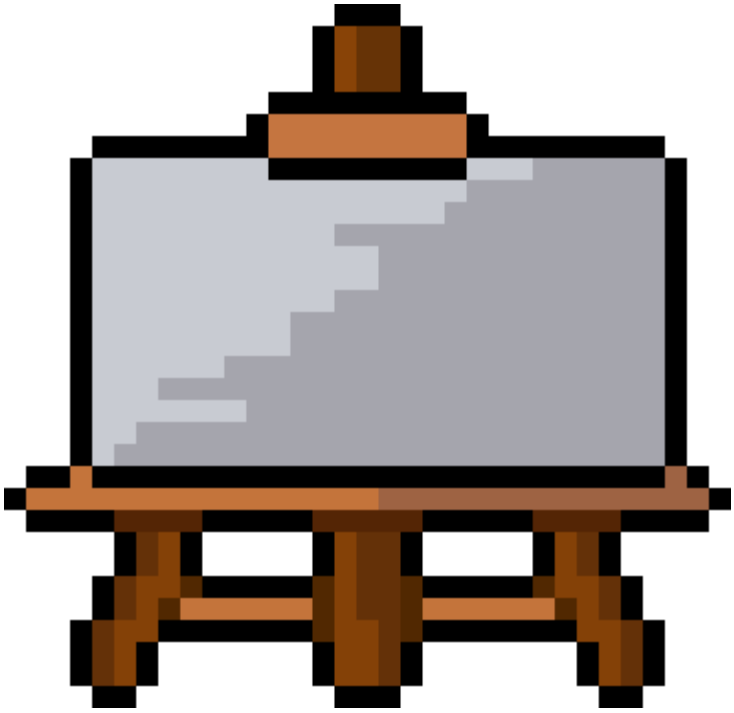
WIP
Work Is Progress
Flutter version documentation



Alfabeticamente, a cura di:
Colleluori Federico, Frisi Emanuele, Giusti Kevin
Versione documento: 1.0

Indice

Introduzione

"Niente è facile come sembra"
"Tutto richiede più tempo di quanto si pensi"
"Ogni soluzione genera nuovi problemi"
- Corollari della legge di Murphy, nonché leggi fondamentali dell'ingegneria del software

Il documento che segue è atto ad accompagnare il lettore nella comprensione di tutti gli elementi costitutivi del software "WIP: Work Is Progress", progettato e sviluppato nell'arco di tre mesi dalle matricole Colleluori Federico, Frisi Emanuele e Giusti Kevin, di cui riportiamo i contatti:

- Colleluori Federico, matr. 1093242, e-mail s1093242@studenti.univpm.it
- Frisi Emanuele, matr. 1092866, e-mail s1092866@studenti.univpm.it
- Giusti Kevin, matr. 1092345, e-mail s1092345@studenti.univpm.it

Università Politecnica delle Marche
Facoltà di Ingegneria Informatica e dell'Automazione
Corso di Programmazione Mobile 2021/22

Generalità

classificazione app, utenti target e software simili

WIP: Work Is Progress è un'applicazione mobile classificabile mediante le seguenti app-categories:

- **Productivity:** categoria di software che aiuta l'utente nella gestione e nell'esecuzione di compiti quotidiani;
- **Time management:** categoria di software che permette all'utente di gestire il tempo;
- **Focus keeping:** categoria di software che impedisce l'uso di alcune funzionalità del device al fine di aumentare la concentrazione dell'utente durante le attività quotidiane;
- **Interactive study:** categoria di software che invoglia l'utente alla produttività mediante l'uso di un device;
- **Pixel art video-game graphic:** categoria di software con UI disegnata in pixel art;

In particolare, l'insieme dei requisiti funzionali e non funzionali derivanti dalla tassonomia appena presentata colloca WIP tra le applicazioni native poiché:

- le native-apps funzionano completamente anche offline;
  nel nostro caso, garantire che l'app sia affidabile e persistente è un requisito non funzionale critico banalmente esplicabile attraverso la seguente implicazione logica:
  l'utente si rivolge alla nostra app se è conscio di avere una soglia di distrazione molto alta e/o è demotivato ⇒ l'app deve funzionare correttamente e totalmente per l'intero tempo di attività dell'user poiché, se così non fosse, essa favorirebbe perdita di concentrazione;
  In conclusione, il software non può dipendere dall'assenza o dalla presenza di connettività nemmeno per le funzionalità di minor rilievo;
- le native-apps si basano su tool del sistema operativo;
  nel nostro caso, il software interagisce ampiamente e continuamente con il S.O per:
  - calcolo del tempo di attività dell'utente; in breve, il cronometro si basa sul clock che conta il tempo di attività del device;
  - rilevare l'activity corrente in foreground;
  - leggere/scrivere dal/sul database in memoria locale;
  - rilevare il lock e l'unlock dello schermo;

- le native-apps sono molto più veloci ed affidabili rispetto alle altre app, e quindi offrono una User Experience di più alto livello;
  Nel nostro caso, la velocità delle operazioni da eseguire è essenziale poiché l'utente deve utilizzare il software solo ed esclusivamente per configurare i parametri della sua sessione di produttività;
  il restante tempo di vita dell'app deve essere in background.

Per quanto riguarda gli utenti target, WIP è un'applicazione di produttività rivolta principalmente a studenti ma che, in generale, può essere utilizzata da qualsiasi individuo necessiti di tracking del tempo e di meccanismi di blocco temporaneo del dispositivo che favoriscono benessere digitale e focus keeping;

Per concludere, è importante citare le applicazioni di riferimento che hanno permesso la progettazione di WIP:

- Forest - Stay focused, be present:
  - dev company: Seekrtech
  - descrizione: app di produttività e di smartphone disintoxication
  - riferimenti: è possibile apprezzare tutte le qualità di Forest nella sezione "informazioni su questa app" presente nel fondo della pagina: https://play.google.com/store/apps/details?id=cc.forestapp&gl=IT
- HourBuddy - Monitoraggio del tempo e produttività:
  - dev company: NeuronDigital
  - descrizione: app per il monitoraggio del tempo, per la produttività e per la produzione di statistiche in base alle attività svolte
  - riferimenti: è possibile apprezzare tutte le qualità di HourBuddy nella sezione "informazioni su questa app" presente nel fondo della pagina: https://play.google.com/store/search?q=hourbuddy&c=apps&gl=IT

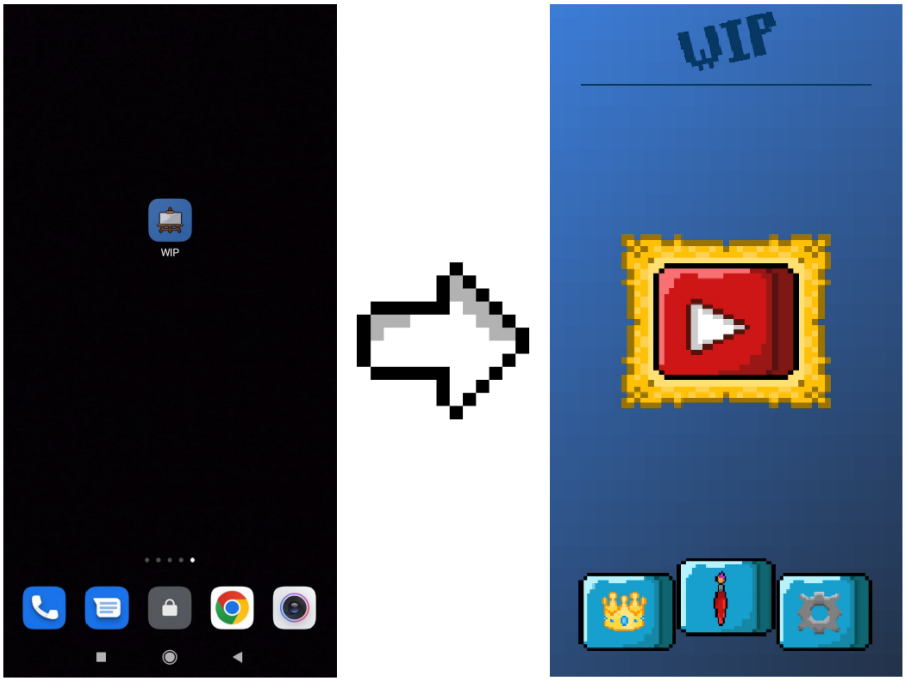
Raccolta dei requisiti

user story e UI navigation

Nota vocabolario: i vocaboli contrassegnati con "*****" compaiono nel glossario dei termini.
Nota grafica: la risoluzione delle immagini aumenta zoomando su di esse.

1. **Avvio app:**
   Michelangelo è un ragazzo che desidera cominciare a studiare/lavorare in modo produttivo e senza distrazioni e, per questo motivo, decide di utilizzare WIP.
   Cliccando sull'icona dell'applicazione, a Michelangelo viene mostrata la schermata principale che prevede:
   - il menù nella parte bassa dello schermo;
   - il pulsante "play" necessario all'avvio di una nuova storia*;

   graficamente:



2. **Navigazione mediante menù:**
   Michelangelo, incuriosito, decide di navigare nell'applicazione mediante l'utilizzo del menù posto nella parte bassa dello schermo;
   tale menù è costituito da tre bottoni:
   - il bottone "pennello", che serve a mostrare a Michelangelo la schermata principale se egli sta visualizzando un'altra schermata;
   - il bottone "corona", che serve a mostrare a Michelangelo la schermata "Kingdom", ovvero lo storico di tutte le sue storie concluse, se egli sta

visualizzando un'altra schermata.
per maggiori info, consulta la user story 6;

Graficamente:

Per promuovere maggiore chiarezza, avanziamo il seguente esempio: Michelangelo si trova nella schermata principale; dunque:

- se preme il pulsante "pennello", non accade nulla;
- se preme il pulsante "corona", viene mostrata la schermata "Kingdom";

3. **Avvio storia:**

Michelangelo ha già avviato l'app e, in questo momento, sta visualizzando la schermata principale;
Per iniziare una nuova storia, ovvero una nuova sessione di produttività, Michelangelo clicca il pulsante "play" e viene reindirizzato alla schermata di impostazione dei parametri della storia;
In questa schermata, Michelangelo :

- scrive, mediante tastiera, il titolo della nuova storia;
- definisce la partizione studio-pausa su una linea temporale di 60 min; ad esempio:
    - selezionando 50 min di studio si impostano automaticamente 10 min di pausa;

