Documentazione MentalMood

1 Introduzione 3

1.1 Informazioni sul progetto 3

1.2 Scopo 3

2 Analisi 4

2.1 Analisi del dominio 4

2.2 Flutter 4

2.3 Analisi e specifica dei requisiti 5

2.3.1 Spiegazione elementi tabella dei requisiti: 7

2.4 Use case 7

2.5 Pianificazione 8

2.6 Analisi dei mezzi 8

2.6.1 Software 8

2.6.2 Hardware 8

3 Progettazione 9

3.1 Design dei dati e database 9

3.1.1 Diagramma ER 9

3.2 Design delle interfacce 10

3.2.1 UI Bozza 10

3.2.2 UI finale 15

3.3 Design procedurale 16

4 Implementazione 17

5 Test 17

5.1 Protocollo di test 17

5.2 Risultati test 18

5.3 Mancanze/limitazioni conosciute 18

6 Consuntivo 18

7 Conclusioni 18

7.1 Sviluppi futuri 18

7.2 Considerazioni personali 18

8 Glossario 18

9 Bibliografia 19

9.1 Bibliografia per articoli di riviste: 19

9.2 Bibliografia per libri 19

9.3 Sitografia 19

10 Allegati 19

# Introduzione

## Informazioni sul progetto

Bryan Ciaponi, i3AC.

SAMT Trevano, sezione informatica, Modulo 306.

Periodo progetto:12.09.2025 – 19.12.2025

## Scopo

Il progetto ha l’obiettivo di sviluppare un’applicazione che permetta agli utenti di monitorare il proprio stato emotivo in maniera semplice e intuitiva. Dal punto di vista operativo, l’app offrirà strumenti per registrare l’umore quotidiano tramite emoji e brevi annotazioni, visualizzare statistiche settimanali e mensili e identificare eventuali pattern emotivi nel tempo. Dal punto di vista didattico, il progetto permette di applicare competenze di sviluppo multi-piattaforma, promuovendo la comprensione di come dati personali e possano essere organizzati e rappresentati efficacemente.

# Analisi

## Analisi del dominio

La gestione e il monitoraggio del proprio stato emotivo è spesso frammentaria e poco strutturata. Gli utenti normalmente tengono traccia dei propri sentimenti in modo informale, ad esempio tramite appunti, diario cartaceo o senza alcuna registrazione. La mancanza di strumenti semplici e intuitivi rende difficile riconoscere tendenze o variazioni emotive nel tempo.

L’applicazione sarà utilizzata principalmente su dispositivi mobili (smartphone o tablet), in contesti personali e quotidiani. Gli utenti possono accedere all’app in qualsiasi momento della giornata per registrare rapidamente il loro stato d’animo.

Esistono già applicazioni dedicate al benessere mentale e al monitoraggio dell’umore, alcune molto complesse e orientate a psicologi o terapisti, altre più semplici ma limitate nelle funzionalità di analisi dei dati. Il mio prodotto si propone di offrire una soluzione semplice, immediata e focalizzata sull’auto-riflessione quotidiana.

Gli utenti principali sono persone interessate a capire e migliorare il proprio stato emotivo, con conoscenze di base nell’uso di dispositivi mobili e applicazioni. Hanno bisogno di strumenti facili da usare per registrare emozioni, osservare trend nel tempo e ricevere spunti di consapevolezza sul proprio benessere.

Gli utenti non necessitano di conoscenze tecniche particolari: è sufficiente familiarità con smartphone e interfacce touch. La comprensione dei dati visualizzati (grafici e statistiche semplici) richiede competenze di base in lettura di grafici.

L’applicazione seguirà convenzioni comuni di interfaccia mobile, come l’uso di emoji standard per rappresentare emozioni, icone intuitive e layout responsive, garantendo facilità d’uso e accessibilità.

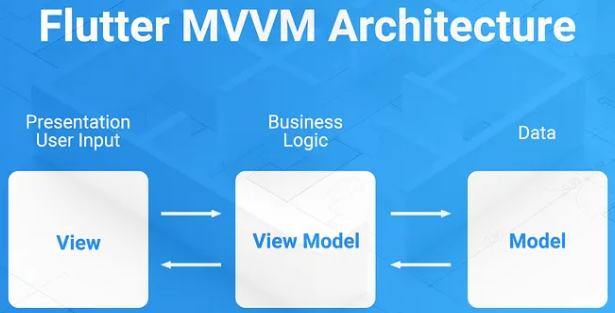
Per sviluppare e far funzionare efficacemente l’app, è necessario conoscere principi di programmazione mobile, gestione di dati temporali, interfacce utente intuitive e rappresentazione grafica dei dati.

Inoltre per rendere compatibile la applicazione su vari sistemi operativi ho deciso di programmarla con Flutter.

