Requisiti UniMeet

Versione 1.0.0 del 20/11/25

Versione	Modifiche
1.0.0	Compilazione

Bregolin Gabriele
Leggeri Leonardo
Vecchi Samuele

1. Ingegneria dei Requisiti

1.1 Introduzione

UniMeet è una piattaforma di organizzazione pensata per gli studenti universitari, il cui scopo è facilitare ed incentivare la creazione di gruppi di studio.
Gli studenti, ovvero gli utenti della applicazione, avranno la possibilità di creare sessioni di studio visibili ai colleghi, quest'ultimi se interessati potranno richiedere di partecipare.

1.2 Tecniche di Elicitazione

L'approccio si è basato su:

- **Interviste agli Stakeholder:** Definizione dei flussi critici per studenti e amministratori.
- **Analisi basata sullo Scenario:** Modellazione del "User Journey" dalla ricerca della sessione al rilascio di un feedback post-studio.
- **Descrizione in Linguaggio Naturale:** Traduzione dei bisogni dell'utente in una lista esaustiva di requisiti atomici.

1.3 Requisiti Funzionali (RF)

A. Gestione Accessi e Identità

- **RF.01:** il sistema deve permettere la registrazione di nuovi utenti.
- **RF.02:** il sistema deve gestire l'autenticazione tramite Spring Security.
- **RF.03:** il sistema deve discriminare gli utenti in tre ruoli: **Studente**, **Amministratore Locale** e **Amministratore Generale**.
- **RF.04:** gli admin sono sono studenti con permessi di accesso e modifica al database.

B. Gestione Risorse Fisiche - Admin

- **RF.04:** l'admin locale ha visibilità del database limitata all'Università di riferimento.
- **RF.05:** l'admin generale ha visibilità completa sul database.
- **RF.06:** l'admin generale è l'unico a poter aggiungere/modificare/eliminare l'entità 'università'.
- **RF.06:** l'admin deve poter configurare/modificare/eliminare la gerarchia: Università > Edificio > Aula > Tavolo > Posti.
- **RF.07:** l'admin deve poter configurare/modificare/eliminare la gerarchia: Università > Dipartimento > Corso di Studio > Materia.
- **RF.11:**l'admin deve poter cercare le sessioni di studio con filtri di ricerca all'interno dell'università.
- **RF.12:**l'admin deve poter consultare la propria area personale compilando con i propri dati personali.
- **RF.13:**l'admin deve avere una statistica per vedere quanti edifici sta gestendo.
- **RF.14:** l'admin deve poter cercare gli utenti attraverso una ricerca fatta attraverso dei filtri di ricerca.
- **RF.15:** l'admin deve poter chiedere amicizia a qualsiasi utente.

- **RF.16:** l’admin deve disporre di un'area per le notifiche da gestire.

C. Gestione Sessioni di Studio

- **RF.08:** deve essere possibile creare ed eliminare una sessione.
- **RF.09:** la sessione dovrà avere i seguenti dettagli: materia, luogo, visibilità, fascia oraria e la descrizione.
- **RF.10:** le sessioni sono di due tipi: Pubbliche e Private.
- **RF.11:** le sessioni Private sono visibili solo dal proprietario. Gli altri utenti possono parteciparvi solo su invito.
- **RF.12:** il sistema deve impedire la creazione di sessioni in aule già saturate o in orari sovrapposti per lo stesso creatore.
- **RF.13**: il sistema deve impedire di invitare utenti che hanno già un impegno per il giorno e orario definito.
- **RF.14:** l'utente deve poter filtrare per materia, location e giorno (indicando da-a).
- **RF.12:** l'utente deve poter indicare la propria presenza a una sessione pubblica.
- **RF.15:** l’utente deve poter abbandonare una sessione.
- **RF.16:** il sistema deve disporre di sezioni per visualizzare le sessioni proprietarie, di colleghi, passate, suggerite e “in corso” (quest’ultima deve risaltare).
- **RF.17:** le sessioni suggerite si basano sulla conciliazione tra la materia trattata e gli esami dichiarati come ‘da sostenere’.
- **RF.18:** deve essere possibile valutare i proprio compagni al termine della sessione.

