Laurea Triennale in informatica-Università di Salerno
Corso di *Ingegneria del Software* - Prof C. Gravino

System Design Document

OutfitMaker

Riferimento	
Versione	1.0
Data	1/12/2023
Destinatario	Studenti di Ingegneria del Software 2023/24
Presentato da	Avallone Francesca Buongiorno Rocco Pio Letterese Michele
Approvato da	

Revision History

DATA	VERSIONE	DESCRIZIONE	AUTORI
23/11/2023	0.1	Prima stesura	Francesca Avallone Rocco Pio Buongiorno Michele Letterese
24/11/2023	0.2	Aggiunta design goal	Francesca Avallone Rocco Pio Buongiorno Michele Letterese
26/10/2023	0.3	Aggiunta trade-off	Francesca Avallone Rocco Pio Buongiorno Michele Letterese
28/11/2023	0.4	Aggiunta scelta "Architettura del sistema"	Francesca Avallone Rocco Pio Buongiorno Michele Letterese
29/11/2023	0.5	Aggiunta "Decomposizione in sottosistemi"e Component Diagram UML	Francesca Avallone Rocco Pio Buongiorno Michele Letterese
30/11/2023	0.6	Aggiunta Diagramma Architetturale	Francesca Avallone Rocco Pio Buongiorno Michele Letterese
1/12/2023	0.7	Aggiunta Mapping Hardware/Software, UML Deployment Diagram	Francesca Avallone Rocco Pio Buongiorno Michele Letterese
2/12/2023	0.8	Aggiunta Gestione dei dati persistenti, Modello logico e vincoli di integrità referenziale	Francesca Avallone Rocco Pio Buongiorno Michele Letterese
3/12/2023	0.9	Aggiunta Controllo degli accessi e sicurezza e Controllo del flusso globale	Francesca Avallone Rocco Pio Buongiorno Michele Letterese
4/12/2023	1.0	Aggiunta Condition Boundary	Francesca Avallone Rocco Pio Buongiorno Michele Letterese
5/12/2023	1.1	Aggiunta Servizi dei sottosistemi	Francesca Avallone Rocco Pio Buongiorno Michele Letterese
6/12/2023	1.2	Revisione e Glossario	Francesca Avallone Rocco Pio Buongiorno Michele Letterese
7/12/2023	1.3	Revisione finale documento	Francesca Avallone Rocco Pio Buongiorno Michele Letterese

Sommario

Revision History 2
1. Introduzione 4
1.1 Scopo del sistema 4
1.2 Design Goals 4
1.3 Trade-offs 7
1.4 Definizioni acronimi e abbreviazioni 7
1.5 Riferimenti 8
1.6 Panoramica 8
2. Architettura del sistema corrente 9
3. Architettura del sistema proposto 9
3.1 Panoramica 10
3.2 Decomposizione in sottosistema 10
3.3 Mapping Hardware/Software 13
3.4 Gestione dei dati persistenti 14
3.4.1 Modello Concettuale Ristrutturato 15
3.4.2 Modello logico 16
3.5 Controllo degli accessi e sicurezza 17
3.6 Controllo del flusso globale del sistema 18
3.7 Condizioni Boundary 19
4. Servizi dei sottosistemi 20

1.Introduzione

1.1 Scopo del sistema

Lo scopo principale del sistema è quello di fornire agli utenti dei consigli di abbigliamento personalizzati in base alle loro preferenze e al contesto. Adattare le raccomandazioni di outfit in base al gusto personale dell'utente considerando tipologie preferiti, colori preferiti, e altre preferenze individuali. Questo, tenendo conto dei capi di abbigliamento già presenti nell'armadio virtuale personale dell'utente. Creando o generando degli outfit con essi, ad hoc per le preferenze richieste dall'utente avendo anche la possibilità di visionare anche dopo gli outfit creati o generati nel proprio archivio outfit. Tutto questo comodamente nella tasca dell'utente.

1.2 Design Goals

In questo paragrafo vengono esposti i Design Goals ("Obiettivi di Design"), che descrivono le qualità chiave del sistema (quelle che devono essere ottimizzate) e stabiliscono le loro priorità. Sviluppando in modo corretto i Design Goal sarà possibile stabilire un percorso di sviluppo ben delineato, solido e dettagliato.
Nella tabella che segue, i design goal sono descritti in base ai seguenti campi, ognuno corrispondente a una colonna:

• Rank: ne indica la priorità rispetto agli altri design goal;
• ID: un identificatore;
• Descrizione: ne descrive le caratteristiche;
• Categoria: raggruppa i design goal che descrivono qualità che rientrano nella stessa macroarea;
• RNF di origine: il requisito non funzionale da cui è stato generato;

In particolare, le categorie in cui sono stati suddivisi i design goal sono:

• Performance: indica, in maniera quantificabile, il livello di prestazioni del sistema software;

• Dependability: relativo all'affidabilità del sistema (gestione degli errori, resistenza ai crash…);

• Cost: indice dell'impatto economico dell'implementazione di una qualità sui costi di sviluppo;

• Maintenance: indica quanto l'effort è necessario per apportare modifiche al sistema dopo la prima release; • End user: relativo ai canoni dell'interazione uomo-macchina e di ingente importanza per valutare la compatibilità di un sistema software con l'utenza.

