delle macchine danneggiate
8	Scarsa conoscenza tecnica	2	4	Ripartizione del lavoro o studio approfondito
9	Previsione sbagliata	4	3	Ricalcolo del lavoro e rapidità decisionale
10	Chiusura spazi di lavoro	3	1	Sostituzione incontri in presenza con incontri online

Classificazione



3.4. MECCANISMI DI MONITORAGGIO E CONTROLLO

I nostri canali di comunicazione principali saranno WhatsApp e Slack, mentre per la condivisione della documentazione e del codice verranno utilizzati GitHub, per la condivisione del codice sorgente, e Google Drive, per la condivisione dei documenti. Inoltre sono previsti più incontri settimanali in modo da confrontarsi sul lavoro svolto, quello da svolgere ed eventuali problemi da risolvere. La divisione dei compiti e i relativi controlli fanno riferimento al punto 2.4 del presente documento.

3.5. PIANIFICAZIONE DELLO STAFF

Per poter sviluppare adeguatamente questo progetto, sono necessarie conoscenze pregresse al corso, come Java (linguaggio ad oggetti), conoscere la struttura dei database NoSQL e saper utilizzare in maniera efficace l'ambiente di sviluppo Android (Android Studio); inoltre, è necessario avere delle conoscenze base di GitHub per lavorare in un proprio branch che poi verrà implementato nel master branch in comune a tutti gli utenti del team. Alcune di queste conoscenze verranno a formarsi durante l'anno, come per esempio la conoscenza dell'ambiente di sviluppo Android. Una volta terminata una parte dello sviluppo sarà compito del backup engineer analizzare la realizzazione, se essa sarà valutata con successo, allora si passerà allo sviluppo del punto successivo, altrimenti si tornerà a lavorare sui punti che non soddisfano i requisiti.

4. DESCRIZIONE DEI PROCESSI TECNICI

4.1. METODI, STRUMENTI E TECNICHE

In questo paragrafo verranno esposti i processi tecnici e la strumentazione usata nei vari processi, è necessario definirli meticolosamente per facilitare il team ad adottare le medesime misure fisse all'interno del progetto.

- **Ambiente di sviluppo:** il progetto verrà sviluppato nell'IDE Android Studio sviluppato da JetBrains, inoltre essendo un ambiente grafico molto avanzato permetterà alle membri del team di testare l'applicazione su un emulatore presente nel software stesso;

- **Linguaggi**: lo sviluppo avverrà nel linguaggio Java per quanto riguarda la programmazione della business logic, mentre per lo sviluppo delle interfacce si lavorerà con il linguaggio XML oppure utilizzando lo strumento di interfaccia grafica di Android Studio (quest'ultimo genererà automaticamente codice XML), per il salvataggio dei dati dell'utente in locale verrà usato il formato JSON per mantenere la persistenza dei dati stessi;

- **Strumentazione di lavoro:** per lo sviluppo di questo progetto sarà necessario adoperare delle macchine dotate di sistema operativo Windows 10 (con i seguenti requisiti tecnici minimi: 4 GB di RAM, WIndows 7, 2 GB di spazio su disco) con relativa connessione dati;

- **Strumentazione di test:** per le operazioni di testing verranno utilizzati Smartphones dotati di sistema operativo Android 6.0+ con le seguenti caratteristiche tecniche minime: 1 GB di RAM, 100 MB di spazio di archiviazione, GPS integrato, connessione dati abilitata. Inoltre per sfruttare le API di android per avere una localizzazione il più precisa possibile è obbligatorio aver installati i Google Play Services 11.0+.
  NOTA: È necessario l'utilizzo di almeno 4 smartphone per effettuare le prove, sia per testare la versione demo dell'applicazione, che per la versione finale del progetto;

- **Standards di sviluppo**: è necessario strutturare e modellare il codice affinché sia modulare e scalabile. Questo modo di realizzare il codice permette una visualizzazione più pulita e più facile da leggere: viene così semplificato il lavoro all'interno del team per maneggiare sezioni di codice scritte da altri membri del team o comunque per sfruttare il meccanismo di incapsulamento. Inoltre, questo permetterà, in futuro, di testare le singole componenti ed effettuare manutenzioni, se necessario, su di essi, inoltre è obbligatorio l'utilizzo dei commenti per spiegare ad altri membri del team, i quali si occuperanno di controllare la qualità ed il corretto il funzionamento delle componenti appena sviluppate, la qualità del codice stesso. Inoltre, è necessario questo requisito per permettere ai membri di questo progetto di facilitare l'evoluzione del progetto (descritta nella sezione 1.3). In aggiunta, per omogeneizzare il lavoro e l'innesto di termini per le variabili/campi/metodi con le keyword di base, l'applicazione sarà creata utilizzando la lingua inglese in tutto ciò che la riguarda.