    - selezionando 40 min di studio si impostano automaticamente 20 min di pausa, ecc…
- seleziona le impostazioni opzionali(se l'applicazione non ha i permessi per impostare tali modalità, l'utente verrà reindirizzato ad una schermata capace di modificare suddetti permessi per poi consentire all'utente di selezionare la modalità desiderata. Questo verrà chiesto all'utente una sola volta per tutto il tempo in cui l'app resta installata sul dispositivo):
    - se si opziona "modalità silenziosa", il telefono va in modalità silenziosa*);
    - se si opziona "modalità hardcore", il telefono va in modalità silenziosa e se si esce* dall'app la storia termina automaticamente;
    - è possibile non selezionare alcuna opzione;
    - è possibile visualizzare a schermo informazioni relative a "modalità silenziosa" e "modalità hardcore" premendo l'apposito tasto "info";
- seleziona il suo avatar* mediante pulsanti a forma di frecce direzionali;
- se preme il tasto "indietro", Michelangelo esce dalla schermata di impostazione dei parametri della storia e torna alla schermata principale dell'app;
- se preme il tasto "start", Michelangelo inizia ufficialmente la sua sessione di produttività e, di conseguenza, gli viene mostrata la schermata "storia avviata".

graficamente:

4. **Sviluppo storia:**
   Michelangelo ha ufficialmente iniziato la sua sessione di produttività e, di conseguenza, sta visualizzando la schermata "storia avviata" che contiene:

- il cronometro che indica da quanto tempo la storia è stata avviata;
- il quadro da dipingere durante la sua sessione di produttività di 60 min;
- il pulsante “stop” che, se premuto, termina la storia;

graficamente:

5. **Conclusione storia:**

Mentre sta visualizzando la schermata “storia avviata”, Michelangelo decide di voler concludere la propria sessione di produttività e, di conseguenza, preme il tasto “stop”; In questo modo:

- il cronometro si ferma;
- le informazioni relative alla storia appena conclusa vengono salvate nella schermata “Kingdom” (solo se il tempo segnato dal cronometro è maggiore o uguale a 10 secondi);
- Michelangelo viene riportato nella schermata principale dell’app;

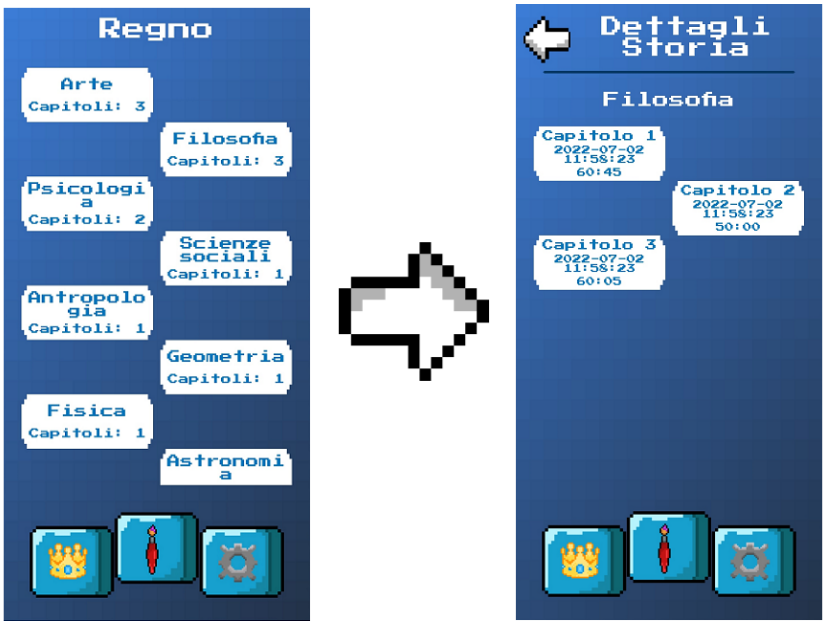
graficamente:

6. **Le mie storie:**
   Michelangelo, dopo aver creato e completato con successo numerose storie, vuole controllare i progressi compiuti;
   Per questo motivo, Michelangelo sta visualizzando la schermata "Kingdom" che contiene:
   - la scaletta di storie create fino a questo istante;

   graficamente:

7. **I miei capitoli*:**
   Michelangelo si trova nella schermata "Kingdom" e vuole visualizzare i dettagli di una storia. Dunque, clicca sulla storia di interesse e viene reindirizzato nella schermata "Dettagli storia" che mostra:
   - il titolo della storia selezionata;
   - tutti i capitoli in cui si articola la storia selezionata;

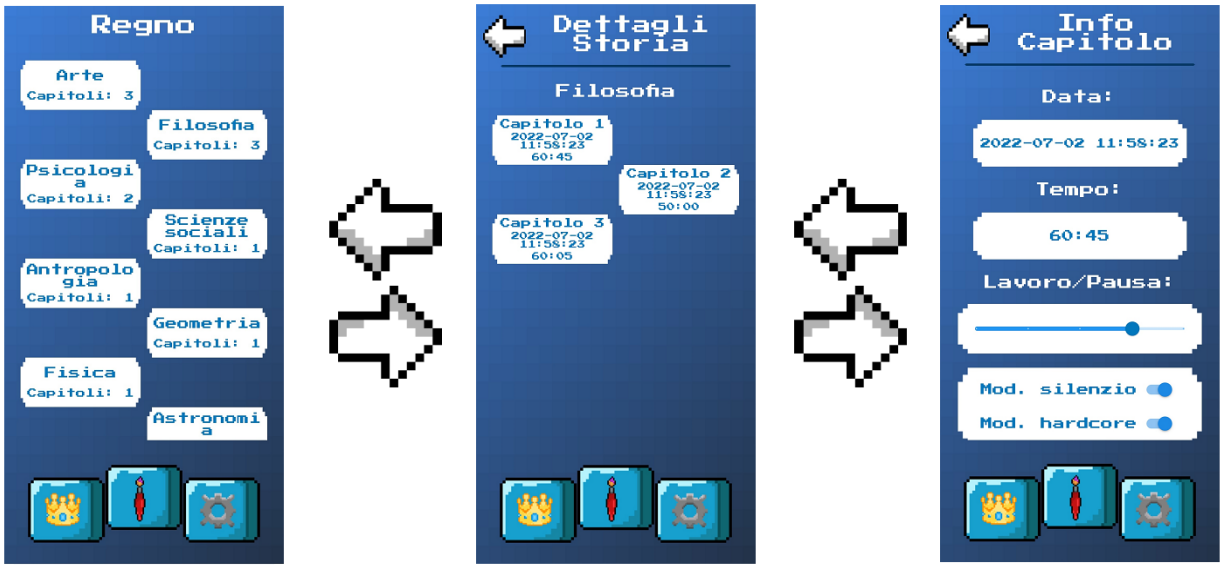
   anche in questo caso, i capitoli sono presentati mediante scaletta;

graficamente: 

nella schermata "Dettagli storia", Michelangelo può:

- tornare alla schermata "Kingdom" premendo il pulsante "indietro";
- apprezzare i dettagli di un capitolo cliccando sul capitolo di interesse;

graficamente:



Nella schermata "Dettagli capitolo", Michelangelo può:

- tornare indietro mediante il bottone "indietro";
- visualizzare data ed ora di inizio del capitolo;
- visualizzare il tempo totale della sessione di produttività;
- visualizzare la partizione studio/pausa selezionata;
- visualizzare la modalità opzionale selezionata;
- visualizzare l'avatar selezionato.

Analisi dei requisiti

divisione delle user story in task

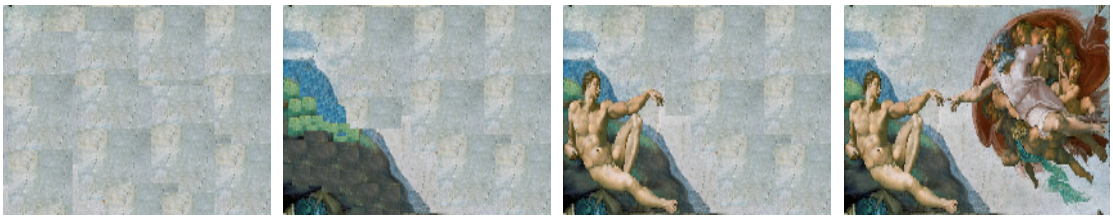
Nota: non tutte le user story necessitano di essere divise in task;
infatti, i task sono approfondimenti dei concetti già espressi nelle user story, utili solo a progettisti e programmatori;

1. **User story 3: Avvio storia:**
    - **Task 1:** autocompletamento: se l'utente vuole inserire, come titolo della storia corrente, un titolo già utilizzato precedentemente, il sistema suggerisce automaticamente il completamento del titolo desiderato;
    - **Task 2:** slide seekbar: l'utente definisce la partizione studio-pausa su una linea temporale che, di default, è di 60 min;
      le partizioni studio/pausa limite configurabili dall'utente sono:
        - [10 min studio, 50 min pausa]
        - [50 min studio, 10 min pausa]

      in altre parole, non è possibile configurare sessioni di produttività che:
        - sono costituite da 0 minuti di studio e da 60 min di pausa;
        - sono costituite da 60 minuti di studio e da 0 min di pausa;

      per concludere, i valori assumibili dai minuti di studio e dai minuti di pausa sono solo ed esclusivamente multipli di 10.
    - **Task 3:** modalità silenziosa:
        - se si opziona "modalità silenziosa", il telefono va in modalità silenziosa;
        - se si opziona "modalità silenziosa" ed il telefono è già in modalità silenziosa, allora non accade nulla;
        - se si deseleziona la "modalità silenziosa", le suonerie vengono ripristinate;
    - **Task 4:** modalità hardcore:
        - se si opziona "modalità hardcore", il telefono va in modalità silenziosa e se si esce dall'app la storia termina automaticamente;
        - se si opziona "modalità hardcore" ed il telefono è già in modalità silenziosa, allora viene impostato solo il parametro di terminazione automatica della storia in caso di uscita dall'app;
        - se si deseleziona la "modalità hardcore", le suonerie vengono ripristinate e la storia non termina automaticamente in caso di uscita dall'app;
        - se si seleziona "modalità hardcore" e si esce dall'app, i dati della storia non vengono memorizzati;

2. **User story 4: Sviluppo storia:**
   - **Task 1:** selezione casuale: quando si avvia una nuova sessione di produttività, al centro dello schermo, viene mostrato un quadro disegnato in pixel art; in particolare:
     - il quadro è selezionato randomicamente dal sistema;
     - i quadri selezionabili dal sistema sono solo ed esclusivamente quelli sbloccati di default e quelli acquistati dall'utente;
   - **Task 2:** algoritmo evolutivo: i quadri evolvono nel tempo; infatti, ogni quadro è costituito da quattro stati;
     graficamente:



STATO 1 **STATO 2** **STATO 3** **STATO 4**

     Matematicamente, l'evoluzione dei quadri dipende dal tempo di studio selezionato dall'utente;
     infatti, tale tempo di studio permette di definire quattro intervalli temporali di evoluzione calcolati nel seguente modo:
     - sia x il tempo di studio selezionato;
     - sia $n = 4$ il numero degli stati di ogni quadro;
     - sia $y = \frac{x}{n}$ il right value del primo intervallo di tempo;
     - il primo intervallo di tempo è $[0, y]$ ed, in esso, viene mostrato lo stato 1 del quadro;
     - il secondo intervallo di tempo è $(y, 2y]$ ed, in esso, viene mostrato lo stato 2 del quadro;
     - il terzo intervallo di tempo è $(2y, 3y]$ ed, in esso, viene mostrato lo stato 3 del quadro;
     - il quarto intervallo di tempo è $(3y, x]$ ed, in esso, viene mostrato lo stato 4 del quadro, ovvero il quadro completo;
     - durante il tempo di pausa viene mostrato il quadro completo.
   - **Task 3:** switch avatar-falò: quando il cronometro entra nell'arco di durata della pausa, si ha che:
     - l'avatar selezionato dall'utente viene sostituito da un falò;
     - la frase di incoraggiamento viene sostituita da una frase di invito al riposo;

- **Task 4:** switch falò-avatar: quando il cronometro esce dall'arco di durata della pausa, si ha che:
    - il falò viene sostituito con l'avatar selezionato dall'utente;
    - la frase di invito al riposo viene sostituita da una frase di incoraggiamento;

  graficamente:

- **Task 5:** ciclicità: la partizione studio-pausa selezionata dall'utente nella schermata di impostazione dei parametri della storia viene ripetuta ciclicamente fino a che non si preme il pulsante "stop" nella schermata "storia avviata";

  in altre parole, l'algoritmo da implementare è il seguente:
    - sia x il tempo di studio selezionato;
    - sia y il tempo di pausa selezionato;
    - sia $z = x + y$ il tempo di una sessione di studio-pausa;
    - sia k il tempo calcolato dal cronometro;

  allora, avremo che:
    - **passo 1:** se $k < x$, viene mostrato a schermo il quadro da evolvere e l'avatar che esprime frasi di incoraggiamento;
    - **passo 2:** se $x < k < z$, viene mostrato a schermo il quadro completo e il falò con frasi di invito al riposo;
    - **passo 3:** se $k > z$, allora si ha che:
        - $z \mathrel{+}= z$
        - $x = x + z$

        - il sistema carica randomicamente un nuovo quadro da dipingere;
        - il nuovo quadro da dipingere selezionato dal sistema deve necessariamente essere diverso da quello selezionato al ciclo precedente.
    - ripetiamo i passi 1, 2 e 3 fino a che non si preme il pulsante "stop";

3. **User story 5: Conclusione storia:**
    - **Task 1:** salvataggio dei dati della storia nella schermata "Kingdom":
      se, nella schermata "Kingdom", è già presente una storia con lo stesso nome di quello inserito dall'user, allora tale nuova storia viene salvata come capitolo della storia già esistente;
      in caso contrario, si crea una nuova storia.

4. **User story 6: Le mie storie:**
    - **Task 1:** struttura delle storie: in relazione alla scaletta delle storie, è importante notare che:
        - le storie sono ordinate dalla più recente alla meno recente;
        - ogni storia è costituita dal nome e dal numero di capitoli in cui si articola;
        - il numero di capitoli di ogni storia è calcolato automaticamente dal sistema; ogni storia ha almeno un capitolo;
        - le sessioni di produttività, definite nella schermata di impostazione dei parametri della storia, che hanno un titolo originale costituiscono una nuova storia;
          quelle prive di titolo originale, al contrario, costituiscono un capitolo di una storia pre-esistente;
          esempio:
            - l'utente avvia una nuova sessione di produttività intitolata "Fisica 1";
            - quando l'utente termina la sessione di produttività, viene creata una nuova storia nella schermata "Kingdom";
            - l'utente avvia nuovamente una sessione di produttività intitolata "Fisica 1";
            - quando l'utente termina la sessione di produttività, viene creato un nuovo capitolo della storia "Fisica 1" nella schermata "Kingdom"; ecc…
        - le storie sono composte da capitoli poiché, in questo modo, l'utente può intendere quanto tempo e quanti sforzi dedica ad una singola attività.

Analisi dei requisiti

glossario dei termini

Termine	Descrizione	Tipo	Sinonimi
Storia	Sessione di produttività compresa tra il primo ciclo di clock del cronometro e la pressione del tasto “stop” da parte dell’utente; ogni storia deve essere almeno di 10 secondi	Termine specifico dell’app	Sessione di produttività
Modalità silenziosa	Modalità che sostituisce il suono delle notifiche e delle chiamate con vibrazione	Smartphone feature	/
Uscire dall’app	Porre l’app in background aprendo applicazioni terze o premendo i tasti home, menù ed indietro; l’uscita dall’app non include la chiusura mediante task-manager	Smartphone feature	/
Avatar	Personaggio selezionabile dall’utente	Videogames	/
Sblocco del cellulare	Wake up del device dopo che esso è stato messo in sleep mode dall’utente o da un tempo prolungato di inattività; wake up ⇒ device diviene interattivo sleep mode ⇒ device perde interattività	Smartphone feature	/
Capitolo	Storia il cui titolo è già stato precedentemente utilizzato dall’utente nel sistema	Termine specifico dell’app	/

Analisi dei requisiti

requisiti utente: requisiti funzionali e non funzionali

Schematizzazione generale dei requisiti del sistema

Requisiti

Funzionali

+ Iniziare una nuova storia
+ Sviluppare una nuova storia
+ Concludere una nuova storia
+ Consultare i propri progressi

Non funzionali

+ Requisiti non funzionali che non richiedono permessi
+ Requisiti non funzionali che richiedono permessi non dangerous

Requisiti funzionali e descrizione strutturata

Funzionali

Iniziare una nuova storia

+ **RF1:** Scrivere il titolo della storia
+ **RF2:** Selezionare la partizione studio/pausa
+ **RF3:** Possibilità di impostare la modalità silenziosa
+ **RF4:** Possibilità di impostare la modalità hardcore
+ **RF5:** Selezionare l'avatar
+ **RF6:** Avviare la sessione di produttività

Sviluppare una nuova storia

+ **RF7:** Calcolare il tempo trascorso mediante cronometro
+ **RF8:** Mostrare avatar selezionato dall'utente e falò in relazione al tempo trascorso
+ **RF9:** Selezionare randomicamente il quadro da evolvere
+ **RF10:** Evolvere il quadro in relazione al tempo trascorso
+ **RF11:** Selezionare randomicamente un nuovo quadro quando un ciclo studio/pausa termina fino a che l'utente non preme il tasto "stop"

Concludere una nuova storia

+ **RF12:** Concludere la storia
+ **RF13:** Memorizzare i dati della storia conclusa

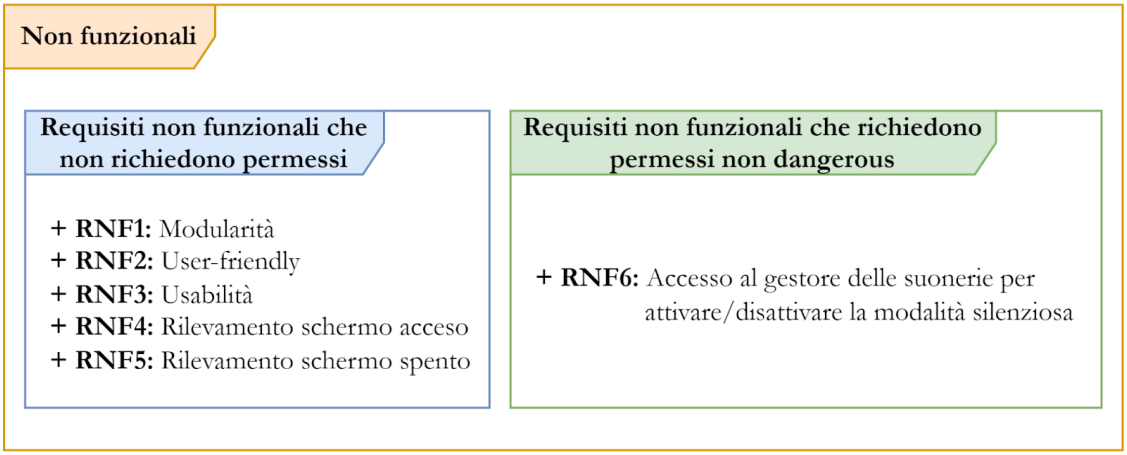
Consultare i propri progressi

+ **RF14:** Navigare tra le storie mediante scorrimento
+ **RF15:** Navigare tra capitoli mediante scorrimento
+ **RF16:** Produrre info sul capitolo selezionato

Requisito	Descrizione
RF1: Scrivere il titolo della storia	L'utente deve necessariamente scrivere, mediante tastiera, il titolo della storia che sta avviando; in caso contrario, il sistema impedisce l'avvio della sessione di produttività
RF2: Selezionare la partizione studio/pausa	L'utente seleziona, mediante slide con il dito, la partizione studio-pausa desiderata; se l'user non interagisce con la seekbar, viene impostata la partizione di default
RF3: Possibilità di impostare la modalità silenziosa	L'utente attiva la modalità silenziosa, ovvero disattiva le suonerie e attiva la vibrazione
RF4: Possibilità di impostare la modalità hardcore	L'utente attiva la modalità silenziosa e, se esce dall'app, la sua storia termina
RF5: Selezionare l'avatar	L'utente seleziona, mediante bottoni per lo scorrimento, l'avatar desiderato; se l'utente non seleziona alcun avatar, il sistema considera l'opzione di default
RF6: Avviare la sessione di produttività	L'utente avvia ufficialmente la propria sessione di produttività; a seconda delle impostazioni opzionali selezionate, l'app rileva se viene posta in background
RF7: Calcolare il tempo trascorso mediante cronometro	Mediante l'uso del clock che traccia il tempo di attività totale del device, il cronometro dell'app calcola il tempo della storia
RF8: Mostrare avatar selezionato dall'utente e falò in relazione al tempo trascorso	La scelta dell'avatar da parte dell'utente deve essere memorizzata e mostrata nelle schermate di interesse; se l'istante di tempo calcolato dal cronometro appartiene al tempo di studio, allora viene mostrato a schermo l'avatar; al contrario, se l'istante di tempo calcolato dal cronometro appartiene al tempo di pausa, allora viene mostrato

	a schermo il falò;
RF9: Selezionare randomicamente il quadro da evolvere	Il sistema seleziona randomicamente il quadro al centro dello schermo
RF10: Evolvere il quadro in relazione al tempo trascorso	Il tempo di studio è scisso in 4 intervalli temporali; in base all'intervallo temporale a cui appartiene l'istante corrente calcolato dal cronometro, il quadro evolve; l'utente vede il quadro completo al termine del tempo di studio
RF11: Selezionare randomicamente un nuovo quadro quando un ciclo studio/pausa termina fino a che l'utente non preme il tasto "stop"	Quando il tempo calcolato dal cronometro eccede il tempo studio+pausa, viene selezionato un nuovo quadro (facendo attenzione che sia diverso dal precedente) e si ripete il ciclo evolutivo; tale algoritmo viene ripetuto finché l'utente non termina esplicitamente la propria sessione di produttività
RF12: Concludere la storia	La sessione di produttività corrente può essere chiusa mediante apposito bottone "stop" o, alternativamente, ponendo l'app in background se si è in modalità hardcore
RF13: Memorizzare i dati della storia conclusa	Una volta eseguito il calcolo delle monete, i dati della storia vanno memorizzati nella schermata "Kingdom"
RF14: Navigare tra le storie mediante scorrimento	Le lista delle storie dell'utente deve consultabile mediante scorrimento verticale
RF15: Navigare tra capitoli mediante scorrimento	Le lista dei capitoli di ogni storia dell'utente deve consultabile mediante scorrimento verticale
RF16: Produrre info sul capitolo selezionato	Quando l'utente accede al capitolo di una storia, vengono mostrate informazioni a riguardo