Priorità	ID	Descrizione	Categoria	RNF di origine
3	**DG_1 Tempi di risposta**	I tempi di risposta non devono superare i 5 secondi per qualsiasi operazione dell'utente	Performance	**RNF[2]**
8	**DG_2 Costo di sviluppo**	Il costo complessivo del progetto ammonta ad un massimo di 150 ore (max 50 ore per ogni membro del gruppo).	Cost	**RNF[6]**
1	**DG_3 Disponibilità**	L'app sarà disponibile 24/7, ed eventuali manutenzioni o aggiornamenti verranno comunicati in anticipo agli utenti	Dependability	**RNF[3]**
4	**DG_4 Robustezza**	Il sistema deve gestire input utente non validi, rilevando e comunicando errori attraverso messaggi chiari. Deve consentire agli utenti di correggere e ricompilare il modulo per un'interazione senza intoppi.	Dependability	**RNF[1]**
5	**DG_5 Sicurezza dati sensibili**	Il sistema deve permettere e tutelare l'accesso ad informazioni sensibili ai soli utenti autorizzati. Per assicurare la sicurezza dei dati facciamo utilizzo di password.	Dependability	**RNF[1]**
2	**DG_6 User Friendly**	Il sistema deve essere di facile comprensione all'utente, con una descrizione delle operazioni da eseguire, per consentire un utilizzo rapido.	End user	**RNF[4]**
6	**DG_7 Portabilità**	Il sistema deve garantire il suo utilizzo su almeno il 95% dei dispositivi mobile e tablet con un sistema operativo Android.	Maintenance	**RNF[5]**
7	**DG_8 Memoria**	La quantità di memoria occupata dal Sistema dipende da quella necessaria al mantenimento dei dati persistenti del database.	Performance	**RNF[6]**

1.3 Trade-offs

Trade-off	Descrizione
Costo vs velocità	Si è deciso di dare più importanza ai tempi di consegna, nonostante questo la qualità del software rimarrà entro determinati standard.
Portabilità vs velocità	Assicurare la compatibilità sul 95% dei dispositivi Android, anche se alcune operazioni potrebbero richiedere più tempo o risorse per adattarsi a diversi dispositivi.
Usabilità vs memoria	Poiché uno degli obiettivi del sistema è l'alta usabilità per gli utenti finali, è necessario memorizzare una grande porzione di dati e ciò porta ad un elevato uso della memoria del database.

1.4 Definizioni acronimi e abbreviazioni

1. Definizioni:
   - **Sottosistema:** Corrisponde alla parte di lavoro che può essere svolta autonomamente da un singolo sviluppatore o da un gruppo di sviluppatori. Si ottiene decomponendo il sistema.
   - **Design Goal:** Proprietà del sistema sulla quale ci si concentra maggiormente.
   - **Trade-off:** Possibilità di ridurre una certa qualità per aumentare il valore di un'altra qualità e viceversa. Il termine è espresso talvolta come costo opportunità, riferendosi a più alternative alle quali si è preferito rinunciare a vantaggio di un'altra scelta.
   - **Mapping Hardware-Software:** Descritto per indicare i vari device hardware utilizzati dal sistema e la loro interazione con le componenti software.
   - **Dati Persistenti:** Dati che devono sopravvivere all'esecuzione singola dell'applicazione. Sono i dati che vengono salvati nel database.

Acronimi:

- **RAD** = Requirement Analysis Document.
- **SDD** = System Design Document.
- **VIR[n]** = **Vincoli d'integrità referenziale.**

1.5 Riferimenti

Per stilare la presente documentazione, si è preso come riferimento le slide fornite dal Docente del corso di Ingegneria del Software, Carmine Gravino, inserite nella sezione "M3" della piattaforma di e-learning della facoltà di Informatica. Inoltre, è stato consultato il libro di testo "Object-Oriented Software Engineering Using UML, Patterns and Java: Third Edition, di Bernd Bruegge ed Allen H. Dutoit".

1.6 Panoramica

Dopo questa prima sezione di introduzione del presente Documento, il punto 2 descriverà brevemente l'architettura del Sistema corrente, mentre al punto 3 verrà fornita una dettagliata descrizione dell'architettura del Sistema proposto. In particolare, questa sezione descriverà la decomposizione in sottosistemi, la corrispondenza tra hardware e software, la gestione dei dati persistenti, il controllo degli accessi, la sicurezza, il flusso di controllo e la gestione delle condizioni limite del Sistema proposto. Infine, il punto 4 descriverà i servizi offerti dai sottosistemi individuati.