4.2. DOCUMENTAZIONE DEL SOFTWARE

La documentazione del software sarà realizzata dal team durante lo sviluppo del progetto, sarà disponibile entro le scadenze indicate nella sezione 1.2.
Inoltre se sarà necessario, verranno apportate modifiche al documento in caso di rischedulazione dei compiti secondo la programmazione citata nella sezione 2.1.

Intanto, verrà prodotta una versione demo dell'app per provare effettivamente la concretezza del progetto, da quel punto si migliorerà e correggerà il prototipo sino ad arrivare alla versione finale dell'applicativo.

4.3. FUNZIONALITÀ DI SUPPORTO AL PROGETTO

Essendo un progetto di gruppo, è necessario adottare anche misure di comunicazione, le quali agiranno come strumenti di supporto per lo stesso progetto. Qui sotto verranno elencate le principali funzionalità di supporto adottate del team:

- **Pianificazione della qualità:** è necessario controllare lo sviluppo del codice man mano che verranno implementate features nell'applicazione, affidando tale pratica agli altri membri del team, o almeno al soggetto identificato come backup engineer. Tale controllo avverrà rispettando determinati pattern di programmazione e la modularità del codice stesso, inoltre è necessario effettuare test molto frequentemente implementando dei test di integrazione nella repository del progetto.
- **Pianificazione della gestione delle configurazioni:** il lavoro del singolo avverrà su un nuovo branch dove ci si impone di sviluppare/correggere una feature, se tale implementazione verrà accettata dal backup engineer, si potrà effettuare una pull request sul branch master.
- **Funzionalità di supporto:** il team utilizzerà **GitHub** per mantenere il codice sorgente con i relativi backup, dove, sia il team che il responsabile di tutorato potranno controllare l'avanzamento del progetto, è molto utile inoltre per tornare a versioni precedenti (possibilmente funzionanti) in caso di malfunzionamenti importanti sul software che si sta sviluppando attualmente.

  Parallelamente, si utilizzerà la suite di Google, in particolare **Google documents** e **Google Drive** per mantenere e redigere la documentazione relativa al progetto.

  Per sviluppare il prototipo di interfaccia grafica verrà utilizzato il software **Adobe Xd**, il quale servirà solamente a mostrare le funzionalità dell'app ai suoi fruitori.

  Il mantenimento dinamico dei dati avverrà sulla piattaforma di Google denominata **Firebase**, la quale offre un servizio di database NoSQL con aggiornamenti in real-time, in quanto, ogni partita (composta da due squadre) avrà una sua entry nel database, nella quale verranno salvate dinamicamente i dati relativi alla geolocalizzazione delle persone denominate come “bandiere”.

5. PIANIFICAZIONE DEL LAVORO, DELLE RISORSE UMANE E DEL BUDGET

5.1. WBS (WORK BREAKDOWN STRUCTURE)

Si esplicita qui la struttura del progetto nelle diverse funzioni, attività e milestones, ovvero la work breakdown structure del progetto.

Il team, come accordato nella sezione 2.1, ha optato per una metodologia Agile, quindi ha proseguito con l'individuazione di una serie di funzioni per avere un costante riscontro, monitoraggio e sviluppo settimanale del progetto nella sua interezza, nonché per l'assegnamento e controllo delle diverse attività svolte e da svolgere.

Queste sono le **funzioni** identificate dal team.

- α **Project Management**
  Funzione che sarà presente durante lo svolgimento di tutto il progetto, adottando la metodologia Agile, sarà necessario al team, organizzare il lavoro per ogni settimana mostrando i risultati prodotti o problematiche da risolvere durante l'ultimo giorno lavorativo della settimana (nel nostro caso il venerdì) e se necessario apportare modifiche alla documentazione e riorganizzare il lavoro tra le componenti del team.

- β **Apprendimento tecniche e tecnologie**
  Apprendimento dei vari linguaggi e tecniche per la realizzazione dell'applicazione. In particolare linguaggi Java/XML, e l'utilizzo delle librerie di Android, inoltre per la gestione dei dati riguardanti le coordinate sarà necessario saper interfacciarsi con il database presente sulla piattaforma Firebase.

- γ **Documentazione**
  La redazione dei documenti avverrà prevalentemente durante lo sviluppo del codice sorgente (ad eccezione del piano di progetto che sarà redatto prima dello sviluppo del codice sorgente), sia per essere più organizzati nel lavoro, sia per favorire una conoscenza comune a tutti i membri del team, e, se necessario, anche per apportare modifiche ai vari documenti che potrebbero riportare dati non aggiornati, man mano che il progetto evolve.

  - γ.1. **Piano di progetto**
    Sviluppo e stesura dei requisiti per poter scrivere il piano di progetto.

  - γ.2. **Documento di analisi e specifica**
    Sviluppo e spiegazione dei vincoli e degli obiettivi riguardanti l'applicazione.