## Flutter

Flutter è un framework open source sviluppato da Google che permette di realizzare applicazioni multipiattaforma partendo da una sola codebase. Utilizza come linguaggio Dart, che ha una sintassi moderna e simile a Java o C#, ed è ottimizzato sia per lo sviluppo rapido, grazie al supporto all’hot reload, sia per la compilazione in codice nativo ad alte prestazioni. Con Flutter è possibile sviluppare app per Android, iOS, Windows, macOS, Linux e persino per il web, mantenendo coerenza grafica e funzionale su tutte le piattaforme.

Alla base del framework c’è il concetto di widget: ogni elemento dell’interfaccia, dal testo ai layout complessi, è un widget che può essere combinato in una “widget tree”. Questa struttura rende lo sviluppo modulare e intuitivo, mentre il motore grafico Skia garantisce uniformità visiva e performance vicine a quelle delle app native.

Dal punto di vista architetturale, è comune adottare il pattern MVVM (Model-View-ViewModel). Il Model rappresenta i dati, la View è l’interfaccia grafica composta dai widget, mentre la ViewModel funge da ponte, gestendo la logica di presentazione e notificando la View dei cambiamenti. In Flutter questo è spesso implementato con ChangeNotifier, Provider o librerie come Riverpod e Bloc, che permettono di mantenere il codice più ordinato e testabile.

I vantaggi di Flutter sono numerosi: sviluppo rapido grazie all’hot reload, una sola codebase per più piattaforme, prestazioni elevate, interfacce facilmente personalizzabili e animate, un ampio ecosistema di pacchetti e strumenti di debugging e testing avanzati.

Ho scelto Flutter per questo progetto perché consente di ridurre i tempi di sviluppo, mantenere un design coerente tra le piattaforme, sfruttare al meglio le animazioni e le UI moderne, oltre a semplificare la manutenzione futura con una sola base di codice.

## Analisi e specifica dei requisiti

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-01 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Memorizzazione dei dati | | | | |
| **Note** | I dati dell’applicativo sono registrati in un DB locale. | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-02 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Registrazione stato emotivo | | | | |
| **Note** | L’utente ogni giorno potrà selezionare il suo stato emotivo attuale. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | L’utente potrà anche selezionare facoltativamente dei campi pre-impostati che gli permetteranno di descrivere al meglio le circostanze al momento della selezione. | | | | |
| **002** | I dati dell’utente riguardo il suo umore, con l’ausilio del requisito Req-01, verranno salvati localmente nel suo dispositivo. | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-03 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Storico dei dati | | | | |
| **Note** | L’utente sarà in grado di vedere le sue selezioni i giorni precedenti, con tutte le informazioni. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Richiede il completo funzionamento dei requisiti: Req-01 e Req-02. | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-04 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Statistiche | | | | |
| **Note** | L’utente sarà in grado di visualizzare dei grafici e delle statistiche che gli mostrano il suo progresso nello stato d’animo. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Richiede il completo funzionamento del requisito Req-03. | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-05 | **Priorità** | 2 | **Versione** | 1.0 |
| **Nome** | Selezione periodo statistiche | | | | |
| **Note** | L’utente sarà in grado di scegliere il periodo di visualizzazione dei grafici e delle statistiche | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Richiede il completo funzionamento del requisito: Req-04. | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-06 | **Priorità** | 2 | **Versione** | 1.0 |
| **Nome** | Riconoscimento tendenze | | | | |
| **Note** | L’utente verrà avvisato nella applicazione se si notano delle tendenze, per poter aumentare l’efficienza del suo stato emotivo. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Richiede il completo funzionamento del requisito: Req-04. | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-07 | **Priorità** | 2 | **Versione** | 1.0 |
| **Nome** | Consigli per l’utente | | | | |
| **Note** | L’utente, in base al suo stato d’animo, vedrà dei consigli che gli permettono di migliorare il suo stato d’animo/ mantenere il suo stato d’animo stabile. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Richiede il completo funzionamento del requisito: Req-02. | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-08 | **Priorità** | 2 | **Versione** | 1.0 |
| **Nome** | Notifiche | | | | |
| **Note** | All’utente verranno inviate delle notifiche, che gli ricorderanno di fare la sua selezione giornagliera/ chiederanno se lo stato d’animo è rimasto lo stesso. | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-09 | **Priorità** | 2 | **Versione** | 1.0 |
| **Nome** | Gestione notifiche | | | | |
| **Note** | L’utente sarà in grado di decidere con che frequenza ricevere notifiche. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Richiede il completo funzionamento del requisito: Req-08. | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-10 | **Priorità** | 2 | **Versione** | 1.0 |
| **Nome** | Rimozione dati | | | | |
| **Note** | L’utente potrà decidere di eliminare i suoi dati anteriori di una X data. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Richiede il completo funzionamento del requisito: Req-03. | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | Req-11 | **Priorità** | 2 | **Versione** | 1.0 |
| **Nome** | Esportazioni grafici | | | | |
| **Note** | L’utente sarà in grado di esportare i grafici | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Richiede il completo funzionamento del requisito: Req-05. | | | | |