Requisiti non funzionali e descrizione strutturata



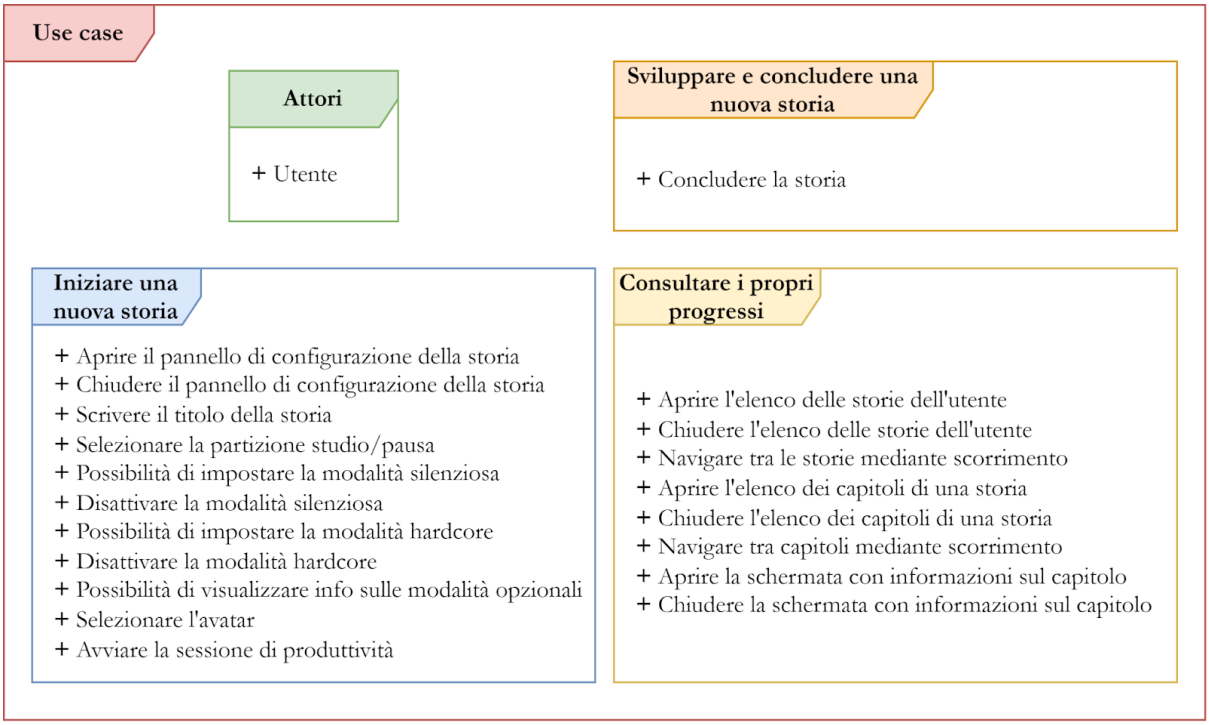
Requisito	Descrizione
RNF1: Modularità	Il sistema dovrà essere composto da moduli (classi)
RNF2: User-friendly	L'interfaccia grafica del sistema dovrà essere facilmente utilizzabile da ogni utente
RNF3: Usabilità	Il sistema dovrà essere usabile per il tipo di utenti per i quali è progettato
RNF4: Rilevamento schermo acceso	Il sistema deve essere in grado di rilevare quando l'utente accende lo schermo; infatti, l'accensione dello schermo è un trigger per eseguire alcune funzionalità durante la sessione di produttività corrente
RNF5: Rilevamento schermo spento	Il sistema deve essere in grado di rilevare quando l'utente spegne lo schermo; infatti, lo spegnimento dello schermo è un trigger per eseguire alcune funzionalità durante la sessione di produttività corrente
RNF6: Accesso al gestore delle suonerie per attivare/disattivare la modalità silenziosa	Il sistema deve poter accedere al gestore delle suonerie per poter attivare/disattivare la modalità silenziosa

Analisi dei requisiti

requisiti funzionali sistema: UML use-case diagram

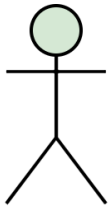
Utilizziamo la seguente notazione insiemistica per dividere, in aree funzionali, i diagrammi dei casi d'uso:

Nota: per rilevare facilmente ciascuna delle aree funzionali nelle sezioni di documento che seguono, è possibile utilizzare il colore.



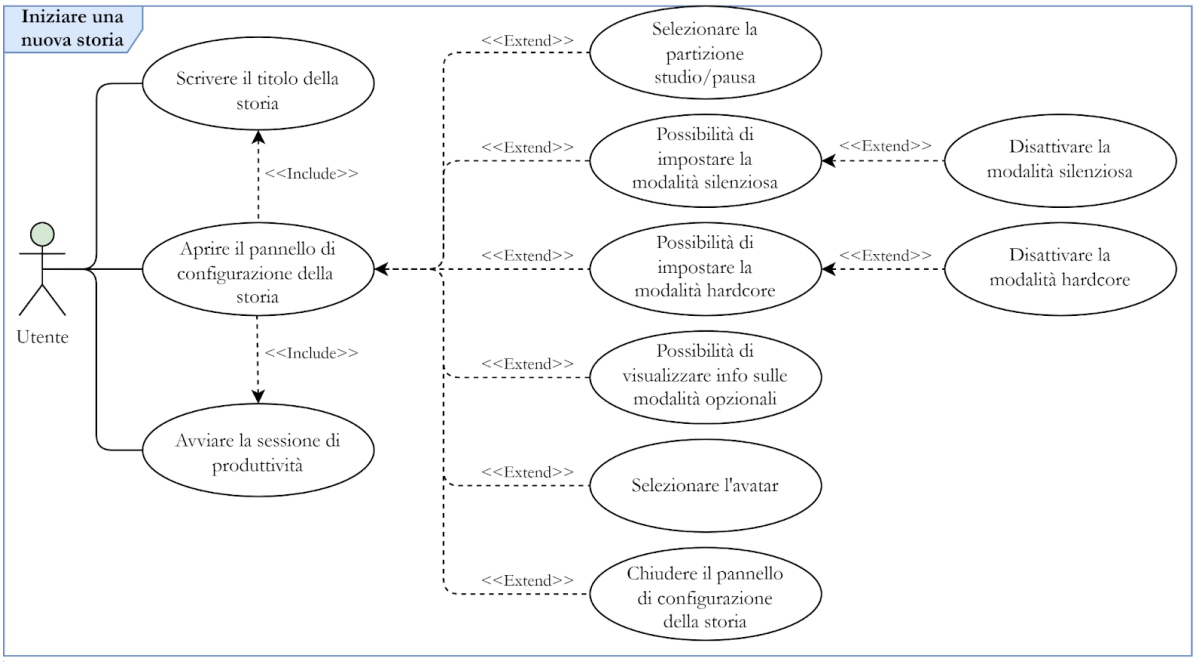
Attori use case

Poiché l'oggetto della trattazione è un software di produttività che non richiede autenticazione per erogare servizi, l'unico attore ad interagire con il sistema è l'utente generico.



Utente

Diagramma use case: iniziare una nuova storia



Descrizione strutturata dei casi d'uso

1. **Use case:** Aprire il pannello di configurazione della storia
    - **Descrizione:** Questo caso d'uso si verifica quando un utente intende avviare una nuova sessione di produttività.
    - **Pre-condizioni:** Nessuna.
    - **Post-condizioni:** Nessuna.
    - **Sequenza degli eventi principale:**
        a. il caso d'uso inizia quando un utente vuole avviare una nuova sessione di produttività.
        b. il sistema mostra il tasto di configurazione dei parametri della storia.
        c. l'utente preme il tasto di configurazione dei parametri della storia.
        d. il sistema mostra il pannello di configurazione della storia.
    - **Sequenza degli eventi alternativa:** Nessuna.

2. **Use case:** Scrivere il titolo della storia
    - **Descrizione:** Questo caso d'uso si verifica quando un utente intende assegnare un nome alla propria sessione di produttività.
      In particolare, il titolo è l'unico parametro ad essere dichiarato <<Include>> poiché non prevede valori di default;
      La scelta di non assegnare valori di default al titolo non è dovuta a difficoltà implementative, ma è dovuta alla volontà dei progettisti di costringere l'utente ad assegnare identità alle attività che lo formano.

- **Pre-condizioni:** il pannello di configurazione della storia deve essere aperto.
- **Post-condizioni:** Nessuna.
- **Sequenza degli eventi principale:**
    a. il caso d'uso inizia quando un utente intende assegnare un nome alla propria sessione di produttività.
    b. il sistema mostra la view "editText" per ricevere il testo in input.
    c. l'utente preme la view "editText" per immettere il testo in input.
    d. il sistema mostra la tastiera all'utente.
    e. l'utente immette il titolo mediante tastiera.
    f. il sistema mostra il titolo della storia appena immesso.
- **Sequenza degli eventi alternativa:** Nessuna.

3. **Use case:** Avviare la sessione di produttività
    - **Descrizione:** Questo caso d'uso si verifica quando un utente ha configurato tutti i parametri della sua storia ed intende avviare la propria sessione di produttività.
    - **Pre-condizioni:** il pannello di configurazione della storia deve essere aperto ed il titolo della storia deve essere stato immesso.
    - **Post-condizioni:** Nessuna.
    - **Sequenza degli eventi principale:**
        a. il caso d'uso inizia quando un utente, dopo aver configurato tutti i parametri della sua storia, intende avviare la propria sessione di produttività mediante apposito bottone.
        b. il sistema mostra il bottone per avviare la storia.
        c. l'utente preme il bottone per avviare la storia.
        d. il sistema mostra la schermata relativa allo svolgimento della storia.
    - **Sequenza degli eventi alternativa:** Nessuna.