D. Dinamiche Social e Networking

- **RF.19:** l’utente deve poter richiedere l’amicizia (diventando colleghi reciprocamente).
- **RF.20:** diventando colleghi, gli utenti si possono invitare alle sessioni private proprietarie e visualizzare le sessioni pubbliche dei colleghi in una sezione apposita.
- **RF.21:** il sistema deve permettere di accettare o rifiutare richieste di "following" (diventare colleghi).

E. Sistema di Notifiche

- **RF.22:** il sistema deve inviare notifiche asincrone per:
    - Invito ad unirsi a una sessione privata.
    - Nuova richiesta di "following".

F. Area Personale e Reputazione

- **RF.23:** l’utente può visualizzare il proprio profilo nell’area personale
- **RF.24:** l’utente può modificare il proprio profilo e visualizzare quello degli altri
- **RF.25:** il profilo deve essere diviso in 3 aree: anagrafica, percorso di studi, carriera
- **RF.26:** l’area ‘anagrafica’ deve contenere: nome, cognome, bio, reputazione
- **RF.27:** l’area ‘percorso di studi’ deve contenere: università, dipartimento, tipo di laurea, corso di laurea e anno che l’utente freuentata.
- **RF.28:** l’area ‘carriera’ deve contenere: corsi preferiti e trovati difficili (max 3 per tipo), esami passati e da sostenere.

- **RF.29:** la reputazione è calcolata sulla base del punteggio ricevuto da ogni compagni di sessione (considerando tutte le sessioni frequentate) facendone una media.

1.4 Requisiti di Sistema e Non Funzionali (RNF)

Requisiti di Sistema (RS)

- **RS.01:** interfaccia Web basata su **Vaadin**.
- **RS.02:** backend sviluppato con **Spring Boot**.
- **RS.05:** gestione delle dipendenze e build tramite **Maven**.
- **RS.06:** persistenza su database embedded **H2** con archiviazione su file non volatile.

Requisiti Non Funzionali - Qualità (RNF)

- **RNF.01 (Sicurezza):**
    - Il sistema deve gestire l'autenticazione tramite Spring Security.
    - Il sistema deve proteggere le credenziali tramite hashing (BCrypt) nel database.
- **RNF.02 (Affidabilità):**
    - il sistema deve garantire la coerenza e consistenza dei dati
- **RNF.03 (Usabilità):**
    - l'applicazione deve permettere una navigazione fluida.
    - l'interfaccia deve essere intuitiva e funzionale.
- **RNF.04:** l'intero sistema dovrà essere sviluppato in lingua inglese

2. Qualità del Software e Metriche

Allo scopo di valutare e monitorare la qualità del software sviluppato, verranno utilizzati tool specifici come stan4j.

2.1 Tool di Analisi: stan4j (stanide)

L'integrazione di stan4j con eclipse, permetterà di valutare le dipendenze tra i package e l'analisi delle principali metriche orientate agli oggetti per valutare la qualità del software sviluppato.
Di seguito alcune metriche che verranno analizzate:

- **Complessità Ciclomatica(McCabe)**: utile ad individuare metodi eccessivamente complessi.
- **Weighted Method per Class**: utile ad evidenziare classi complesse e difficili da comprendere e mantenere.
- **Coupling Between Objects:** utile ad evidenziare relazioni tra classi e pacchetti.
- **Response For a Class:** utile per valutare la complessità d'uso e di test di una classe considerando il numero di metodi eseguiti in risposta a un messaggio ricevuto da un oggetto della classe.
- **Lack of Cohesion in Methods:** utile ad evidenziare la coesione interna di una classe valutando quanto i metodi della classe stessa utilizzano la medesima variabile di istanza.