2. Architettura del sistema corrente

Attualmente, nel mercato degli strumenti dedicati alla creazione di outfit, esistono diverse web application e mobile app che offrono servizi simili a quelli proposti da "OutfitMaker". Tuttavia, poiché queste soluzioni sono sviluppate da aziende private, la descrizione dettagliata delle architetture e degli strumenti utilizzati non è disponibile.

3. Architettura del sistema proposto

Il sistema proposto è una mobile app per piattaforme portatili, che vuole fornire agli utenti consigli di abbigliamento in base alle loro preferenze e al contesto comodamente nella propria tasca. L'architettura software scelta per la realizzazione del Sistema è la **Three Tier**, la quale, fornendo la possibilità di eseguire ciascun tier sulla propria infrastruttura, offre numerosi vantaggi, tra cui uno sviluppo più veloce e una maggiore scalabilità, affidabilità e sicurezza.

Questo pattern architetturale si compone di tre livelli:

1. **Interface:** è il tier di presentazione, dunque delle interfacce utente. Si occupa di visualizzare le informazioni all'utente e di raccogliere informazioni da quest'ultimo;
2. **Application Logic:** è il tier della logica di business dell'applicazione. Si occupa di elaborare le informazioni raccolte nel tier Interface e di aggiungere, rimuovere o modificare i dati nel tier Storage;
3. **Storage:** è il tier dell'accesso ai dati. Si occupa della persistenza delle informazioni e della loro gestione e archiviazione.

Come linguaggi e piattaforme di sviluppo app adottati per lo sviluppo del sistema si annoverano:

- **Android Studio:** Per lo sviluppo dell'interfaccia grafica e del back-end.
- **Firestore:** utilizzato come database basato su cloud per la gestione dei dati.
- **Firebase:** adottato per implementare il sistema di autenticazione degli utenti.
- **Firebase SDK:** impiegato per stabilire la connessione con il database.

Nel Sistema realizzato con la suddetta architettura tutte le comunicazioni passano attraverso l'Application Logic tier; l'Interface tier e lo Storage tier non comunicano mai direttamente tra loro.

3.1 Panoramica

Di seguito, vengono illustrati i risultati della fase di progettazione del Sistema proposto. Viene descritta l'architettura del sistema, stabilita in base alla struttura del sito web e alle funzionalità che deve offrire. Particolare attenzione è stata riposta nella gestione dei dati persistenti, i quali risultano fondamentali allo scopo del sistema proposto.

3.2 Decomposizione in sottosistema

Il sistema è stato decomposto nei seguenti sottosistemi:

- **Registrazione**: Il sottosistema si occupa della creazione dell'utente.
- **Login**: Il sottosistema è garante dell'accesso al sistema degli attori. (Admin, Utente)
- **Area Utente**: Il sottosistema che definisce la visualizzazione dell'area personale e la relativa modifica dei dati dell'area personale e il log out.
- **Generazione**: Il sottosistema che si occupa dell'operazione di generazione di un outfit, tramite l'inserimento di preferenze da parte dell'utente e il consulto del sottosistema "Armadio", per verificare i capi disponibili.
- **Creazione:** Il sottosistema si occupa della creazione di un outfit, accedendo ai dati dell'armadio e scegliendo i vari capi per la creazione.
- **Armadio**: Il sottosistema è responsabile dell'aggiunta, la rimozione e la modifica di un capo, e la visualizzazione dell'armadio.
- **Archivio:** Il sottosistema gestisce il salvataggio e la rimozione di un outfit in archivio in seguito ad una generazione di un outfit.
- **Firestore:** Il database che si occupa di gestire la persistenza dei dati assicurando che le informazioni sugli utenti, sugli outfit e sull'armadio virtuale siano salvate e recuperate correttamente.
- **Firebase SDK:** Il sottosistema collega i sottosistemi Registrazione, Armadio e Archivio al database.

Component Diagram UML

Nel seguente Component Diagram UML sono mostrati i sottosistemi e le relative dipendenze tra essi.