4. **Use case:** Selezionare la partizione studio-pausa
    - **Descrizione:** Questo caso d'uso si verifica quando un utente intende selezionare la partizione studio-pausa.
    - **Pre-condizioni:** il pannello di configurazione della storia deve essere aperto.
    - **Post-condizioni:** Nessuna.
    - **Sequenza degli eventi principale:**
        a. il caso d'uso inizia quando un utente intende selezionare la partizione studio-pausa.
        b. il sistema mostra la seekbar che consente di impostare la partizione studio-pausa desiderata.

    c. l'utente interagisce con la seekbar mediante slide con il dito.
    d. il sistema legge l'input da sliding.
    e. il sistema mostra a schermo il tempo di studio ed il tempo di pausa a seconda della posizione del dito nella seekbar.
- **Sequenza degli eventi alternativa:** l'utente non interagisce con la seekbar e, di conseguenza, il sistema considera la partizione di default.

5. **Use case:** Possibilità di impostare la modalità silenziosa
    - **Descrizione:** Questo caso d'uso si verifica quando un utente intende attivare la modalità silenziosa.
    - **Pre-condizioni:** il pannello di configurazione della storia deve essere aperto.
    - **Post-condizioni:** Nessuna.
    - **Sequenza degli eventi principale:**
        a. il caso d'uso inizia quando un utente intende attivare la modalità silenziosa.
        b. il sistema mostra lo switch che consente di attivare la modalità silenziosa.
        c. l'utente interagisce con lo switch mediante tap con il dito.
        d. il sistema legge l'input da tocco.
        e. il sistema notifica all'utente, mediante vibrazione, che la modalità silenziosa è stata attivata.
    - **Prima sequenza degli eventi alternativa:** se l'utente ha già attivato la modalità silenziosa, l'attivazione dello switch non genera alcuna callback.
    - **Seconda sequenza degli eventi alternativa:** l'utente non interagisce con lo switch per l'attivazione della modalità silenziosa.

6. **Use case:** Disattivare la modalità silenziosa
    - **Descrizione:** Questo caso d'uso si verifica quando un utente intende disattivare la modalità silenziosa.
    - **Pre-condizioni:** il pannello di configurazione della storia deve essere aperto e l'utente deve aver attivato la modalità silenziosa mediante switch.
    - **Post-condizioni:** Nessuna.
    - **Sequenza degli eventi principale:**
        a. il caso d'uso inizia quando un utente intende disattivare la modalità silenziosa.
        b. il sistema mostra lo switch che consente di disattivare la modalità silenziosa.
        c. l'utente interagisce con lo switch mediante tap con il dito.
        d. il sistema legge l'input da tocco.

    e. il sistema notifica all'utente che la modalità silenziosa è stata disattivata e che le suonerie sono state ripristinate.
- **Sequenza degli eventi alternativa:** Nessuna.

7. **Use case:** Possibilità di impostare la modalità hardcore
    - **Descrizione:** Questo caso d'uso si verifica quando un utente intende attivare la modalità hardcore.
    - **Pre-condizioni:** il pannello di configurazione della storia deve essere aperto.
    - **Post-condizioni:** Nessuna.
    - **Sequenza degli eventi principale:**
        a. il caso d'uso inizia quando un utente intende attivare la modalità hardcore.
        b. il sistema mostra lo switch che consente di attivare la modalità hardcore.
        c. l'utente interagisce con lo switch mediante tap con il dito.
        d. il sistema legge l'input da tocco.
        e. il sistema attiva la modalità silenziosa.
        f. il sistema notifica all'utente, mediante vibrazione, che la modalità hardcore è stata attivata.
    - **Sequenza degli eventi alternativa:** l'utente non interagisce con lo switch per l'attivazione della modalità hardcore.

8. **Use case:** Disattivare la modalità hardcore
    - **Descrizione:** Questo caso d'uso si verifica quando un utente intende disattivare la modalità hardcore.
    - **Pre-condizioni:** il pannello di configurazione della storia deve essere aperto e l'utente deve aver attivato la modalità hardcore mediante switch.
    - **Post-condizioni:** Nessuna.
    - **Sequenza degli eventi principale:**
        a. il caso d'uso inizia quando un utente intende disattivare la modalità hardcore.
        b. il sistema mostra lo switch che consente di disattivare la modalità hardcore.
        c. l'utente interagisce con lo switch mediante tap con il dito.
        d. il sistema legge l'input da tocco.
        e. il sistema disattiva la modalità silenziosa.
        f. il sistema notifica all'utente che la modalità hardcore è stata disattivata e che le suonerie sono state ripristinate.
    - **Sequenza degli eventi alternativa:** Nessuna.

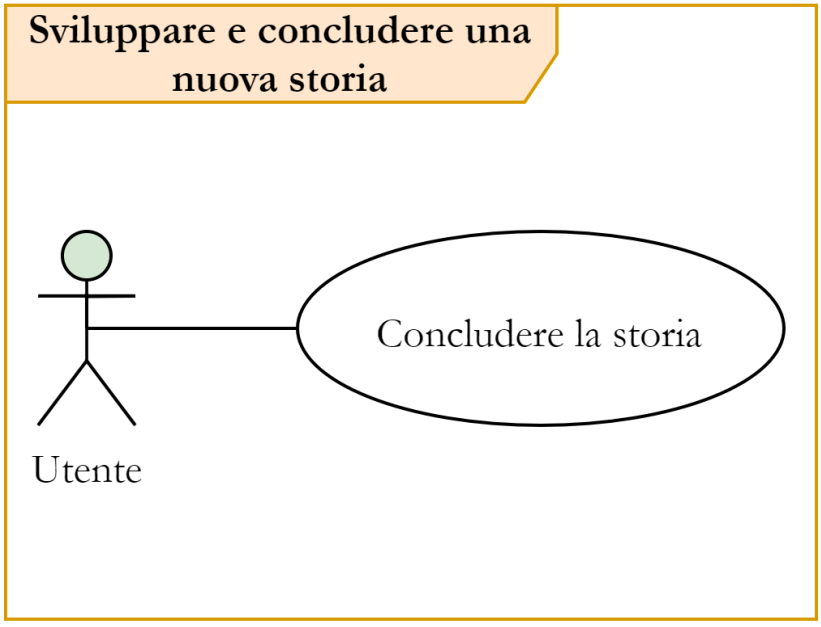
9. **Use case:** Possibilità di visualizzare info sulle modalità opzionali
    - **Descrizione:** Questo caso d'uso si verifica quando un utente intende visualizzare info sulle modalità opzionali.
    - **Pre-condizioni:** il pannello di configurazione della storia deve essere aperto.
    - **Post-condizioni:** Nessuna.
    - **Sequenza degli eventi principale:**
        a. il caso d'uso inizia quando un utente intende visualizzare info sulle modalità opzionali.
        b. il sistema mostra il pulsante che consente di visualizzare info sulle modalità opzionali.
        c. l'utente interagisce con il pulsante mediante tap con il dito.
        d. il sistema legge l'input da tocco.
        e. il sistema disattiva mostra le info sulle modalità opzionali.
    - **Sequenza degli eventi alternativa:** l'utente non interagisce con il pulsante che consente di visualizzare info sulle modalità opzionali.

10. **Use case:** Selezionare l'avatar
    - **Descrizione:** Questo caso d'uso si verifica quando un utente intende selezionare un avatar.
    - **Pre-condizioni:** il pannello di configurazione della storia deve essere aperto.
    - **Post-condizioni:** Nessuna.
    - **Sequenza degli eventi principale:**
        a. il caso d'uso inizia quando un utente intende selezionare un avatar.
        b. il sistema mostra i pulsanti direzionali che consentono di scorrere gli avatar disponibili.
        c. l'utente interagisce con i pulsanti mediante tap con il dito.
        d. il sistema legge l'input da tocco.
        e. il sistema mostra un nuovo avatar ogni volta che l'utente preme i pulsanti direzionali.
    - **Sequenza degli eventi alternativa:** l'utente non interagisce con i pulsanti e viene automaticamente selezionato l'avatar di default.

11. **Use case:** Chiudere il pannello di configurazione della storia
    - **Descrizione:** Questo caso d'uso si verifica quando un utente intende chiudere il pannello di configurazione della storia.
    - **Pre-condizioni:** il pannello di configurazione della storia deve essere aperto.

- **Post-condizioni:** Nessuna.
- **Sequenza degli eventi principale:**
    - **a.** il caso d'uso inizia quando un utente intende chiudere il pannello di configurazione della storia.
    - **b.** il sistema mostra il pulsante "indietro".
    - **c.** l'utente interagisce con il pulsante "indietro".
    - **d.** il sistema legge l'input da tocco.
    - **e.** il sistema chiude il pannello di configurazione della storia.
- **Sequenza degli eventi alternativa:** Nessuna.

Diagramma use case: sviluppare e concludere una nuova storia

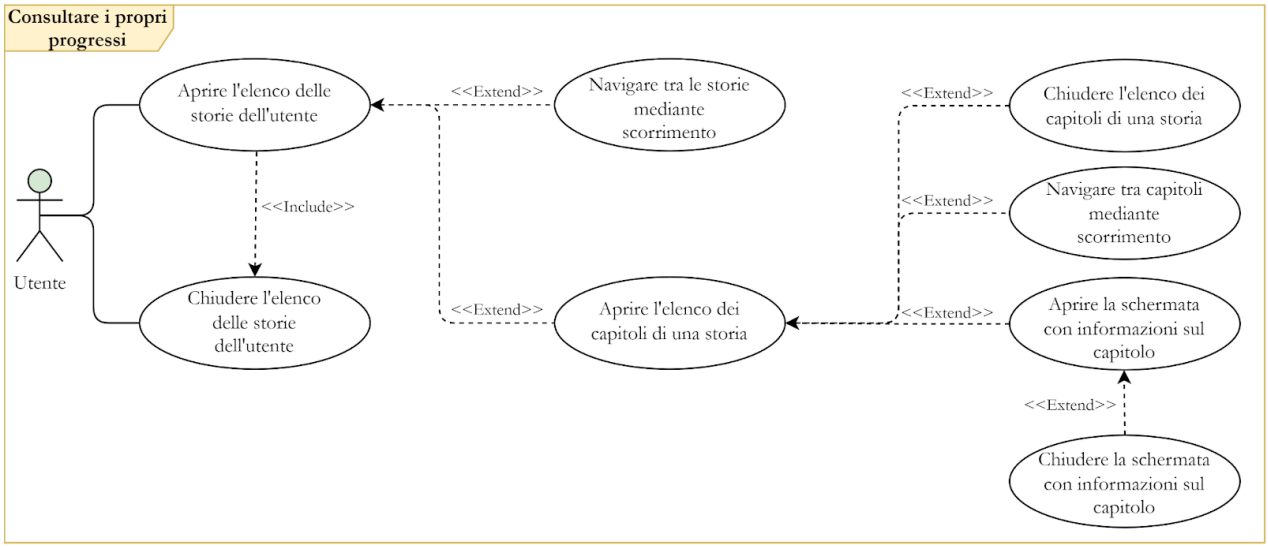


Descrizione strutturata dei casi d'uso

1. **Use case:** Concludere la storia
    - **Descrizione:** Questo caso d'uso si verifica quando un utente intende concludere la storia
    - **Pre-condizioni:** Nessuna.
    - **Post-condizioni:** Nessuna.
    - **Sequenza degli eventi principale:**
        - **a.** il caso d'uso inizia quando un utente intende concludere la storia.
        - **b.** il sistema mostra il pulsante per concludere la storia.
        - **c.** l'utente interagisce con il pulsante mediante input touch.
        - **d.** il sistema rileva la pressione del pulsante.
        - **e.** il sistema conclude la sessione di produttività.