### Spiegazione elementi tabella dei requisiti:

**ID**: identificativo univoco del requisito

**Nome**: breve descrizione del requisito

**Priorità**: indica l’importanza di un requisito nell’insieme del progetto, definita assieme al committente. Ad esempio, poter disporre di report con colonne di colori diversi ha priorità minore rispetto al fatto di avere un database con gli elementi al suo interno. Solitamente si definiscono al massimo di 2-3 livelli di priorità.

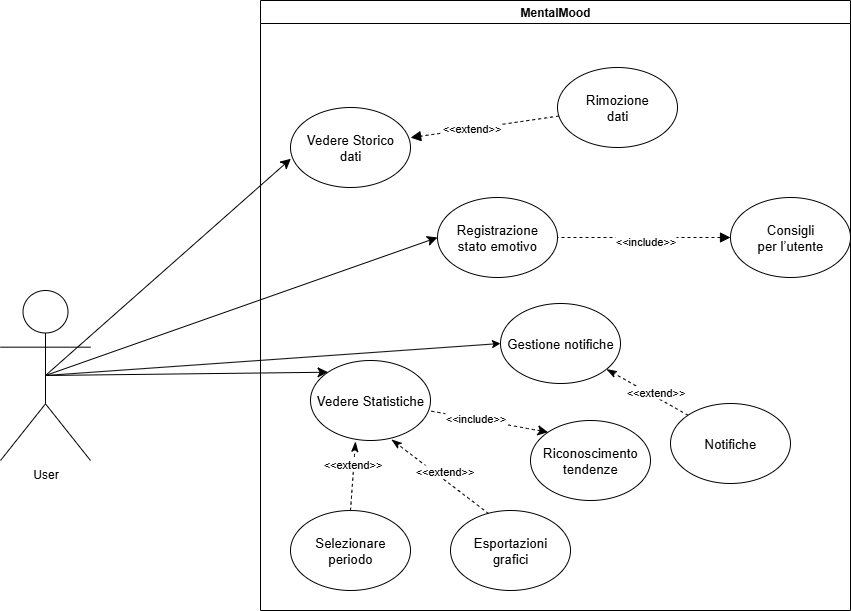
**Versione**: indica la versione del requisito. Ogni modifica del requisito avrà una versione aggiornata.

Sulla documentazione apparirà solamente l’ultima versione, mentre le vecchie dovranno essere inserite nei diari.

**Note**: eventuali osservazioni importanti o riferimenti ad altri requisiti.

**Sotto requisiti**: elementi che compongono il requisito.

## Use case



Questo è lo Use Case che descrive il funzionamento della mia applicazione MentalMood. Come si può vedere l'utente può registrare il proprio stato emotivo e visualizzare lo storico dei dati inseriti.

Inoltre può visualizzare le statistiche dei propri stati d'animo e gestire le notifiche dell'applicazione. Attraverso la visualizzazione delle statistiche, il sistema riconosce automaticamente le tendenze emotive dell'utente.

Dallo storico dati e dalle statistiche, l'utente può selezionare periodi specifici e esportare i grafici dei propri stati emotivi. In questo modo l'utente può monitorare e analizzare completamente la propria salute emotiva nel tempo.

Infine l'utente può ricevere notifiche personalizzate che lo aiutano a mantenere costante la registrazione del proprio stato d'animo.

## Pianificazione

Prima di stabilire una pianificazione bisogna avere almeno una vaga idea del modello di sviluppo che si intende adottare. In questa sezione bisognerà inserire il modello concettuale di sviluppo che si seguirà durante il progetto. Gli elementi di riferimento per una buona pianificazione derivano da una scomposizione top-down della problematica del progetto.

La pianificazione può essere rappresentata mediante un diagramma di Gantt:

|  |
| --- |
|  |

Se si usano altri metodi di pianificazione (p.es. scrum), dovranno apparire in questo capitolo.

## Analisi dei mezzi

Per la realizzazione del progetto è stato utilizzato un PC scolastico con Android Studio e Flutter. Inoltre per il debugging è stato usato un tablet scolastico.

### Software

Dart SDK version: 3.9.3

Flutter version: 3.35.3

Android SDK version: 15.0 («VanillaIceCream»)

Android Studio version: 2024.3.2 Patch 1

### Hardware

Questa applicazione dovrà risultare eseguibile su tutti i dispositivi più moderni.

