SDD

**1.1 Scopo del sistema**

La piattaforma LetsMeet di pone l’obiettivo di poter dare l’opportunità, a qualsiasi tipologia di utente interessato alla creazione e monitoraggio di eventi, di poter creare eventi in real-time sul territorio attraverso una mappa interattiva.

Usualmente, chi interagisce con sistemi che offrono servizi di creazione e condivisione degli stessi eventi, mettono in secondo luogo l’interazione con la mappa, la quale funge solo come eventuale scorciatoia per accedere ad altre applicazioni, come Google Maps, per la visualizzazione del luogo d’incontro.

LetsMeet vuole concentrare l’attenzione proprio sull’utilizzo di una mappa dando anche la possibilità agli utenti di visionare in tempo reale anche il numero dei partecipanti effettivi agli eventi, cosa che effettivamente manca ad altre tipo di piattaforme in circolazione che si prefiggono l’obiettivo di creare eventi.

**1.2 Obiettivi di Design**

La piattaforma LetsMeet deve poter essere il più efficiente ed intuitiva possibile attraverso agevoli interfacce. Tale efficienza sarà costruita attraverso rapidi tempi di risposta ad ogni genere di input ma anche differenti politiche di tolleranza all’errore.

Per ottenere gli obiettivi finali vanno seguiti dei criteri di progettazione tenendo presente:  
Performance, Affidabilità, Costi, Manutenzione e Utente Finale.

**1.2.1 Criteri di Performance**

**Tempo di Risposta:**

LetsMeet deve essere reattivo per tutte le operazioni più immediate come la creazione di un evento o la verifica della partecipazione ad un evento.

Per operazioni massive, come il caricamento di più eventi sulla mappa, il sistema deve garantire tempi di risposta ragionevoli nell’ordine delle decine di secondi, ma, essendo un sistema web, molto dipenderà dalla qualità della connessione e dalla congestione della rete sul sistema online.

**Throughput:**

I picchi di carico, fino a circa 500 (cinquecento) utenti simultaneamente collegati, devono essere gestiti dal sistema senza rallentamenti, garantendo fluidità e una latenza molto bassa.

Il sistema deve garantire consistenza in tutte le operazioni che includono chiamate al database, specialmente nei momenti di maggiore carico.

**Memoria:**

Il sistema utilizza un database relazionale per memorizzare tutti i dati e la mole dei dati non rappresenterà un problema di performance del sistema.

**1.2.2 Criteri di affidabilità**

**Robustezza**

I componenti devono essere affidabili ed essere in grado di poter  
mantenere i propri dati anche in caso di guasti come problemi  
elettrici, guasti dell'hardware, attacchi informatici, problemi legati al  
browser.

**Disponibilità**

LetsMeet deve essere disponibile all’uso, 24 ore su 24, da parte degli  
utenti, grazie ad un server sempre attivo.

**Tolleranza all’errore**

**(vede vittorio)**

**Sicurezza**

Le tecniche utilizzate, per garantire la sicurezza, sono basate prevalentemente su una ‘login’, la quale permette il riconoscimento dell’utente.  
Tutti gli accessi al sistema avvengono tramite un’operazione di routing che controlla se l’utente che sta operando sul sistema ha i permessi adeguati per usufruire di determinate funzionalità.

C’è, quindi, un controllo che verifica i permessi degli utenti che verranno adeguatamente distribuiti sulla base del grado di importanza e responsabilità degli Utenti.

L’Utente Moderatore potrà godere di funzionalità aggiunte sopra gli altri Utenti, potendo sospenderli o meno dalla piattaforma.

La figura del super-admin è sviluppata per poter indicare un Utente della piattaforma e renderlo Moderatore.

🡪(controllare se è fattibile) Inoltre, le password sono codificate in MD5 senza essere rese in chiaro in nessun campo di nessuna tabella che tiene traccia di  
queste informazioni all’interno del database.

**1.2.3 Criteri di costo**

**Sviluppo**

I costi di sviluppo previsti riguardano l’affitto del server per la fase di sviluppo. È nostra intenzione utilizzare come web Server Altervista(vedere vittorio). Per quanto riguarda il DBMS utilizzeremo MySQL che è open source.

Nei costi di sviluppo rientrerebbero anche le licenze di Google per poter usufruire del servizio Maps, costi mensili per il numero di interazioni con la mappa.

**Deployment**

**Vedere vittorio**

**1.2.4 Criteri di mantenimento**

**Estensibilità**

La progettazione del sistema sarà condotta in modo da agevolare la facile introduzione di nuove funzionalità utilizzando il linguaggio di markup HTML5, i fogli di stile CSS3, JQuery e   
Java.