  - γ.3. **Piano di testing**
    Ideazione e realizzazione delle date per i relativi test per verificare l'operatività dell'applicazione.

  - γ.4. **Documento di progettazione**
    Definizione dell'architettura del sistema, costruzione dei modelli UML, progettazione UI.

δ **Controllo qualità**
Controllo della qualità dell'applicazione, deve essere in linea con gli obiettivi descritti nel piano di progetto,

ε **Testing**
La fase di testing è la fase dove verranno effettuati dei test per verificare se la implementazioni sviluppate potranno considerarsi adeguate al funzionamento finale dell'applicazione oppure avranno bisogno di altro lavoro perché incomplete o con malfunzionamenti.

ε.1. **Unit**
Test delle singole activities terminate.

ε.2. **Integration**
Test di tutte le activities dopo aver integrato l'activity appena terminata.

Il progetto è stato diviso in più specifiche attività, alcune delle quali presentano una più attenta analisi e, quindi, una suddivisione in task. Si è deciso di non realizzare una analisi altrettanto (o addirittura più) specifica per il resto delle attività, in quanto il team vedrà di settimana in settimana di analizzare più approfonditamente le attività per assegnare nello specifico le task a ciascun membro del team.

Queste sono le **attività** identificate dal team.

A **Ideazione del progetto**
Comprende le attività di brainstorming tra i membri del team per ideare un possibile progetto da sviluppare per il corso di Ingegneria del software.

B **Analisi di fattibilità**
In questa fase si analizzano obiettivi, requisiti e vincoli per la realizzazione del progetto ideato nel precedente punto.

C **Definizione del progetto**
Si vuole mostrare al team quale progetto verrà sviluppato e a quale scopo.

D **Prototipazione dell'interfaccia grafica**
Verrà sviluppato un mockup semplice per avere una interfaccia grafica che mostri le funzionalità dell'applicazione.

E **Definizione dello staff**
Vengono definiti i ruoli e le relative mansioni per ogni membro del gruppo in modo da spartire il lavoro in modo equo e definire una struttura lavorativa.

F **Analisi dei requisiti**
In questa fase si analizzeranno i requisiti tecnici decisi dal committente e dal team stesso.

G **Analisi e gestione dei rischi**
Vengono presi in considerazione le eventuali problematiche che potranno manifestarsi durante lo sviluppo dell'applicativo e la loro gestione.

H **Pianificazione delle attività**
Discussione, definizione e pianificazione delle attività e funzioni e delle relative tempistiche.

I **Costruzione WBS, Pert e Gantt**
Definire l'ordine cronologico delle varie attività e la loro durata, tenendo conto dei vincoli temporali tra un'attività e l'altra.

J **Analisi delle problematiche e ideazione delle risoluzioni**
Verranno analizzate le possibile problematiche relative al progetto, i sistemi di prevenzione e le possibili soluzioni nel caso si dovesse riscontrare uno o più problemi.

K **Progettazione della demo**
Sviluppo di un prototipo funzionante di base dove verranno implementate le funzionalità principali del progetto.

- K.1 **Configurazione del database**
  Fase preliminare del progetto dove avverrà l'inizializzazione del database definendo le possibili tabelle in esso.
- K.2 **Costruzione UI e Activities di base**
  Sviluppo di una base solida, sia UI che codice sorgente dove il progetto dovrà essere scritto.
- K.3 **Progettazione meccanismo di tracking**
  Sviluppo del sistema di gestione delle coordinate tramite il GPS e la connessione dati.
- K.4 **Sviluppo delle classi**
  Ideazione e sviluppo delle entità che comporranno le business logic.
- K.5 **Collegamento UI alla business logic**
  Link delle varie activities con le rispettive schermate dell'applicazione per la visualizzazione degli output.
- K.6 **Sviluppo delle Activities**
  Utilizzo delle classi create per la risoluzione delle problematiche precedentemente analizzate.

L **Sviluppo progetto completo**
Sviluppo dell'applicazione finale ampliando alcune funzionalità di base.

- L.1 **Ampliamento e miglioramento classi e Activities**
  Ampliamento e ottimizzazione delle activities e delle classi sviluppate nella demo.
- L.2 **Progettazione Chat di gruppo**
  Implementazione della possibilità di mandare messaggi ad altri utenti che stanno partecipando alla partita.
- L.3 **Gestione molteplici partite simultanee**
  Implementazione nel database della possibilità di gestire più partite contemporaneamente ed in maniera dinamica.

M **Sviluppo UI definitiva**
Costruzione dell'interfaccia grafica definitiva sulla base del prototipo grafico realizzato su Adobe Xd.

N **Features extra**
Implementazione di effetti audio in base al contesto dell'attività attualmente in corso sull'applicativo.

- **Controllo e consegna del progetto definitivo**
  Ultimi test dell'applicazione e successiva consegna della documentazione completa e del codice sorgente.