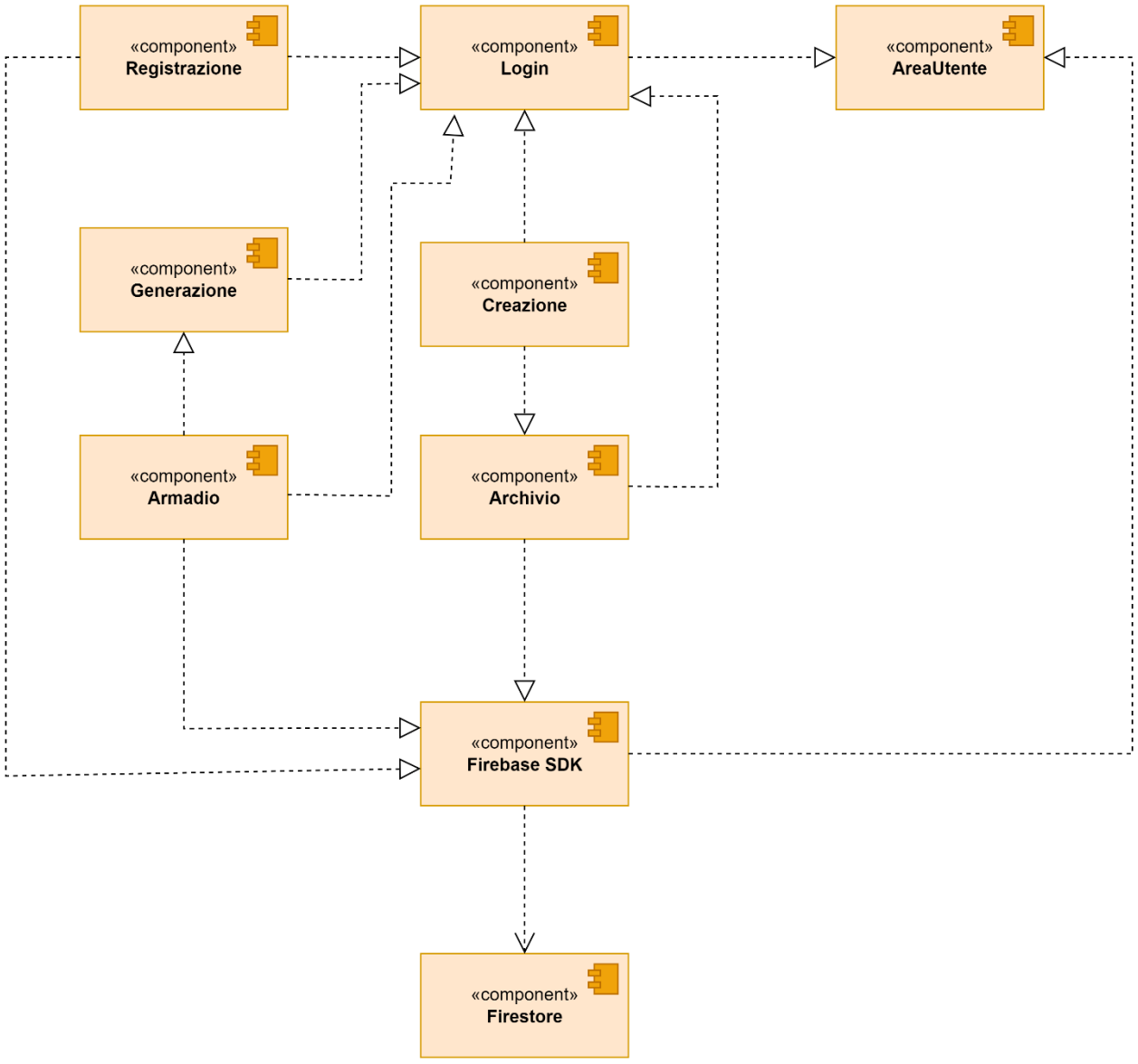
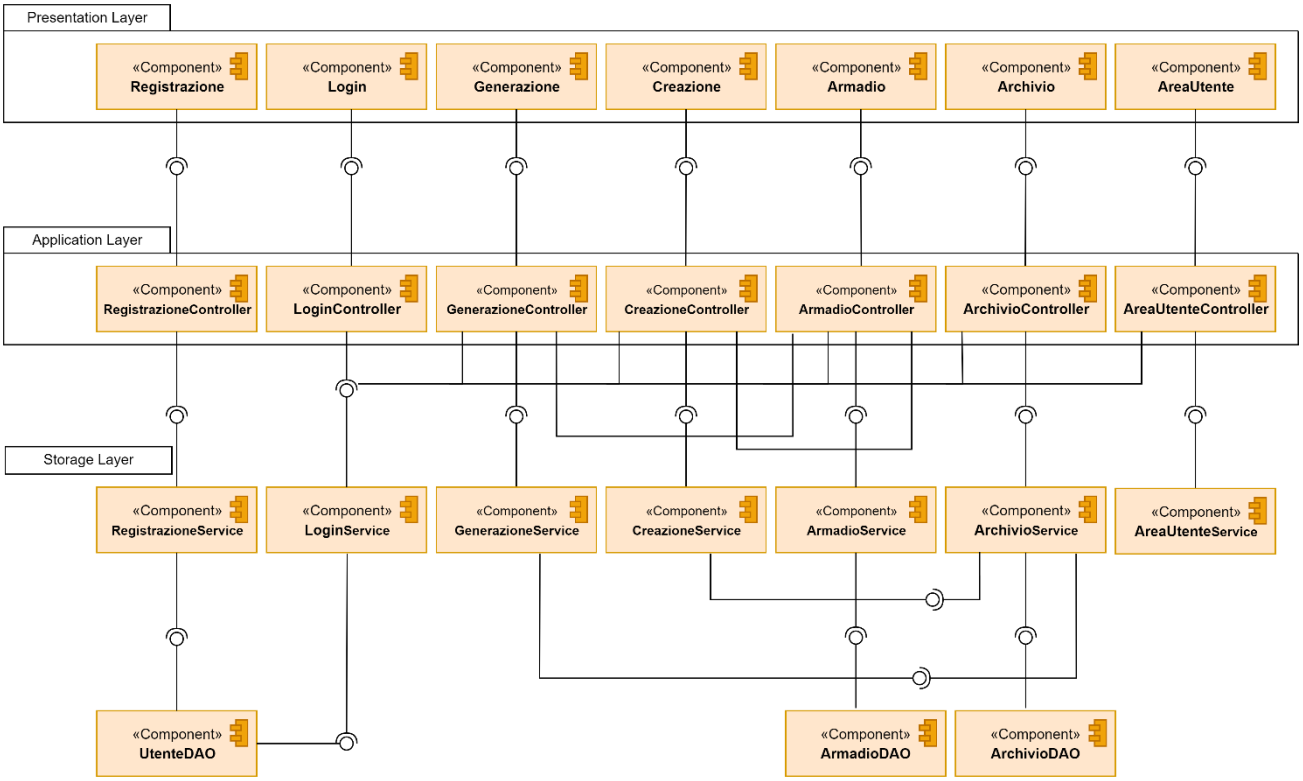