Per la realizzazione di questo progetto ho utilizzato un pc scolastico con Windows 11 installato.

**HW computer Windows**

CPU: 13th Gen Intel® Core™ i7-13700

GPU: NVIDIA T400 4GB

RAM: 32GB

**HW Tablet con Android 11**

Modello: Samsung Galaxy Tab A 10.1

RAM: 2GB

CPU: Exynos 7904

# Progettazione

## Design dei dati e database

Ho implementato il database del mio progetto Flutter utilizzando Drift (versione ^2.18.0), un framework scelto per la sua eccellente integrazione con Dart che garantisce la sicurezza dei tipi e genera automaticamente il codice per la gestione di query e tabelle. Per assicurare la compatibilità e l'accesso alle librerie SQLite native, ho incluso la dipendenza sqlite3\_flutter\_libs (versione ^0.5.40). Questo approccio ha reso la gestione dei dati pulita, moderna e performante.

### Diagramma ER

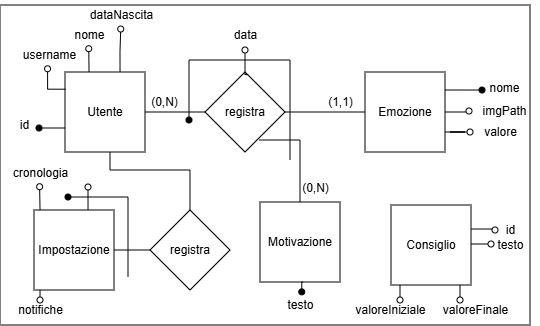


Figura 1: Schema ER DataBase Applicativo

Il database è strutturato in cinque entità principali che verranno mappate in tabelle.  
La prima entità è Consiglio, dedicata alla memorizzazione dei consigli che vengono mostrati all’utente, i quali sono generati in base al range di valori (valoreIniziale e valoreFinale) dell’umore registrato.  
L'entità centrale è Utente, che contiene i dati personali come id, username, nome (nome completo), e dataNascita. L'Utente è collegato all'entità Impostazione, che registra le preferenze dell'utente relative alla cronologia.  
L'entità Utente è collegata a una complessa relazione registra che coinvolge le entità Emozione e Motivazione. L'entità Emozione registra l'umore selezionato dall'utente in una data specifica (attributo data sulla relazione), con dettagli come nome, valore e imgPath. L'entità Motivazione registra il testo che l'utente associa a quella specifica emozione in quella data. Grazie a questa complessa relazione, è possibile creare uno storico delle emozioni e delle motivazioni per ogni utente, permettendo di consultare in seguito i dati passati e analizzare l'evoluzione dello stato d'animo nel tempo.

## Design delle interfacce

La progettazione delle interfacce è stata sviluppata a partire dalle informazioni raccolte nella fase di analisi ed è stata realizzata tramite la piattaforma Figma.com. Tutte le schermate sono state progettate pensando principalmente all’utilizzo su dispositivi mobili, con particolare attenzione alla fruizione da smartphone.

### UI Bozza

Le interfacce presentate di seguito rappresentano delle prime bozze preliminari dell’applicazione, create nella fase iniziale del progetto. Questi prototipi sono stati sviluppati sulla base delle aspettative e degli obiettivi individuati durante l’analisi, con lo scopo di definire l’aspetto grafico generale e l’esperienza utente desiderata. Non si tratta quindi di versioni definitive, ma di linee guida utili per valutare la disposizione degli elementi, la navigazione tra le schermate e la coerenza visiva complessiva, così da poter apportare eventuali modifiche prima della realizzazione effettiva.