Per una corretta scansione e cadenza del progetto, il team ha individuato tre milestones che lo caratterizzano. Queste milestones saranno dei punti cardine e fondamentali del progetto, in quanto rappresentano dei successi e dei pilastri del progetto stesso, senza i quali il progetto non raggiungerebbe la sua conclusione.

Queste sono le **milestones** identificate dal team.

- **Rilascio Demo**
  Verrà rilasciata la versione dimostrativa dell'applicazione per mostrare il suo effettivo effettivo funzionamento nelle procedure di base, quali, la localizzazione e la scrittura/lettura delle coordinate nel database, avvicinamento tra uno o più dispositivi, la creazione dinamica della partita (di almeno 4 giocatori) e l'interazione tra più dispositivi durante l'avvicinamento al dispositivo identificato come "bandiera".

- **Versione Completa**
  Rilascio della versione completa dell'applicazione con ampliamento di alcune funzionalità. Si amplieranno le funzionalità di base affinché le partite vengano gestite dinamicamente (più partite simultanee), inoltre verrà implementata una chat di gruppo per facilitare la comunicazione tra i membri della stessa squadra.

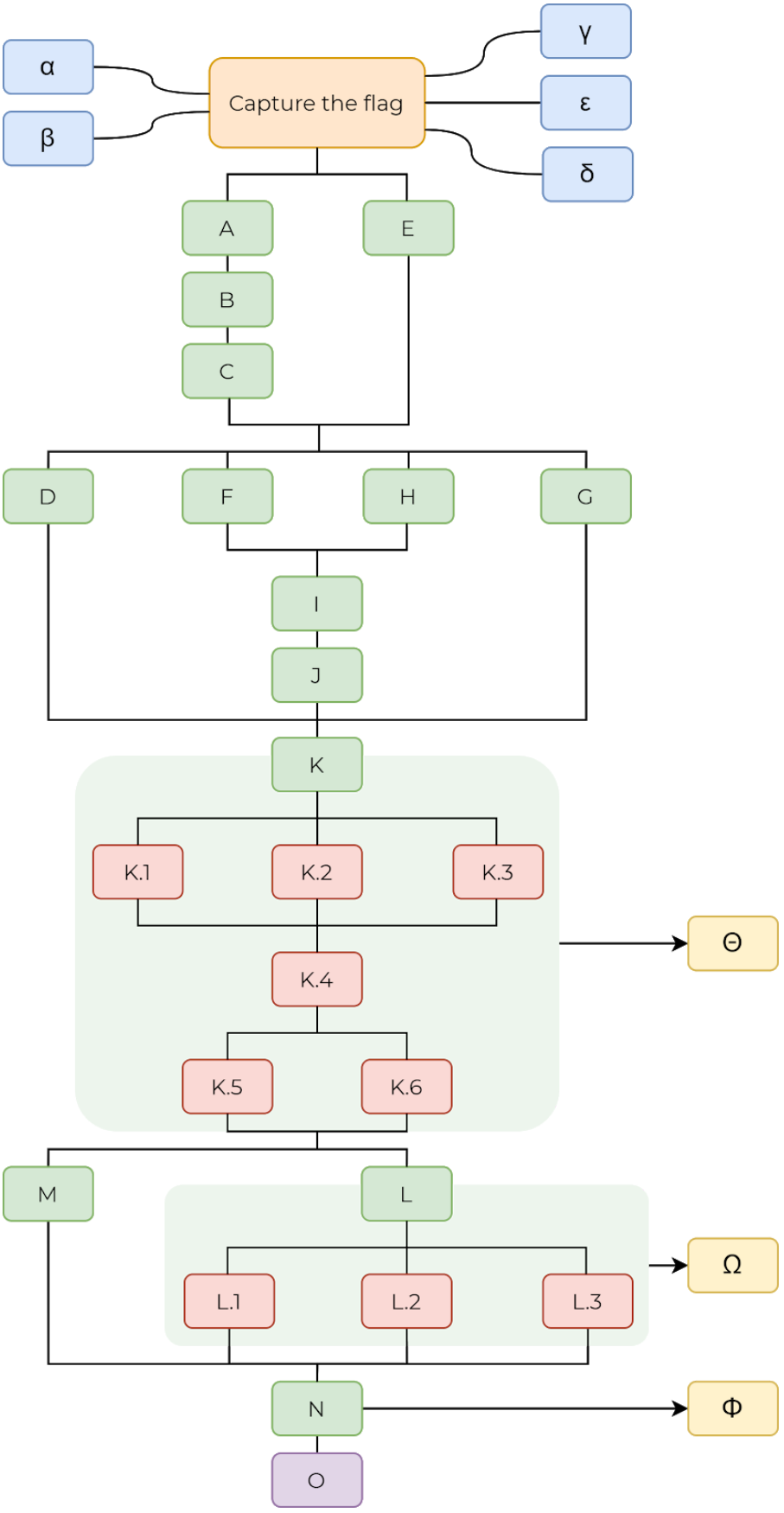
- **App finale**
  Rilascio della versione completa dell'applicazione con la grafica corretta e finale, non più basilare. L'applicazione funzionerà nella sua totalità e sarà pronta per l'utilizzo. Saranno anche sviluppate delle feature extra per rendere più accattivante ed interattiva l'applicazione.