- **Sequenza degli eventi alternativa:** se la modalità hardcore è attiva e l'utente pone WIP in background, il sistema conclude la sessione di produttività corrente.

Diagramma use case: consultare i propri progressi



Descrizione strutturata dei casi d'uso

1. **Use case:** Aprire l'elenco delle storie dell'utente
   - **Descrizione:** Questo caso d'uso si verifica quando un utente intende visualizzare lo storico delle sessioni di produttività svolte
   - **Pre-condizioni:** Nessuna.
   - **Post-condizioni:** Nessuna.
   - **Sequenza degli eventi principale:**
     a. il caso d'uso inizia quando un utente intende visualizzare lo storico delle sessioni di produttività svolte.
     b. il sistema mostra il pulsante d'accesso allo storico delle sessioni di produttività svolte.
     c. l'utente interagisce con il pulsante mediante input touch.
     d. il sistema rileva la pressione del pulsante.
     e. il sistema mostra a schermo lo storico delle sessioni di produttività svolte.
   - **Sequenza degli eventi alternativa:** Nessuna.

2. **Use case:** Chiudere l'elenco delle storie dell'utente
   - **Descrizione:** Questo caso d'uso si verifica quando un utente intende chiudere lo storico delle sessioni di produttività svolte

- **Pre-condizioni:** lo storico delle sessioni di produttività svolte dall'utente deve essere aperto.
- **Post-condizioni:** Nessuna.
- **Sequenza degli eventi principale:**
    a. il caso d'uso inizia quando un utente intende chiudere lo storico delle sessioni di produttività svolte.
    b. il sistema mostra il bottom menù.
    c. l'utente interagisce con il bottom menù.
    d. il sistema rileva l'interazione.
    e. il sistema chiude lo storico delle sessioni di produttività svolte.
- **Sequenza degli eventi alternativa:** Nessuna.

3. **Use case:** Navigare tra le storie mediante scorrimento
    - **Descrizione:** Questo caso d'uso si verifica quando un utente intende visualizzare storie non mostrate a schermo
    - **Pre-condizioni:** lo storico delle sessioni di produttività svolte dall'utente deve essere aperto.
    - **Post-condizioni:** Nessuna.
    - **Sequenza degli eventi principale:**
        a. il caso d'uso inizia quando un utente intende visualizzare le storie non mostrate a schermo.
        b. in generale, per ragioni legate alla lunghezza dei device, il sistema mostra a schermo un sottoinsieme di storie.
        c. l'utente interagisce con lo schermo effettuando scrolling verso il basso o verso l'alto al fine di cercare la storia desiderata.
        d. il sistema rileva l'interazione.
        e. il sistema permette la navigazione tra le storie mediante scorrimento.
    - **Sequenza degli eventi alternativa:** se tutte le storie entrano nello schermo, la funzionalità di scrolling viene temporaneamente disabilitata.

4. **Use case:** Aprire l'elenco dei capitoli di una storia
    - **Descrizione:** Questo caso d'uso si verifica quando un utente intende aprire l'elenco dei capitoli di una storia
    - **Pre-condizioni:** lo storico delle sessioni di produttività svolte dall'utente deve essere aperto.
    - **Post-condizioni:** Nessuna.
    - **Sequenza degli eventi principale:**
        a. il caso d'uso inizia quando un utente intende aprire l'elenco dei capitoli di una storia.

    b. il sistema mostra le storie disponibili a schermo.
    c. l'utente esegue tap sulla storia desiderata.
    d. il sistema rileva l'interazione.
    e. il sistema mostra l'elenco dei capitoli relativi alla storia selezionata.
- **Sequenza degli eventi alternativa:** Nessuna.

5. **Use case:** Chiudere l'elenco dei capitoli di una storia
    - **Descrizione:** Questo caso d'uso si verifica quando un utente intende chiudere l'elenco dei capitoli di una storia
    - **Pre-condizioni:** lo storico delle sessioni di produttività svolte dall'utente deve essere aperto; l'elenco dei capitoli di una storia deve essere aperto.
    - **Post-condizioni:** Nessuna.
    - **Sequenza degli eventi principale:**
        a. il caso d'uso inizia quando un utente intende chiudere l'elenco dei capitoli di una storia.
        b. il sistema mostra il tasto di uscita.
        c. l'utente preme il tasto di uscita.
        d. il sistema rileva l'interazione.
        e. il sistema chiude l'elenco dei capitoli relativi alla storia selezionata.
    - **Sequenza degli eventi alternativa:** Nessuna.

6. **Use case:** Navigare tra capitoli mediante scorrimento
    - **Descrizione:** Questo caso d'uso si verifica quando un utente intende visualizzare i capitoli non mostrati a schermo
    - **Pre-condizioni:** lo storico delle sessioni di produttività svolte dall'utente deve essere aperto; l'elenco dei capitoli di una storia deve essere aperto.
    - **Post-condizioni:** Nessuna.
    - **Sequenza degli eventi principale:**
        a. il caso d'uso inizia quando un utente intende visualizzare i capitoli non mostrati a schermo.
        b. in generale, per ragioni legate alla lunghezza dei device, il sistema mostra a schermo un sottoinsieme di capitoli.
        c. l'utente interagisce con lo schermo effettuando scrolling verso il basso o verso l'alto al fine di cercare il capitolo desiderato.
        d. il sistema rileva l'interazione.
        e. il sistema permette la navigazione tra i capitoli mediante scorrimento.
    - **Sequenza degli eventi alternativa:** se tutti i capitoli entrano nello schermo, la funzionalità di scrolling viene temporaneamente disabilitata.

7. **Use case:** Aprire la schermata con informazioni sul capitolo
    - **Descrizione:** Questo caso d'uso si verifica quando un utente intende visualizzare le informazioni di un capitolo
    - **Pre-condizioni:** lo storico delle sessioni di produttività svolte dall'utente deve essere aperto; l'elenco dei capitoli di una storia deve essere aperto.
    - **Post-condizioni:** Nessuna.
    - **Sequenza degli eventi principale:**
        a. il caso d'uso inizia quando un utente intende visualizzare le informazioni di un capitolo.
        b. il sistema mostra a schermo i capitoli della storia selezionata.
        c. l'utente seleziona il capitolo desiderato mediante tap su schermo.
        d. il sistema rileva l'interazione.
        e. il sistema mostra informazioni sul capitolo selezionato.
    - **Sequenza degli eventi alternativa:** Nessuna.

8. **Use case:** Chiudere la schermata con informazioni sul capitolo
    - **Descrizione:** Questo caso d'uso si verifica quando un utente intende nascondere le informazioni di un capitolo
    - **Pre-condizioni:** lo storico delle sessioni di produttività svolte dall'utente deve essere aperto; l'elenco dei capitoli di una storia deve essere aperto; la schermata con informazioni sul capitolo deve essere aperta.
    - **Post-condizioni:** Nessuna.
    - **Sequenza degli eventi principale:**
        a. il caso d'uso inizia quando un utente intende nascondere le informazioni di un capitolo.
        b. il sistema mostra a schermo le informazioni sul capitolo selezionato ed il tasto indietro.
        c. l'utente seleziona il tasto indietro.
        d. il sistema rileva l'interazione.
        e. il sistema nasconde le informazioni sul capitolo selezionato.
    - **Sequenza degli eventi alternativa:** Nessuna.

Progettazione database

schema concettuale, schema logico

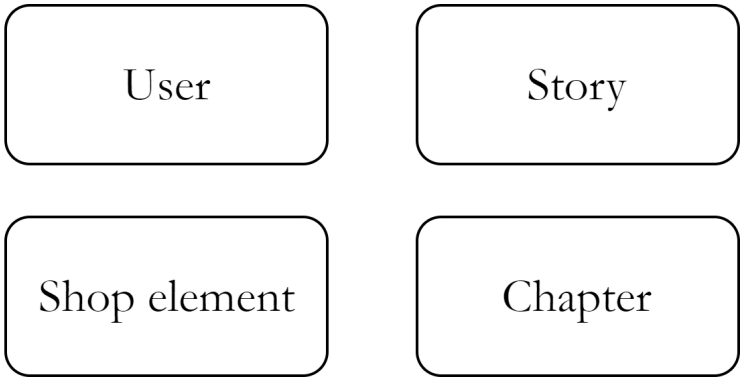
La gestione delle informazioni nel software Work Is Progress è delegata ad un database relazionale locale al dispositivo di ciascun utente;
In particolare, la scelta di utilizzare un DB fondato sul modello relazionale dei dati è dovuto alle seguenti motivazioni:

- le informazioni trattate nel software trovano naturale rappresentazione mediante tabelle, ovvero mediante relazioni matematiche;
- la rigidità dello schema facilita il mapping tra le entità nel DB e le classi nel codice sorgente;
- il modello relazionale offre un buon compromesso tra complessità della base di dati e complessità del codice sorgente dell'app;
  per intendere al meglio tale affermazione, si osservi che se il DB dell'app fosse stato un file JSON, ovvero un file potenzialmente schema-less, allora la complessità del codice sarebbe stata notevolmente superiore a quella della base di dati poiché si sarebbe dovuto gestire, con logica imperativa, i vincoli di dominio ed i vincoli di integrità;
- garanzia di assenza di occorrenze duplicate grazie alle chiavi primarie;

Per la progettazione della base di dati dell'app WIP, si è optato per l'adozione della strategia Top-Down: lo schema concettuale è stato prodotto mediante raffinamenti successivi, cioè sono stati identificati gli oggetti essenziali e, mediante raffinamenti, sono stati aggiunti dettagli;
in sintesi, lo schema concettuale è stato modellato, analizzando i requisiti, dal generale al particolare.