Diagramma Architetturale

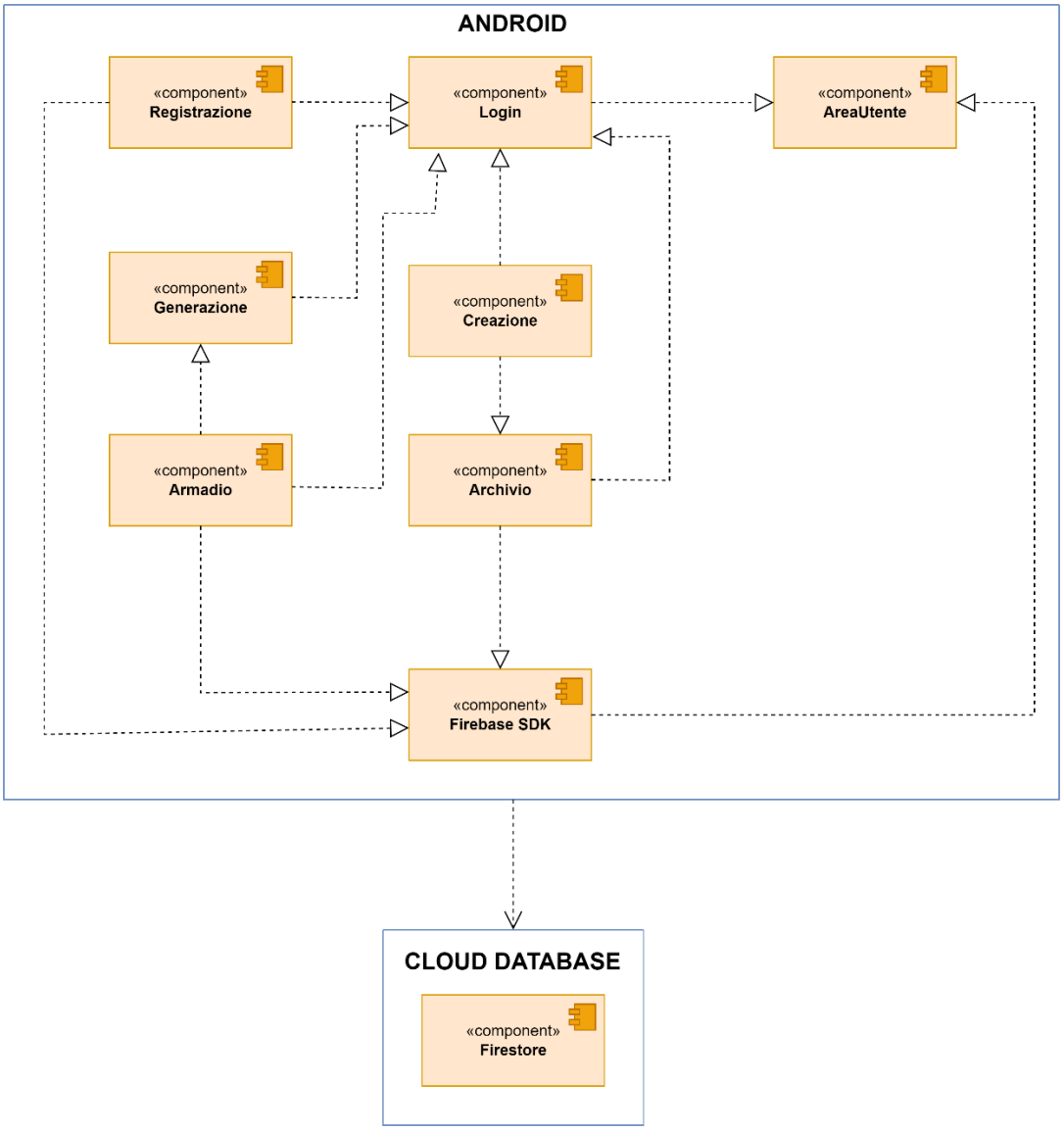


3.3 Mapping Hardware/Software

La piattaforma sviluppata sarà una mobile app scaricabile dagli utenti su dispositivi mobile Android con una connessione ad Internet. I dati sono memorizzati su un cloud database con cui l'applicazione comunica direttamente tramite appositi plugin forniti dal cloud provider.

Di seguito un UML Deployment Diagram che descrive il mapping hardware/software:

UML Deployment Diagram

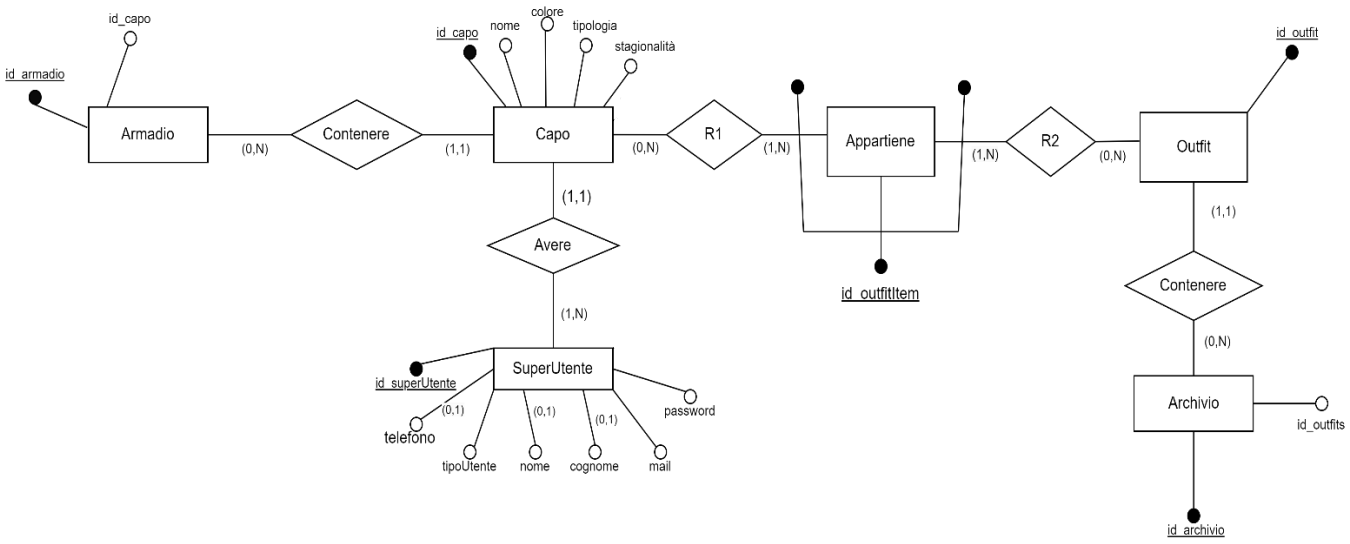


3.4 Gestione dei dati persistenti

Per la gestione dei dati persistenti del nostro sistema, abbiamo scelto di adottare un database NoSQL, non relazionale (Firebase). Questa scelta è stata guidata da diverse ragioni specifiche alle esigenze della nostra applicazione, che include l'uso di Android Studio, SQLiteOpenHelper e Firebase SDK:

1. **Prestazioni Ottimizzate:** Firebase è stato scelto per la sua efficienza nel gestire grandi quantità di dati, garantendo alte prestazioni. Questa caratteristica è cruciale per la nostra applicazione, che prevede la gestione di una mole considerevole di informazioni.
2. **Inserimento Sicuro dei Dati:** Firebase semplifica l'inserimento sicuro dei nuovi dati, offrendo un'interfaccia facile e veloce. Rispetto ai database SQL tradizionali.
3. **Migliore Gestione dei Dati Offline:** Firebase, integrato offre una gestione avanzata dei dati quando l'applicazione è offline. Questo aspetto è fondamentale, specialmente considerando la natura mobile della nostra piattaforma, garantendo che nessun dato venga perso anche in assenza di connessione Internet.
4. **Architettura App Serverless e Cloud-Native:** Firebase consente di sviluppare app senza la necessità di un server dedicato. Ogni utente avrà il proprio database locale, gestito tramite SQLiteOpenHelper, che sarà collegato al database cloud di Firestore. Questo approccio semplifica significativamente la gestione dei dati, consentendo un salvataggio diretto nel cloud.