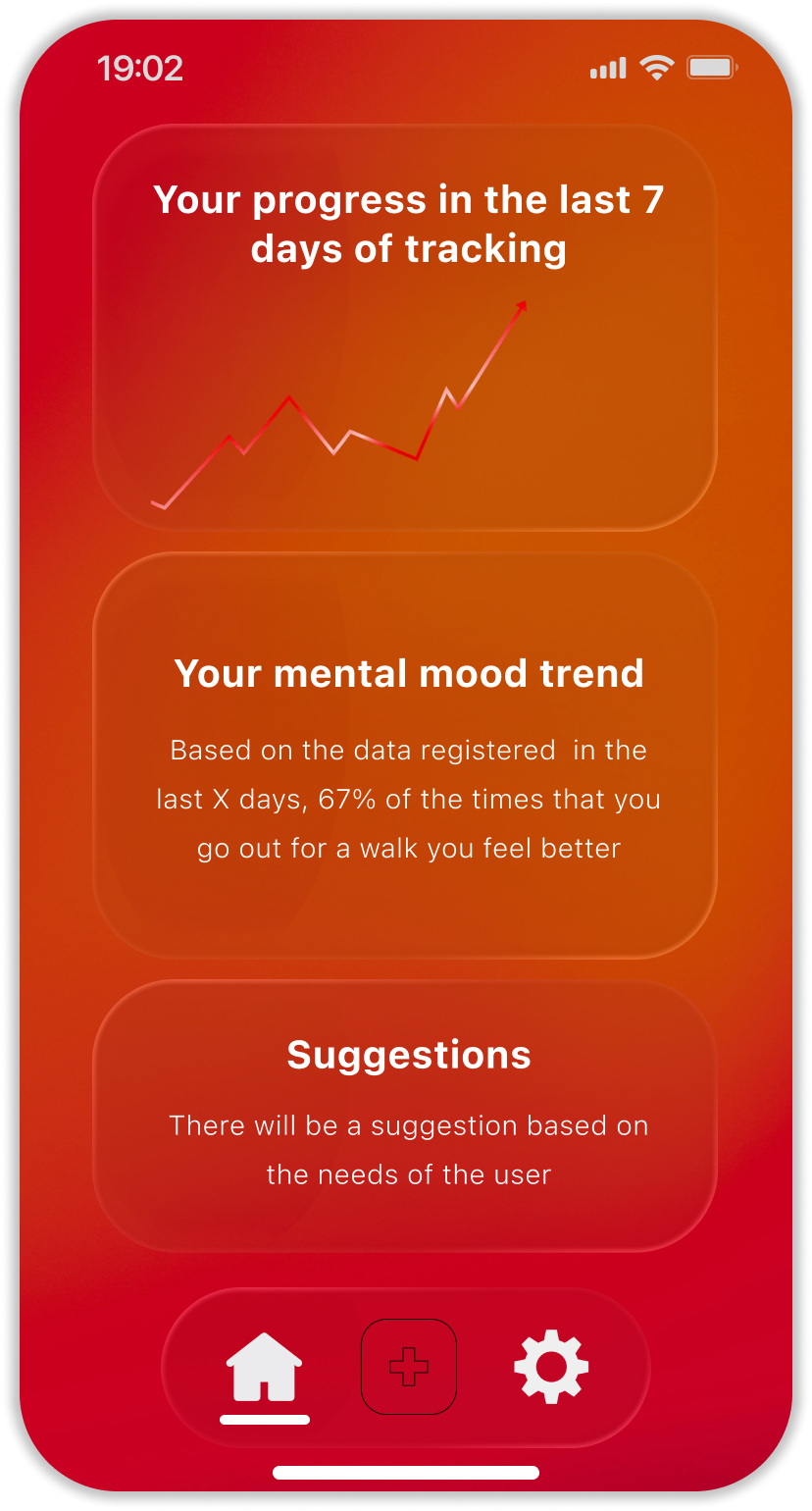


Figura 2: Mockup iniziale HomePage



Figura 3: Mockup iniziale EmotionSelectionPage

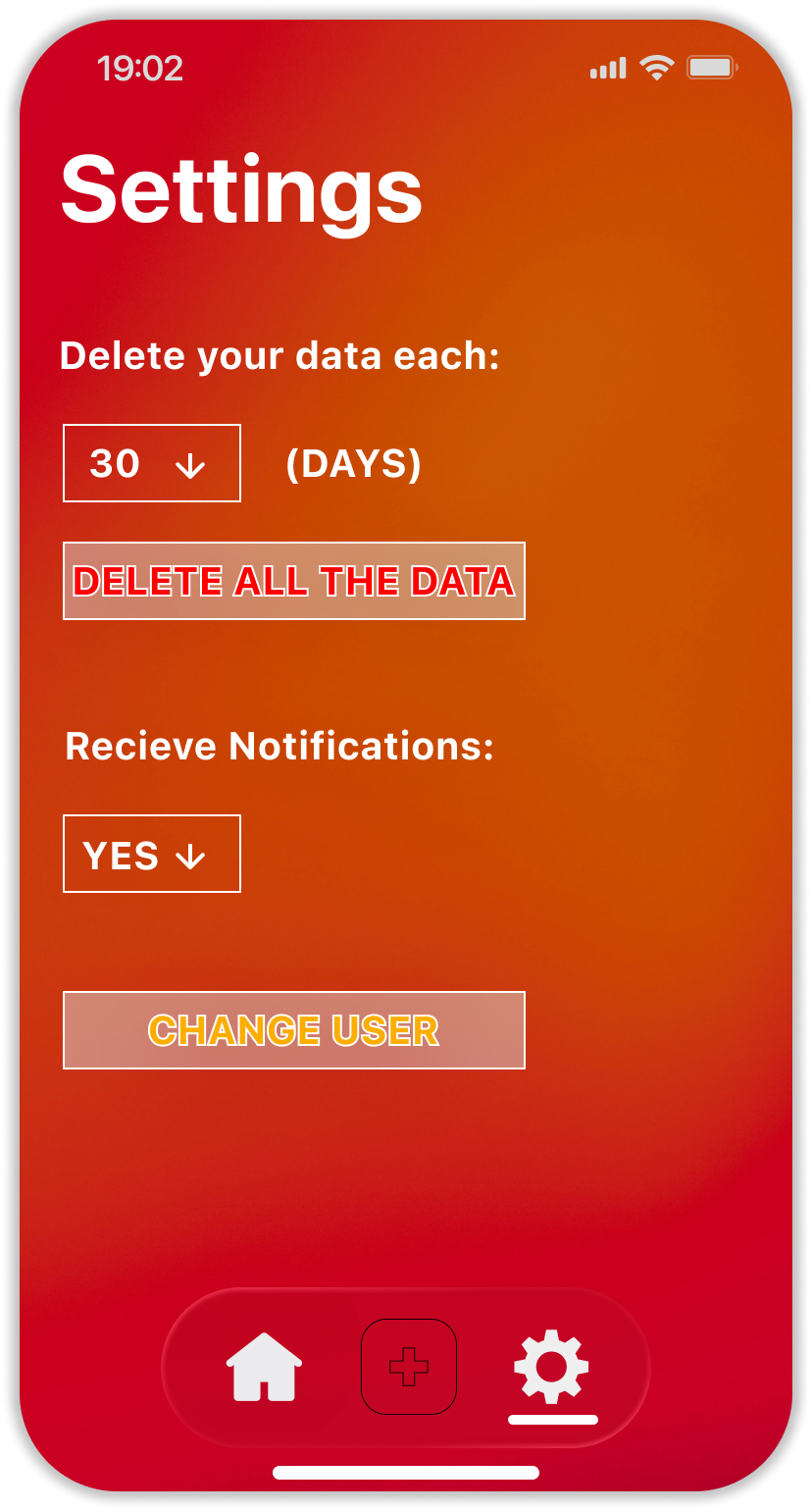


Figura 4: Mockup iniziale UserSettingsPage

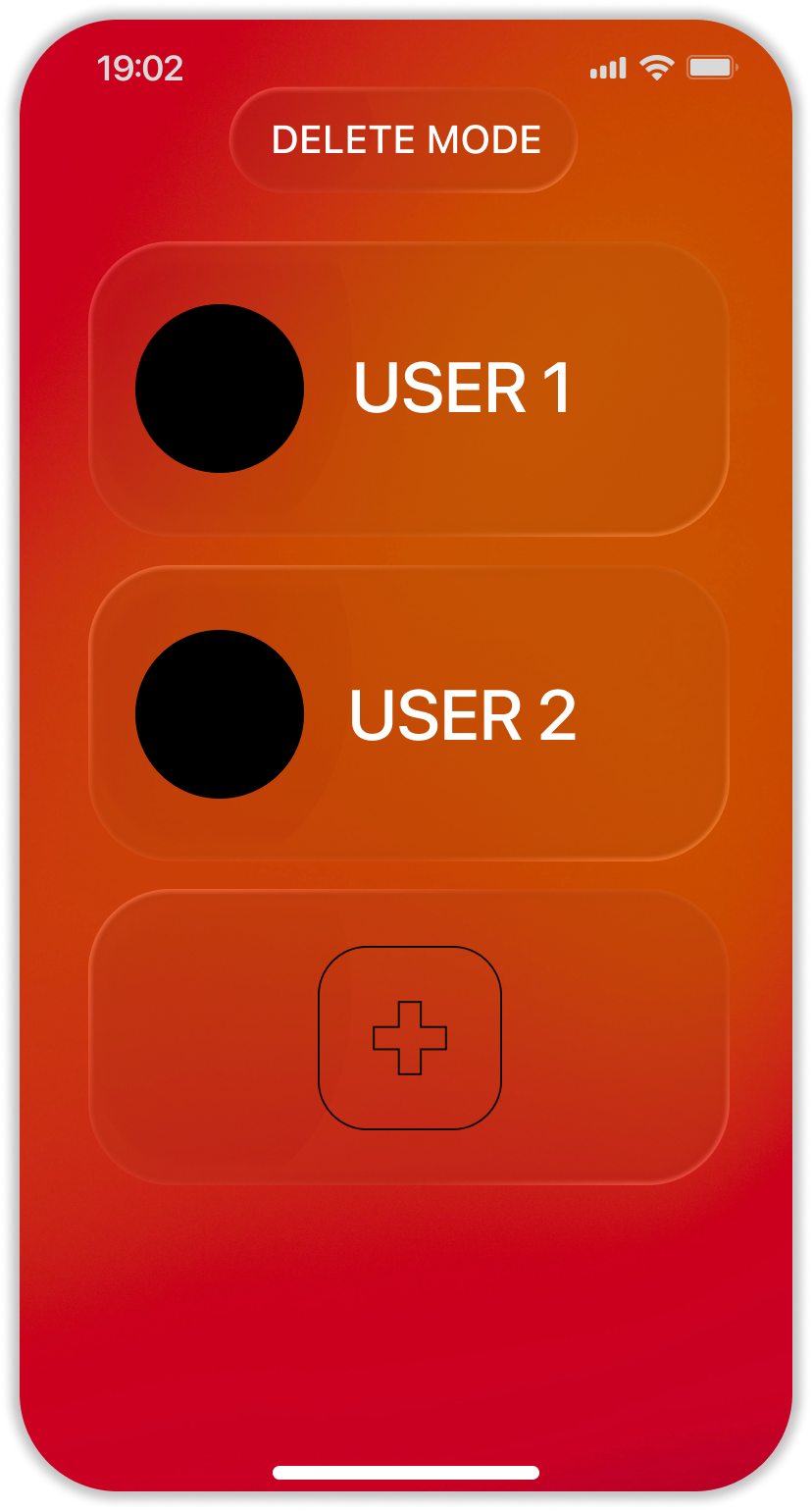


Figura 5: Mockup iniziale UserSelectionPage

### UI finale

Le interfacce presentate di seguito rappresentano la UI effettiva dell’applicazione, così come è stata realizzata nella sua versione finale. Esse non sono più bozze o prototipi, ma le schermate definitive che compongono l’app, sviluppate sulla base delle scelte grafiche e funzionali stabilite durante il progetto. Questa