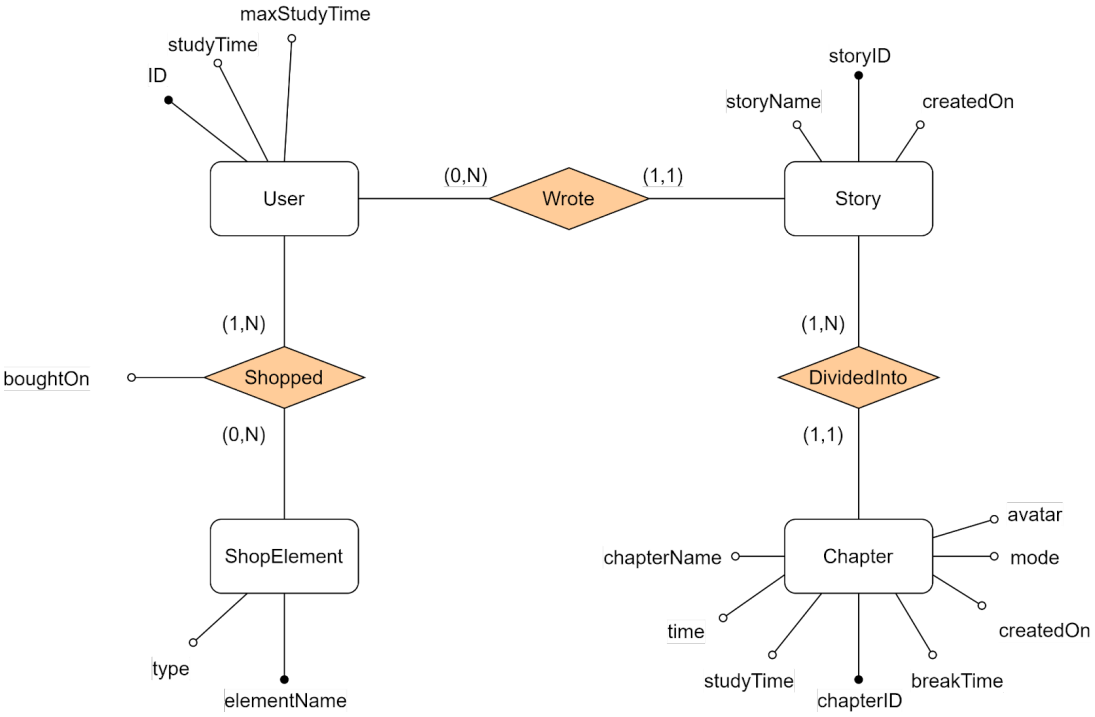
Il flusso di lavoro che descrive la modellazione del DB è costituito dai seguenti passi:

1. **Identificazione delle entità essenziali dello schema concettuale:**



le informazioni relative a tali entità sono apprezzabili nel dizionario dei dati.

2. **Schema concettuale e schema logico:**
   Poiché la realtà da modellare non si basa su astrazioni particolarmente complesse, in fase di traduzione del modello concettuale in modello logico si è osservato che, in realtà, schema logico e schema concettuale coincidono;



segue l'analisi dello schema logico:

- **Relazione tra le entità User e Shop element:**
  - **descrizione:** relationship che assegna all'utente le opere d'arte acquistate nello shop;
  - **cardinalità:**
    - User - Shopped (1, N): l'istanza "utente" appartiene almeno una volta alla relationship poiché l'app prevede opere d'arte gratuite, ovvero acquistate di default;
    - ShopElement - Shopped (0, N): la partecipazione opzionale è necessaria poiché non è certo che una specifica opera d'arte venga acquistata dall'utente; la cardinalità massima è N, e non 1, perchè anche se ogni utente può acquistare un'opera d'arte una sola volta, nelle versioni successive dell'app, si intende implementare una base di dati online che supporti più utenti e che:
      - salvi online i dati dell'utente in caso il dispositivo abbia accesso alla rete;
      - salvi in locale i dati dell'utente in caso il dispositivo non abbia accesso alla rete; segue poi il dump quando la connessione viene ripristinata;

in questo modo, si supporta il device swapping.

- **Relazione tra le entità User e Story:**
  - **descrizione:** relationship che assegna all'utente le sessioni di produttività svolte;
  - **cardinalità:**
    - User - Wrote (0, N): in generale, un utente svolge molte sessioni di produttività;
      tuttavia, la partecipazione opzionale è necessaria poiché è possibile che l'utente non avvii mai una storia;
      un esempio è dato dal primo avvio dell'app;
    - Story - Wrote (1, 1): la partecipazione obbligatoria è necessaria poiché una storia può esistere se e solo se è stata creata dall'utente;
- **Relazione tra le entità Story e Chapter:**
  - **descrizione:** relationship che assegna ad ogni storia il numero di capitoli in cui è divisa;
  - **cardinalità:**
    - Story - DividedInto (1, N): ogni storia ha almeno un capitolo;
    - Chapter - DividedInto (1, 1): la partecipazione obbligatoria è necessaria poiché un capitolo può esistere se e solo se appartiene ad una storia;

Dizionario dei dati

dizionario entità e relationships

La descrizione delle entità, delle relationship e degli attributi di entità e relationship è apprezzabile nel seguente dizionario dei dati:

Entità			
Nome	**Descrizione**	**Attributi**	**Identificatore**
User	Struttura dati che memorizza: ● le preferenze relative al time management dell'utente quando vengono configurati i parametri di una nuova sessione di produttività; ● le monete in possesso dell'utente;	Id **Tipo:** numerico-intero	Id
		studyTime **Tipo:** numerico-intero **studyTime:** tempo di studio nella partizione. studio-pausa	
		maxStudyTime **Tipo:** numerico-intero **maxStudyTime:** tempo studio sommato a tempo pausa	
Story	Struttura dati che memorizza le informazioni atte all'identificazione univoca di una storia	storyId **Tipo:** numerico-intero	storyId
		storyName **Tipo:** stringa	
		createdOn **Tipo:** stringa	
Shop element	Struttura dati che memorizza le informazioni atte alla profilazione di ciascuna opera d'arte;	elementName **Tipo:** stringa	elementName
		type **Tipo:** stringa **type:** avatar o quadro	
		chapterId **Tipo:** numerico-intero	
		chapterName **Tipo:** stringa	
		time	

Chapter	Struttura dati che memorizza tutte le informazioni relative ad una sessione di produttività	**Tipo:** stringa	chapterId
		studyTime **Tipo:** numerico-intero	
		breakTime **Tipo:** numerico-intero	
		avatar **Tipo:** stringa	
		mode **Tipo:** stringa	
		createdOn **Tipo:** stringa	

Relationship			
Nome	**Descrizione**	**Attributi**	**Entità coinvolte**
Shopped	Associa un elemento dello shop ad un utente	boughtOn **Tipo:** stringa	User (1,N) ShopElement (0,N)
Wrote	Associa un utente ad una storia	/	User (1,N) Story (1,1)
DividedInto	Associa una storia ai suoi capitoli	/	Story (1,N) Chapter (1,1)

Vincoli non esprimibili

vincoli intra-relazionali

Di seguito, sono riportati i vincoli che non è stato possibile esprimere mediante i costrutti del modello concettuale, ovvero mediante:

- entità
- relationship
- attributo
- cardinalità
- identificatore
- generalizzazione

ma che, comunque, sono di notevole importanza e significatività nel modello progettato:

3. l'attributo "studyTime", relativo all'entità "User", può assumere solo ed esclusivamente valori interi multipli di 10;
4. l'attributo "studyTime", relativo all'entità "User", può assumere solo ed esclusivamente valori interi al più uguali a $maxStudyTime - 10$
5. l'attributo "maxStudyTime", relativo all'entità "User", può assumere solo ed esclusivamente valori interi multipli di 10;
6. l'attributo "maxStudyTime", relativo all'entità "User", può assumere solo ed esclusivamente valori interi compresi tra 60 e 120;
7. l'attributo "type", relativo all'entità "ShopElement", può assumere solo ed esclusivamente valori "avatar" e "background"
8. l'attributo "avatar", relativo all'entità "Chapter", può assumere solo ed esclusivamente gli identificatori assegnati alle avatar-image-resources;
9. l'attributo "mode", relativo all'entità "Chapter", può assumere solo ed esclusivamente valori "none", "silent" e "hardcore".

Implementazione del database

traduzione nel modello relazionale

Note:

- le primary key sono sottolineate;
- le foreign key sono evidenziate mediante "*****"

Entità/relazione	Traduzione
User	User(ID, studyTime, maxStudyTime)
Story	Story(storyID, storyName, createdOn, user*)
Chapter	Chapter(chapterID, chapterName, time, createdOn, studyTime, breakTime, mode, avatar, story*)
ShopElement	ShopElement(elementName, type)
Shopped	Shopped(user*, ShopElement*, boughtOn)

Analisi dell'architettura

descrizione dei moduli dell'architettura dell'app

Nella sezione che segue vengono brevemente descritti i moduli appartenenti all'architettura dell'applicazione.

View

Per facilità di gestione, le view vengono chiamate tramite rotte:
"/" => SplashPage();
"/home" => Home();
"/start-story" => StartStory();
"/story-started" => StoryStarted();
"/kingdom" => Kingdom();
"/story-detail" => StoryDetail()
"/chapter-info" => ChapterInfo();
"/settings" => Settings().

Esistono però view che non vengono utilizzate da rotte ma che appartengono al layout di una view come "WIPMenu" e view che fungono da popup come "WIPDialog".

Metodo main():
l'applicazione parte da qui, richiamando il metodo runApp(), che utilizza la classe "WIP" per l'inizializzazione dell'applicazione stessa. Inoltre fa sì che durante la visualizzazione delle interfacce, la barra delle notifiche sia trasparente e che l'applicazione non possa essere messa in modalità landscape.

Classi in dettaglio

1. **WIP:**
    - **classe estesa:** StatelessWidget; viene eseguito l'override del metodo build();
    - **funzionalità:** inizializza l'applicazione dichiarando un titolo e le rotte precedentemente descritte. Inoltre dichiara la prima rotta che deve essere visualizzata all'avvio dell'applicazione.

2. **SplashPage:**

- **classe estesa:** StatelessWidget; viene eseguito l'override del metodo build();
- **funzionalità:** la classe SplashPage racchiude le seguenti funzionalità:
    - mostra la splash page dell'app e, dopo un secondo (1000 millisecondi), fa il push della rotta "/home".

3. **Home:**
    - **classe estesa:** StatefulWidget; viene eseguito l'override del metodo createState(), che utilizza la classe _HomeState per disegnare l'interfaccia.

4. **_HomeState:**
    - **classe estesa:** State; viene eseguito l'override dei metodi initState(), didChangeDependencies() e build();
    - **funzionalità:** la classe _HomeState racchiude le seguenti funzionalità:
        - recupera un'istanza del database. Questo fa sì che al primo avvio dell'app che essa è stata installata, il database venga popolato correttamente;
        - imposta nelle shared preferences l'id dell'utente(1);
        - tramite il pulsante "play" è possibile andare alla rotta "/start-story";
        - visualizza un istanza del widget personalizzato "WIPMenu".

5. **WIPMenu:**
    - **classe estesa:** StatefulWidget; viene eseguito l'override del metodo createState(), che utilizza la classe _WIPMenuState per disegnare l'interfaccia.

6. **_WIPMenuState:**
    - **classe estesa:** State; viene eseguito l'override dei metodi initState(), didChangeDependencies() e build();
    - **funzionalità:** la classe _WIPMenuState racchiude le seguenti funzionalità:
        - tramite il pulsante "kingdom" è possibile, se non lo si è già, andare alla rotta "/kingdom";
        - tramite il pulsante "home" è possibile, se non lo si è già, andare alla rotta "/home";
        - tramite il pulsante "settings" è possibile, se non lo si è già, andare alla rotta "/settings". Tuttavia la view settings nel progetto in flutter non è stata implementata.
7. **StartStory:**

- **classe estesa:** StatefulWidget; viene eseguito l'override del metodo createState(), che utilizza la classe _StartStoryState per disegnare l'interfaccia.