Dopo aver analizzato la WBS nelle sue varie parti, viene riportata la sua struttura graficamente in uno schema. Qui sono inserite tutte le funzioni, le attività, i task e le milestones secondo un ordine logico, in più si identifica anche un sequenzialità o parallelizzazione delle stesse attività (o task).

WBS

Legenda

Funzione	Attività	Attività finale	Task	Milestone


α
β
Capture the flag
γ
ε
δ
A
E
B
C
D
F
H
G
I
J
K
K.1
K.2
K.3
Θ
K.4
K.5
K.6
M
L
Ω
L.1
L.2
L.3
N
Φ
O

5.2. DIPENDENZE

Per dipendenze si intendono quelle attività che sono strettamente legate al termine di una attività precedente. Questo perché alcune attività dipendenti hanno bisogno di alcune parti del progetto terminate, senza contare che la forza lavoro limitata può saturarsi facilmente, ed è quindi necessario distribuire il lavoro in maniera equa e dilatata nel tempo. Le attività che prevedono una dipendenza sono forzate ad essere sequenziali, mentre, per attività che non hanno obblighi l'una verso l'altra, è possibile utilizzare una parallelizzazione delle stesse attività per ottimizzare i tempi di lavoro e permettere ai membri del team di lavorare separatamente su più attività in contemporanea per completare diversi tasselli che poi andranno ad incastrarsi per formare un unico grande puzzle. La scelta della metodologia Agile (vedi sezione 2.1), è stata fatta anche per operare bene secondo questa proposta di lavoro in parallelo, così da non dipendere sempre da altre attività.

Qui di seguito è stilata una tabella delle dipendenze con le attività (o task) che verranno svolte dal team con relativa durata e data di inizio e di fine da cui è possibile ricavare il cammino critico delle attività che devono essere assolutamente assolte nel periodo indicato (vedi Diagramma di Pert).

Oltre alla data di inizio e di fine prestabilite, sono anche segnate, per ogni attività (o task), le date di inizio e di fine ultime a cui si può fare riferimento per non sforare con i tempi previsti per la durata totale del progetto.

È stato tenuto conto che il team lavorerà durante la settimana dal Lunedì fino al Venerdì di ogni settimana, tutti i giorni Sabato e Domenica sono stati esclusi. Esclusi anche i giorni dove cadranno le festività principali, come il Natale.

Per questioni di tempo, la festività dell'8 Dicembre (festa dell'Immacolata) verrà considerata come giorno lavorativo, pertanto, lo svolgimento delle attività proseguirà regolarmente in remoto per tale giornata.

Dalla tabella delle dipendenze, è stato ricavato il Diagramma di Pert che mostra graficamente il cammino del progetto con il relativo cammino critico. Per ogni attività (o task) sono state indicati l'ID, la durata, l'ES, l'EF, il LS ed il LF.

Per semplicità di lettura, il Diagramma di Pert mostra solo le attività e a lato sono riportati i sotto-diagrammi delle attività K ed L, per le quali erano state individuate delle task precise.

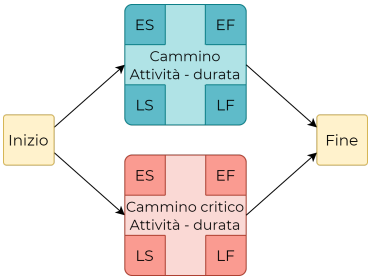
Notare che la funzioni e le milestones che sono state definite precedentemente in questa tabella e nel Diagramma di Pert non sono presenti, poiché le milestones corrispondono al completamento di relative attività, e la funzioni ricoprono un ruolo perpetuo all'interno dell'intero progetto, ripresentandosi e mantenendosi per la sua intera durata.