UI riflette l’aspetto reale dell’applicazione e mostra in modo concreto come l’utente interagisce con i vari elementi e le funzionalità previste. Esso segue le orme del mockup visualizzato precedentemente.

Inoltre dovuta alla limitazione di fare il debugging in un tablet si vedra il formato essere diverso dal previsto.

DA METTERE IMMAGINI QUANDO FINISCE IL PROGETTO

## Design procedurale

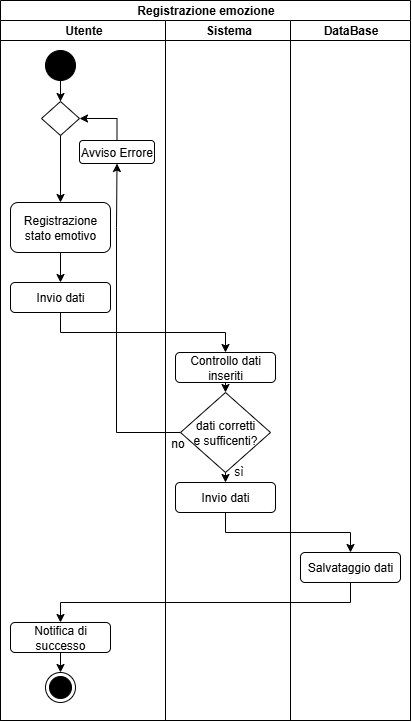


Figura 6: Swimlane della fase di registrazione nell'applicativo

# Implementazione

In questo capitolo dovrà essere mostrato come è stato realizzato il lavoro. Questa parte può differenziarsi dalla progettazione in quanto il risultato ottenuto non per forza può essere come era stato progettato.

Sulla base di queste informazioni il lavoro svolto dovrà essere riproducibile.

In questa parte è richiesto l’inserimento di codice sorgente - Print Screen - di maschere solamente per quei passaggi particolarmente significativi e/o critici.

Inoltre, dovranno essere descritte eventuali varianti di soluzione o scelte di prodotti con motivazione delle scelte.

Non deve apparire nessuna forma di guida d’uso di librerie o di componenti utilizzati. Eventualmente questa va allegata.

Per eventuali dettagli si possono inserire riferimenti ai diari.