8. **_StartStoryState:**
   - **classe estesa:** State; viene eseguito l'override dei metodi initState(), didChangeDependencies() e build();
   - **funzionalità:** la classe _StartStoryState racchiude le seguenti funzionalità:
     - recupera l'id dell'utente salvato nelle shared preferences;
     - recupera dal database tutti i nomi delle storie create dall'utente, che verranno salvato in una lista. Questa lista è utile per l'autocompletamento del nome della storia;
     - recupera dal database i valori utili per preimpostare lo slider lavoro-pausa;
     - recupera dal database i nomi degli avatar che poi verranno visualizzati nella sezione "scegli avatar" e i nomi dei background che verranno passati alla rotta "/story-started";
     - gestione delle azioni relative alle frecce che l'utente può usare per cambiare avatar;
     - impostazione delle modalità silenziosa e hardcore;
     - tramite il pulsante "indietro" è possibile tornare alla home;
     - tramite il pulsante "start" è possibile andare alla rotta "/story-started". A questa rotta verranno passati dei parametri sotto forma di istanza StoryStartedArguments.

9. **StoryStartedArguments:**
   - classe che racchiude i paramenti che vengono passati dalla rotta "/start-story" alla rotta "/story-started".
   - **Attributi:**
     - storyTitle: titolo della storia scelto dall'utente;
     - studyTime: tempo di studio selezionato dall'utente tramite lo slider;
     - breakTime: tempo di pausa generato tramite lo studyTime e il maxStudyTime(tempo massimo consentito dallo slider);
     - selectedAvatar: nome dell'avatar scelto dall'utente;
     - backgroundNames: nome dei background che verranno visualizzati randomicamente in StoryStarted;
     - mode: modalità scelta dall'utente. Può assumere i valori:
       - none: nessuna modalità;
       - silent: modalità silenziosa;
       - hardcore: modalità harcore.

10. StoryStarted:

- **classe estesa:** StatefulWidget; viene eseguito l'override del metodo createState(), che utilizza la classe _StoryStartedState per disegnare l'interfaccia.

11. _StoryStartedState:

- **classe estesa:** State; viene eseguito l'override dei metodi initState(), didChangeDependencies(), build() e dispose();
- **funzionalità:** la classe _StoryStartedState racchiude le seguenti funzionalità:
    - recupera l'id dell'utente salvato nelle shared preferences;
    - si occupa di mostrare a schermo adeguatamente il timer. Viene eseguito il rebuild dell'interfaccia ogni secondo;
    - si occupa della gestione del background, nonché della sua evoluzione. Sceglie un background casuale dalla lista passatagli da StoryStartedArguments;
    - gestisce la creazione di una nuova storia e di un nuovo capitolo o la creazione di un capitolo nel caso in cui la storia esistesse già nel database;
    - se l'utente sceglie la modalità hardcore, la classe si occupa del caso in cui l'utente uscisse dall'applicazione violando le regole della modalità stessa;
    - una volta conclusa la storia, l'utente viene reindirizzato alla home.

12. Kingdom:

- **classe estesa:** StatefulWidget; viene eseguito l'override del metodo createState(), che utilizza la classe _KingdomState per disegnare l'interfaccia.

13. _KingdomState:

- **classe estesa:** State; viene eseguito l'override dei metodi initState() e build();
- **funzionalità:** la classe _KingdomState racchiude le seguenti funzionalità:
    - recupera l'id dell'utente salvato nelle shared preferences;
    - recupera dal database tutte le storie e i capitoli in cui queste ultime si suddividono, mettendo il tutto in un Map con chiave i nomi delle storie e con valori liste di capitoli;

    - gestisce un widget di tipo ListView verticale, popolandolo con riquadri contenenti le informazioni delle storie;
    - tramite ogni riquadro è possibile visualizzare i dettagli di quella storia tramite la rotta "/story-detail". A questa rotta verranno passati dei parametri sotto forma di istanza StoryDetailArguments;
    - visualizza un istanza del widget personalizzato "WIPMenu".

14. StoryDetailArguments:

- classe che racchiude i paramenti che vengono passati dalla rotta "/kingdom" alla rotta "/story-detail".
- **Attributi:**
    - storyId: id di una storia. L'id è relativo all'attributo id della tabella "story" presente sul database;
    - storyName: nome che l'utente ha dato originariamente alla storia. Anche questo attributo fa riferimento alla tabella "story".

15. StoryDetail:

- **classe estesa:** StatefulWidget; viene eseguito l'override del metodo createState(), che utilizza la classe _StoryDetailState per disegnare l'interfaccia.

16. _StoryDetailState:

- **classe estesa:** State; viene eseguito l'override dei metodi initState(), didChangeDependencies() e build();
- **funzionalità:** la classe _StoryDetailState racchiude le seguenti funzionalità:
    - recupera i capitoli dal database tramite l'id della storia;
    - recupera dal database tutte le storie e i capitoli in cui quest'ultime si suddividono, mettendo il tutto in un Map con chiave i nomi delle storie e con valori liste di capitoli;
    - tramite il pulsante "indietro" è possibile tornare alla view kingdom;
    - gestisce un widget di tipo ListView verticale, popolandolo con riquadri contenenti le informazioni dei capitoli;
    - tramite ogni riquadro è possibile visualizzare i dettagli di quel capitolo tramite la rotta "/chapter-info". A questa rotta verranno passati dei parametri sotto forma di istanza ChapterInfoArguments;
    - visualizza un istanza del widget personalizzato "WIPMenu".

17. ChapterInfoArguments:

- classe che racchiude i paramenti che vengono passati dalla rotta "/story-detail" alla rotta "/chapter-info".
- **Attributi:**
    - createdOn: data di creazione del capitolo;
    - time: tempo in secondi di durata del capitolo;
    - studyTime: tempo di lavoro scelto dall'utente;
    - breakTime: tempo di pausa scelto dall'utente;
    - mode: modalità scelta dall'utente;
    - avatar: avatar selezionato dall'utente.

18. ChapterInfo:

- **classe estesa:** StatefulWidget; viene eseguito l'override del metodo createState(), che utilizza la classe _ChapterInfoState per disegnare l'interfaccia.

19. _ChapterInfoState:

- **classe estesa:** State; viene eseguito l'override dei metodi initState(), didChangeDependencies() e build();
- **funzionalità:** la classe _ChapterInfoState racchiude le seguenti funzionalità:
    - visualizza a schermo le informazioni del capitolo selezionato nella view story-detail tramite l'istanza passatagli del tipo ChapterInfoArguments;
    - tramite il pulsante "indietro" è possibile tornare alla view story-detail;
    - visualizza un istanza del widget personalizzato "WIPMenu".

Nelle view StartStory e StoryStarted vengono visualizzati dei popup contenenti nel primo caso, informazioni riguardanti le modalità che l'utente può scegliere nell'affrontare una nuova storia e nel secondo, informazioni sulla conclusione della storia. La storia può concludersi in due modi:

- l'utente sceglie di chiuderla mediante il pulsante stop.
- l'utente avendo precedentemente selezionato la modalità hardcore, esce dall'app mentre il timer scorre. Nel momento in cui decide di rientrare nell'applicazione, apparirà un popup che gli comunicherà di aver violato le regole della modalità.

20. WIPDialog:

- **classe estesa:** StatefulWidget; viene eseguito l'override del metodo createState(), che utilizza la classe _WIPDialogState per disegnare l'interfaccia.

21. _WIPDialogState:

- **classe estesa:** State; viene eseguito l'override dei metodi initState(), didChangeDependencies() e build();
- **funzionalità:** la classe _WIPDIalogState racchiude le seguenti funzionalità:
    - visualizza a schermo le informazioni tramite l'istanza del tipo WIPDialogArguments passata al costruttore della classe WIPDialog.

22. WIPDialogArguments:

- classe che racchiude i paramenti che vengono passati alla classe WIPDialog.
- **Attributi:**
    - children: lista di widget che verranno visualizzati nel popup;
    - height: altezza del popup,
    - popUntilRoot: attributo booleano che specifica se, una volta terminato il ciclo di vita del popup, l'applicazione debba tornare al punto di inizio(root). Nel nostro caso il punto di inizio è la rotta "/home".

Contributi

lavorare in team

L'app Work Is Progress è stata progettata in modo tale da ripartire, in egual misura, la responsabilità dei membri del team rispetto a ciascuna delle attività costitutive del processo software.
Di seguito, si illustrano i contributi di ciascuno dei partecipanti allo sviluppo dell'app:

Colleluori Federico	• User story • Glossario dei termini • Requisiti funzionali utente • Requisiti funzionali sistema • Use case • Documentazione degli use case • Dizionario dei dati • Mappa dell'architettura • Analisi dei moduli dell'architettura • Documentazione Android • Test • Codice: in generale, tutto il codice ma approfondimento verso la sezione Kingdom
Frisi Emanuele	• User story • Requisiti funzionali utente • Requisiti funzionali sistema • Use case • Documentazione degli use case • Implementazione del database • Mappa dell'architettura • Analisi dei moduli dell'architettura • Documentazione Android • Documentazione Flutter • Intent map • Codice: in generale, tutto il codice ma approfondimento verso la sezione Shop
Giusti Kevin	• User story • Divisione delle user story in tasks • UI • Requisiti funzionali utente • Requisiti funzionali sistema • Use case • Documentazione degli use case • Vincoli non esprimibili

	• Mappa dell'architettura • Analisi dei moduli dell'architettura • Documentazione Android • Intent map • Codice: in generale, tutto il codice ma approfondimento verso la sezione startStory

Strumenti utilizzati

IDE, librerie, risorse online, ecc…

Per sviluppare l'applicazione WIP ci si è avvalsi dei seguenti strumenti (per i quali è possibile effettuare, in caso di software, il download, oppure essere reindirizzati al sito, tramite CTRL + Clic sul nome stesso dello strumento):

1. Android Studio: IDE;
2. Dart: linguaggio di programmazione;
3. Git: software per il versioning del software;
4. Github: risorsa online per la gestione del repository;
5. Google Docs: editor online per la produzione della documentazione software;
6. Stack Overflow: risorsa online per soluzione dubbi relativi alla programmazione;
7. Discord: software per le comunicazioni in team;
8. Anydesk: software per condivisione del desktop tra pc remoti;
9. Pixilart: risorsa online per la produzione degli elementi in pixel art;
10. Draw.io: risorsa online per la produzione di diagrammi UML e schemi;
11. Photoshop: software per la modifica delle immagini.

FINE

Gli studenti:
Colleluori Federico, Frisi Emanuele, Giusti Kevin