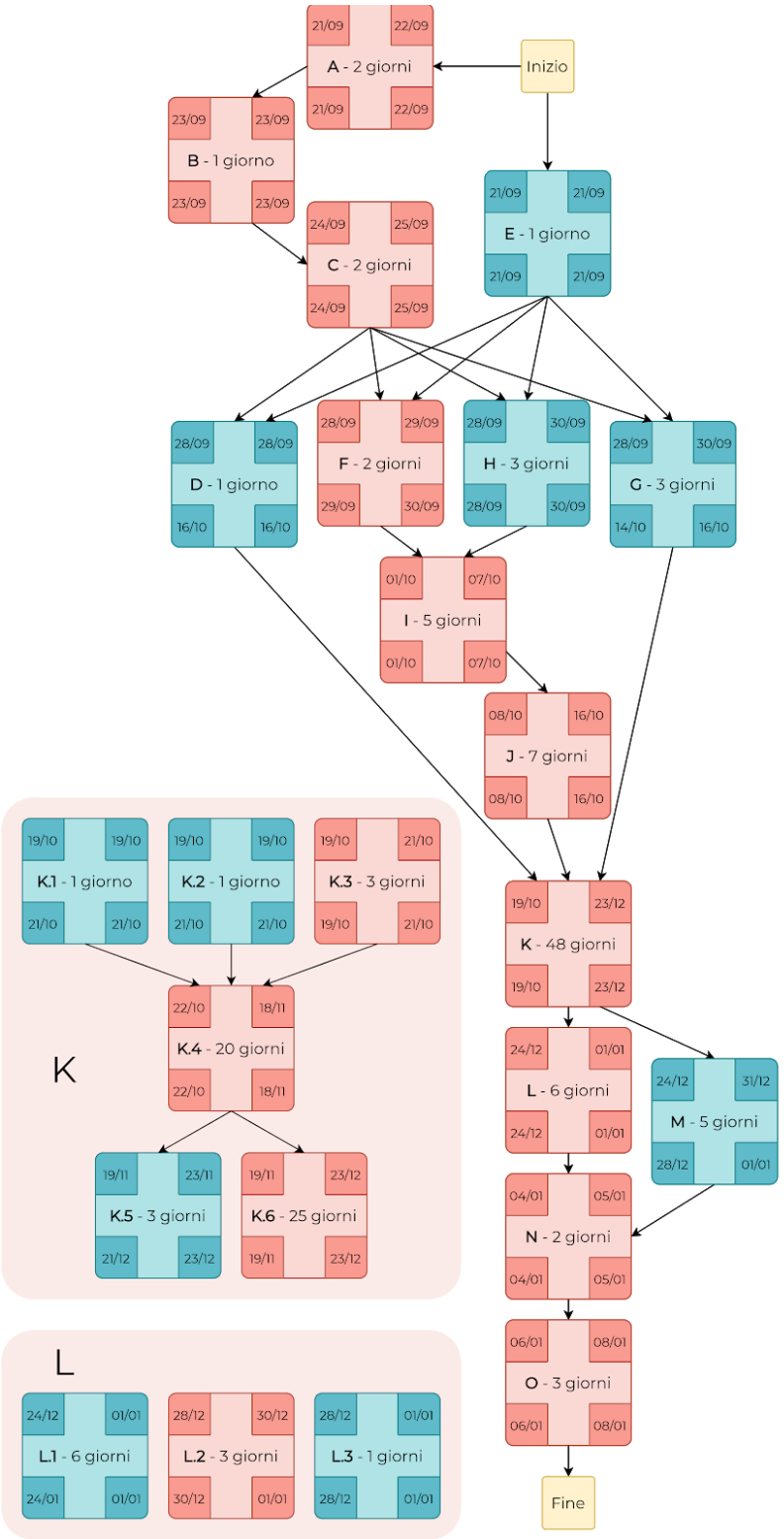
Tabella delle dipendenze

ID	Dipendenze	Durata	ES	EF	LS	LF
A	-	2 gg	21/09/20	22/09/20	21/09/20	22/09/20
B	A	1 gg	23/09/20	23/09/20	23/09/20	23/09/20
C	B	2 gg	24/09/20	25/09/20	24/09/20	25/09/20
D	C - E	1 gg	28/09/20	28/09/20	16/10/20	16/10/20
E	-	1 gg	21/09/20	21/09/20	25/09/20	25/09/20
F	C - E	2 gg	28/09/20	29/09/20	29/09/20	30/09/20
G	C - E	3 gg	28/09/20	30/09/20	14/10/20	16/10/20
H	C - E	3 gg	28/09/20	30/09/20	28/09/20	30/09/20
I	F - H	5 gg	01/10/20	07/10/20	01/10/20	07/10/20
J	I	7 gg	08/10/20	16/10/20	08/10/20	16/10/20
K	D - G - J	48 gg	19/10/20	23/12/20	19/10/20	23/12/20
K.1	D - G - J	1 gg	19/10/20	19/10/20	21/10/20	21/10/20
K.2	D - G - J	1 gg	19/10/20	19/10/20	21/10/20	21/10/20
K.3	D - G - J	3 gg	19/10/20	21/10/20	19/10/20	21/10/20
K.4	K.1 - K.2 - K.3	20 gg	22/10/20	18/11/20	22/10/20	18/11/20
K.5	K.4	3 gg	19/11/20	23/11/20	21/12/20	23/12/20
K.6	K.4	25 gg	19/11/20	23/12/20	19/11/20	23/12/20
L	K	6 gg	24/12/20	01/01/21	24/12/20	01/01/21
L.1	K	6 gg	24/12/20	01/01/21	24/12/20	01/01/21
L.2	K	3 gg	28/12/20	30/12/20	30/12/20	01/01/21
L.3	K	1 gg	28/12/20	28/12/20	01/01/21	01/01/21
M	K	5 gg	24/12/20	31/12/20	28/12/20	01/01/20
N	M - L	2 gg	04/01/21	05/01/21	04/01/21	05/01/21
O	N	3 gg	06/01/21	08/01/21	06/01/21	08/01/21