3.4.1 Modello Concettuale Ristrutturato



3.4.2 Modello logico

SuperUtente: id_superUtente (PK), tipoUtente, nome (nullable), cognome (nullable), mail, password, sesso(nullable), telefono (nullable)

Capo: id_capo, id_superUtente (FK), nome, colore, tipologia, stagionalità, immagine (nullable)

Armadio Virtuale: id_armadio, id_capo (FK, 0, N)

Appartiene: id_outfitItem, id_capo (FK, 0, N), id_outfit (FK, 0, N)

Outfit: id_outfit

Archivio Outfit: id_archivio, id_outfit(FK, 0, N)

Vincoli di integrità referenziale
(VIR1) La chiave esterna "id_capo" della tabella "Capo" ha un vincolo di integrità referenziale con la chiave primaria "id_superUtente" della tabella "SuperUtente"
(VIR2) La chiave esterna della tabella "Armadio Virtuale" ha un vincolo di integritàreferenziale con la chiave primaria "id_capo" della tabella "Capo"
(VIR3) La chiave esterna della tabella "Appartiene" ha due vincoli di integrità referenziali con la chiave primaria "id_capo" della tabella "Capo" e con la chiave primaria "id_outfit" della tabella "Outfit"
(VIR4) La chiave esterna della tabella "Archivio Outfit" ha un vincolo di integrità referenziale con la chiave primaria "id_outfit" della tabella "Outfit"

3.5 Controllo degli accessi e sicurezza

Nella seguente sezione si riporta la matrice degli accessi per poter tracciare degli attori, che possono accedere ai servizi offerti dal sistema.

Oggetti/Attori	Utente	Admin
Registrazione	Registrazione	-
Login	Login	Login
AreaUtente	VisualizzaDati ModificaDati CancellazioneAccount Logout	VisualizzaDatiApp RimuoviUtente BloccaTemporaneamenteApp Logout
Generazione	GeneraOutfit SalvaOutfit	GeneraOutfit SalvaOutfit
Creazione	CreaOutfit SalvaOutfit	CreaOutfit SalvaOutfit
Armadio	VisualizzaIndumenti InserisciIndumento ModificaIndumento RimuoviIndumento CercaIndumento	VisualizzaIndumenti InserisciIndumento ModificaIndumento RimuoviIndumento CercaIndumento
Archivio	VisualizzaOutfit RimuoviOutfit	VisualizzaOutfit RimuoviOutfit

3.6 Controllo del flusso globale del sistema

Il sistema OutfitMaker è un sistema software interattivo; ciò significa che, tramite l'utilizzo di un'interfaccia grafica, l'utente impartisce dei comandi che avviano una funzionalità specifica. L'utente invia un comando tramite interfaccia grafica, il sistema selezionerà quindi il controllo corrispondente. Questo porterà il flusso allo strato logico contenente i moduli software che erogano dei servizi implementati in maniera modulare. Infine, si va allo strato logico che contiene la struttura dati utilizzata per immagazzinare, processare e condividere informazioni. Come control Flow si usa Event-Driven Control.

1. Interfaccia Grafica e Comandi Utente:

L'utente interagisce con il sistema attraverso un'interfaccia grafica, impartendo comandi.

2. Selezione del Controllo Corrispondente:

Quando l'utente invia un comando tramite l'interfaccia grafica, il sistema seleziona il controllo corrispondente.

3. Strato Logico dei Moduli Software:

Il flusso si sposta allo strato logico contenente moduli software che erogano servizi implementati in modo modulare. Questo suggerisce una struttura in cui i vari moduli si occupano di specifiche funzionalità del sistema.

4. Strato Logico della Struttura Dati:

Infine, il flusso si sposta allo strato logico che contiene la struttura dati utilizzata per immagazzinare, processare e condividere informazioni.

3.7 Condizioni Boundary

Si presentano le boundary condition inerenti all'avvio del sistema, spegnimento del sistema, fallimento del sistema ed errore di accesso ai dati persistenti.