# Test

## Protocollo di test

Definire in modo accurato tutti i test che devono essere realizzati per garantire l’adempimento delle richieste formulate nei requisiti. I test fungono da garanzia di qualità del prodotto. Ogni test deve essere ripetibile alle stesse condizioni.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-001  REQ-012 | **Nome:** | Import a card with KIC, KID and KIK keys, but not shown with the GUI |
| **Descrizione:** | Import a card with KIC, KID and KIK keys with no obfuscation, but not shown with the GUI | | |
| **Prerequisiti:** | Store on local PC: Profile\_1.2.001.xml (appendix n\_n) and Cards\_1.2.001.txt (appendix n\_n).  PIN (OTA\_VIEW\_PIN\_PUK\_KEY) and ADM (OTA\_VIEW\_ADM\_KEY) user right not set. | | |
| **Procedura:** | 1. Go to “Cards manager” menu,  in main page click “Import Profiles” link, Select the “1.2.001.xml” file, Import the Profile 2. Go to “Cards manager” menu,  in main page click “Import Cards” link, Select the “1.2.001.txt” file, Delete the cards,  Select the “1.2.001.txt” file, Import the cards 3. Research the “41795924770” Card, Click the imsi card link Check the card details 4. Execute the SQL: SELECT imsi, dir, keyset, cntr, rawtohex(kickey), rawtohex(kidkey), rawtohex(kikkey), rawtohex(chv), rawtohex(dap)FROM otacardkey a where imsi='340041795924770' ORDER BY keyset; | | |
| **Risultati attesi:** | Keys visible in the DB (OtaCardKey) but not visible in the GUI (Card details) | | |

## Risultati test

Tabella riassuntiva in cui si inseriscono i test riusciti e non del prodotto finale. Se un test non riesce e viene corretto l’errore, questo dovrà risultare nel documento finale come riuscito (la procedura della correzione apparirà nel diario), altrimenti dovrà essere descritto l’errore con eventuali ipotesi di correzione.

## Mancanze/limitazioni conosciute

Descrizione con motivazione di eventuali elementi mancanti o non completamente implementati, al di fuori dei test case. Non devono essere riportati gli errori e i problemi riscontrati e poi risolti durante il progetto.

# Consuntivo

Consuntivo del tempo di lavoro effettivo e considerazioni riguardo le differenze rispetto alla pianificazione (cap. 1.7) (ad esempio Gantt consuntivo).

# Conclusioni

Quali sono le implicazioni della mia soluzione? Che impatto avrà? Cambierà il mondo? È un successo importante? È solo un’aggiunta marginale o è semplicemente servita per scoprire che questo percorso è stato una perdita di tempo? I risultati ottenuti sono generali, facilmente generalizzabili o sono specifici di un caso particolare? ecc.

## Sviluppi futuri

Migliorie o estensioni che possono essere sviluppate sul prodotto.

* **(Widget per la home**: un piccolo pulsante/emoji da toccare subito senza aprire l’app (per registrare più velocemente l’umore).)
* (Se possibile con il tempo, una implementazione di IA per dare consigli migliori o per analizzare i dati nel modo più efficente.)
* Aggiunta grafici e statistiche
* Aggiunta notifiche

## Considerazioni personali

Cosa ho imparato in questo progetto? ecc.

# Glossario

Inserite una semplice tabella con due colonne che spieghi i termini specifici del progetto (lista dei termini in ordine alfabetico A-Z)

Esempio:

|  |  |
| --- | --- |
| **Termine** | **Descrizione** |
| AJAX | **Asynchronous JavaScript And XML**: una tecnica che permette di eseguire richieste ed ottenere dati da una pagina web in modo asincrono. |
| CSS | **Cascading Style Sheets**: linguaggio che permette di definire il layout e la grafica di una pagina web. |

# Bibliografia

## Bibliografia per articoli di riviste:

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo dell’articolo (tra virgolette),
3. Titolo della rivista (in italico),
4. Anno e numero
5. Pagina iniziale dell’articolo.

[Figura 1: Schema ER DataBase Applicativo 9](#_Toc216187254)

[Figura 2: Mockup iniziale HomePage 11](#_Toc216187255)

[Figura 3: Mockup iniziale EmotionSelectionPage 12](#_Toc216187256)

[Figura 4: Mockup iniziale UserSettingsPage 13](#_Toc216187257)

[Figura 5: Mockup iniziale UserSelectionPage 14](#_Toc216187258)

[Figura 6: Swimlane della fase di registrazione nell'applicativo 16](#_Toc216187259)

## Bibliografia per libri

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo del libro (in italico),
3. ev. Numero di edizione,
4. Nome dell’editore,
5. Anno di pubblicazione,
6. ISBN.

## Sitografia

1. URL del sito (se troppo lungo solo dominio, evt completo nel diario),
2. Eventuale titolo della pagina (in italico),
3. Data di consultazione (GG-MM-AAAA).

**Esempio:**

* http://standards.ieee.org/guides/style/section7.html, *IEEE Standards Style Manual*, 07-06-2008.

# Allegati

Elenco degli allegati, esempio:

* Diari di lavoro
* Codici sorgente/documentazione macchine virtuali
* Istruzioni di installazione del prodotto (con credenziali di accesso) e/o di eventuali prodotti terzi
* Documentazione di prodotti di terzi
* Eventuali guide utente / Manuali di utilizzo
* Mandato e/o QdC
* Prodotto
* …