Diagramma di Pert

Legenda





5.3. RISORSE NECESSARIE

Si riconoscono molteplici categorie di risorse necessarie alla realizzazione di tale progetto:

- **Risorse umane:** identificato dai membri del team e le loro competenze tecniche (come lo sviluppo di software e la stesura della documentazione).
- **Risorse hardware:** composto da tutta la strumentazione hardware necessaria, quali, le macchine da lavoro ed i dispositivi Android di test.
- **Risorse software:** sarà necessario munirsi di software come Android Studio per lo sviluppo generale dell'applicazione, Google Drive e Documenti per la condivisione la stesura della documentazione del progetto stesso, Git per facilitare lo sviluppo del software ed una piattaforma di comunicazione istantanea tra i membri del gruppo (ad esempio WhatsApp/Telegram).
- **Risorse temporali:** composto dalle ore lavorative necessarie alla realizzazione del progetto, è necessaria una discreta gestione del tempo in quanto i membri del team non si dedicheranno solamente questo progetto, ma anche alla frequenza e allo studio di altri corsi.
- **Risorse documentative:** fa riferimento a tutta la documentazione di riferimento per sviluppare l'applicazione, quali la documentazione e le guide sulle API di Android, la documentazione sulle API per la gestione del database sulla piattaforma FIrebase, un piccolo manuale per usare Git e i requisiti di progetto.

5.4. ALLOCAZIONE DEL BUDGET E DELLE RISORSE

Non sono presenti spese economiche per lavorare su questo progetto, in quanto la strumentazione di lavoro e la strumentazione di test sono effettivamente i dispositivi già in possesso da ogni membro del team, inoltre i software che verranno utilizzati sono gratuiti (Android Studio, piattaforma Google Drive/Documenti, Github, piano base di Firebase).

Si potrebbe assumere di quantificare economicamente il lavoro di ogni singolo membro del team in Euro per ora di lavoro. A tal proposito, sapendo che i meeting di lunedì e venerdì saranno complessivamente di 2 ore, il martedì si lavorerà in presenza 3 ore e il giovedì si lavorerà sempre in presenza per 6 ore, arrivando ad un monte di 11 ore settimanali. Sapendo che dal 21 settembre al 28 dicembre trascorrono poco più di 13 settimane e definendo 13 Euro per ogni ora lavorativa del singolo membro (il team è composto da 4 persone), si può stimare che solamente per le ore lavorative di tutti i membri, il progetto avrà un costo base di 7.644.

Nota: vista la scelta della metodologia Agile vengono contate solamente le ore di lavoro in gruppo, mentre per le ore di lavoro individuali, la corretta gestione del tempo è affidata al membro stesso.

Gruppo	Settore	Prezzo	Totale parziale
Ore contate	Uomo-ora * 4	€572/settimana	€7.644
Documenti	Piano di progetto	€250	€2.150
	Documento di analisi e specifica	€800	
	Piano di testing	€600	
	Documento di progettazione	€500	
Features	Business logic	€1.600	€3.600
	Database dinamico	€400	
	Partite simultanee	€400	
	Chat di gruppo	€400	
	Extra	€300	
Grafica	Grafica applicazione	€500	€500
		Totale finale (IVA esclusa)	€13.394