Identificativo	UCBC_1 – Avvio del sistema	Data	04/12/2023
		Versione	1.0
		Autori	Letterese Michele
Descrizione	L'UC consente l'avvio del sistema		
Attore principale	Utente		
Attore secondario	NA		
Entry Condition	L'utente è connesso ad internet AND L'utente vuole avviare l'app AND L'utente deve essere registrato o loggato		
Exit Condition On success	L'utente può utilizzare l'app correttamente.		
Exit condition On failure	L'utente riceve una notifica in caso di problemi durante l'avvio		
Flusso di eventi principali			
1	Utente	L'utente procede ad avviare il sistema mediante l'apposito comando	
2	Sistema	Verifica la connessione al database Firestore e recupera i dati dal database cloud	
4	Utente	Utilizza l'applicazione	

Identificativo	UCBC_2 – Spegnimento del sistema	Data	04/12/2023
		Versione	1.0
		Autori	Letterese Michele
Descrizione	L'UC descrive il comportamento del sistema durante la terminazione		
Attore principale	Utente		
Attore secondario	NA		
Entry Condition	Il sistema è stato precedentemente avviato correttamente AND Il sistema non è ancora spento		
Exit Condition On success	Il sistema si chiude correttamente		
Exit condition On failure	Il sistema non viene chiuso correttamente		
Flusso di eventi principali			
1	Utente	L'utente richiede la chiusura dell'app o esegue il comando di spegnimento del Sistema	
2	Sistema	Salva i dati nel database	
3	Sistema	Procede all'arresto dell'applicazione	

Identificativo	UCBC_3 –Errore di accesso ai dati persistenti	Data	04/12/2023
		Versione	1.0
		Autori	Buongiorno Rocco Pio
Descrizione	L'UC descrive il comportamento del sistema in caso di errore di accesso ai dati persistenti o quando non i dati persistenti risultano essere corrotti.		
Attore principale	Sistema		
Attore secondario	NA		
Entry Condition	Il sistema non riesce ad accedere ai dati persistenti OR I dati risultano corrotti		
Exit Condition On success	Il sistema riprende il normale funzionamento		
Exit condition On failure	Il sistema non riprende il normale funzionamento		
Flusso di eventi principali			
1	Sistema	Cessa di eseguire le richieste.	
2	Sistema	Restituisce un messaggio di errore e invita l'utente a riprovare più tardi	

4. Servizi dei sottosistemi

Servizi dei sottosistemi

Sottosistema Registrazione

Servizio	Descrizione	Interfaccia
Registrazione utente	Questa funzionalità permette la registrazione all'ospite dell'app, permettendogli di diventare un utente del sistema	*RegistrazioneService*

Sottosistema Login

Servizio	Descrizione	Interfaccia
Login utente	Questa funzionalità permette di effettuare l'accesso all'app tramite le proprie credenziali	*LoginService*

Sottosistema AreaUtente

Servizio	Descrizione	Interfaccia
Logout utente	Questa funzionalità permette all'utente di disconnettere l'account dall'applicazione	*AreaUtenteService*
Visualizzazione dati personali	Questa funzionalità permette all'utente di visualizzare i dati personali inseriti nel sistema	*AreaUtenteService*
Modifica dati personali	Questa funzionalità permette all'utente di modificare i dati personali inseriti nel sistema	*AreaUtenteService*
Cancellazione account	Questa funzionalità permette all'utente di eliminare il proprio account dal sistema	*AreaUtenteService*

Sottosistema Generazione

Servizio	Descrizione	Interfaccia
Generazione Outfit	Questa funzionalità permette all'utente di generare un outfit con il supporto dell'intelligenza artificiale	*GenerazioneService*
Salva Outfit	Questa funzionalità permette all'utente di salvare l'outfit generato	*GenerazioneService*

Sottosistema Creazione

Servizio	Descrizione	Interfaccia
Creazione Outfit	Questa funzionalità permette all'utente di creare il proprio outfit componendolo manualmente con i propri indumenti	*CreazioneService*
Salva Outfit	Questa funzionalità permette all'utente di salvare l'outfit creato	CreazioneService

Sottosistema Armadio Virtuale

Servizio	Descrizione	Interfaccia
Visualizzazione indumenti	Questa funzionalità permette all'utente di visualizzare gli indumenti inseriti nel suo armadio	*ArmadioService*
Inserimento indumento	Questa funzionalità permette all'utente di inserire un indumento nel suo armadio	*ArmadioService*
Ricerca indumento	Questa funzionalità permette all'utente di ricercare un indumento nel suo armadio	*ArmadioService*
Modifica indumenti	Questa funzionalità permette all'utente di modificare i dati di un indumento inserito nell'armadio	*ArmadioService*
Rimozione indumenti	Questa funzionalità permette all'utente di rimuovere un indumento inserito nell'armadio	*ArmadioService*

Sottosistema Archivio Outfit

Servizio	Descrizione	Interfaccia
Visualizzazione outfit	Questa funzionalità permette all'utente di visualizzare gli outfit salvati nell'archivio	*ArchivioService*
Rimozione outfit	Questa funzionalità permette all'utente di rimuovere un outfit salvato nell'archivio	*ArchivioService*