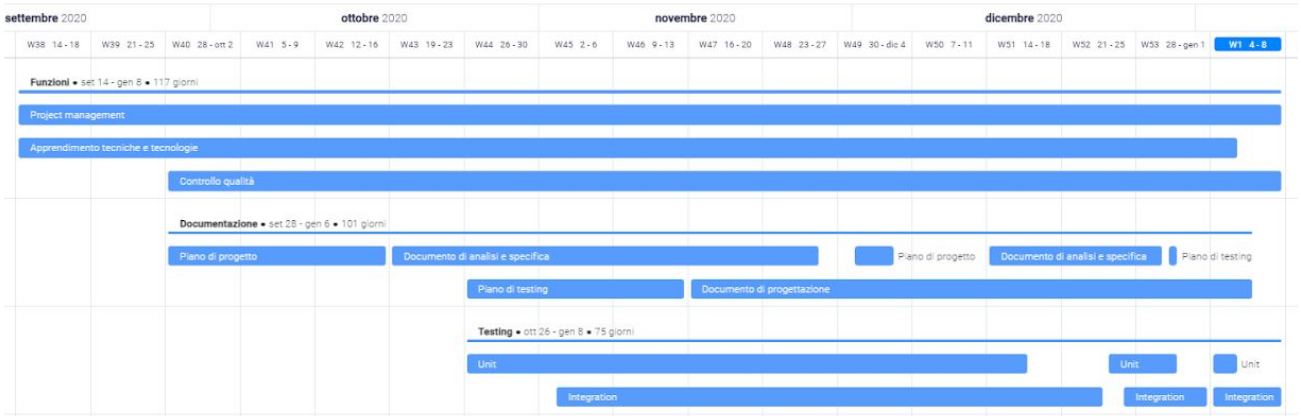
5.5. PIANIFICAZIONE

Durante la pianificazione del progetto vengono fissate delle deadline che dovranno essere rispettate e verranno considerate tali, quando verrà consegnata la documentazione relativa. Oltre le scadenze principali sono state considerate anche le Milestones che sono punti imprescindibili del progetto pur non facendo parte delle Deadlines, il loro compito, è di affermare la riuscita di una parte importante del piano di lavoro, dando la visione del compimento di una parte del progetto o facendo emergere delle problematiche da risolvere. Di seguito sono riportate le Deadlines riguardanti la consegna della documentazione:

1. Piano di Progetto (18/10/2020)
2. Documento di analisi e specifica (02/11/2020)
3. Piano di Testing (19/11/2020)
4. Milestone 1: Rilascio demo (30/12/2020)
5. Milestone 2: Versione completa (05/01/2020)
6. Documento di progettazione (14/12/2020)
7. Milestone 3: App finale (08/01/2021)
8. Realizzazione (15/01/2021).

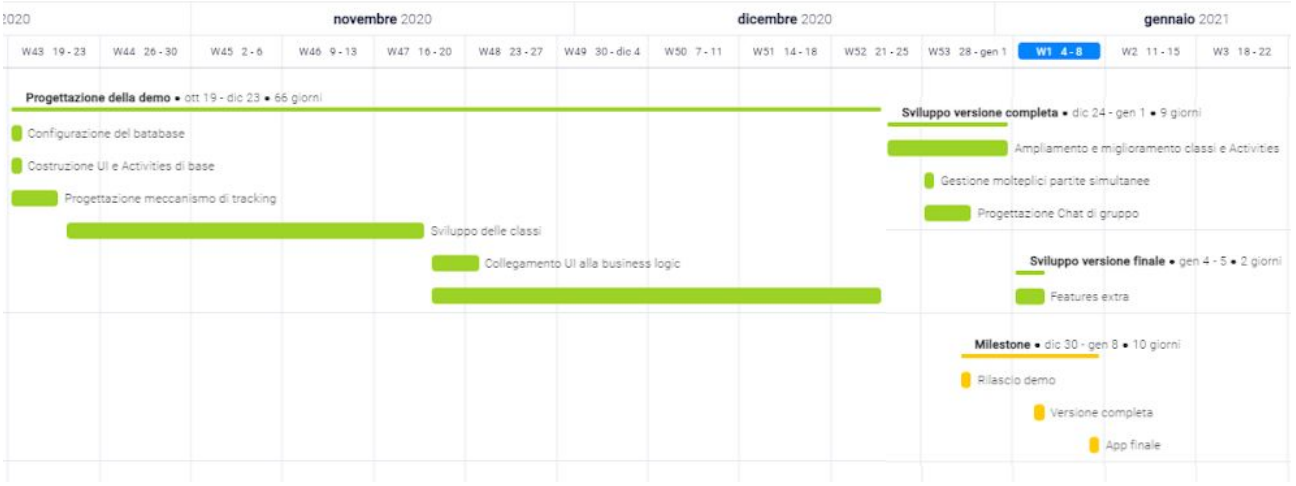
Diagramma di Gantt

Funzioni



Attività

Task e milestones



Distribuzione attività staff

Di seguito viene mostrata l'organizzazione dello staff in base alle attività da svolgere per singolo membro del team. Le attività fanno riferimento all'indice della WBS indicato nella sezione 5.1: in questo grafico vengono mostrate le attività in base alla settimana in cui ci si dedicherà a tale mansione (viene solamente indicato il primo lunedì di ogni settimana per ogni mese), per avere una visione più precisa delle attività vedere la tabella delle dipendenze alla sezione 5.2.

Questo grafico permette al team di schedulare al meglio tutte le attività necessarie al fine di realizzare il progetto.

6. RIFERIMENTI

Per creare questo documento sono stati utilizzati come riferimento:

- "piano di progetto" di alcuni gruppi degli anni passati;
- materiale messo a disposizione dal professore.