**Modificabilità**

Deve essere possibile intervenire sul codice esistente per correggere eventuali bugs o implementare nuove funzionalità. Bisogna garantire che il codice sia leggibile per rendere agevole la modifica.

**Leggibilità**

Il codice sarà ben strutturato per semplificare eventuali interventi su di esso.

**Tracciabilità dei requisiti**

Grazie alla tracciabilità̀ dei requisiti, sarà̀ possibile effettuare le modifiche necessarie al corretto funzionamento del sistema, valutando correttamente i costi e i rischi che le modifiche porteranno.

**1.2.5 Criteri per l’utente finale**

**Usabilità**

LetsMeet deve essere facilmente apprendibile (l’utente deve essere in grado di interagire con il sistema e padroneggiare le funzionalità in modo rapido), deve essere flessibile e robusto (l’utente deve essere in grado di capire quando ha successo nel perseguire i suoi obiettivi nel sistema o quando sta sbagliando qualcosa per poter intervenire).

**1.3 Definizione ed acronimi**

* **LetsMeet:** La piattaforma di creazione eventi in questione.
* **Utente:** Utente generico registrato alla piattaforma che può creare, partecipare ad eventi ed utilizzare tutte le funzioni a disposizione della piattaforma.
* **Moderatore:** Utente generico registrato alla piattaforma che può visualizzare le segnalazioni fatte dagli Utenti e può sospenderli eventualmente.
* **Super-Admin:** Utente esterno che può indicare un Utente come Moderatore.
* **Login:** attività di accesso all’account;
* **Logout:** attività di uscita dell'account connesso;
* **DBMS:** Database Management System;
* **SQL:** Structured Query Language, ed è linguaggio di interrogazione (dei database)  
  strutturato.
* **RAD:** Requirements Analysis Document

**1.4 Riferimenti**

L’insieme del materiale di riferimento utilizzato per la realizzazione del progetto e per la  
stesura di questo stesso documento comprende:

* Libro di testo: B.Bruegge, A.H. Dutoit, Object Oriented Software Engineering - Using  
  UML, Patterns and Java, Prentice Hall, 3rd edition, 2009;
* Slide del docente, reperibili sulla piattaforma;
* Documento RAD del progetto LetsMeet;

**1.5 Panaromica**

Il documento si compone di una prima parte in cui vengono introdotti gli obiettivi di design.  
Verrà poi brevemente spiegata l’architettura corrente del sistema prima e dopo il reengineering ma prima di addentrarci nel discutere l’architettura, è doveroso fare un excursus alle attività di system design che compongono le basi per l’architettura software del sistema.

* **Decomposizione del sistema**: il sistema viene decomposto in sottosistemi allo scopo di  
  poter assegnare ad un team parti di software semplici da sviluppare. Un sottosistema è  
  una collezione di classi, associazioni, operazioni, eventi e vincoli che sono in relazione tra  
  di loro. Un insieme di operazioni correlate forma un servizio. Ogni sottosistema quindi sarà  
  caratterizzato dai servizi che offre ad altri sottosistemi; l’insieme dei servizi che un  
  sottosistema espone sarà denominato Interfaccia(API).
* **Mapping Hardware/Software**: descrive come i sottosistemi vengono assegnati  
  all’hardware e alle componenti “off-the-shelf”. Elenca anche le problematiche introdotte da  
  nodi multipli e dal riuso del software.
* **Gestione dati persistenti**: descrive i dati persistenti memorizzati dal sistema e  
  l’infrastruttura di gestione richiesta per essi.
* **Politiche di accesso e sicurezza**: descrive il modello utente del sistema in termini di una  
  matrice degli accessi, stabilendo in modo più preciso le operazioni e le informazioni  
  effettuabili da ogni singolo attore e come questi si autenticano al sistema.
* **Flusso di controllo globale**: descrive quali operazioni eseguire ed in che ordine, per  
  garantire il corretto flusso di controllo del sistema.
* **Condizioni Limite**: descrive lo start-up, lo shutdown e i comportamenti errati del sistema.

**2. Architettura del software corrente**

Attualmente un sistema come LetsMeet non è in circolazione, ma esistono piattaforme che si prefiggono l’obiettivo di condividere eventi ed utilizzano una mappa interattiva unicamente per poter visionare il percorso oppure il luogo dell’evento interessato.

La maggior parte delle piattaforme offrono all’utente unicamente una pagina con le informazioni dell’evento ed eventualmente un numero di partecipanti, senza dare un’effettiva conferma del numero dei partecipanti effettivi ad un evento.

Google in parte offre un servizio servizi di visualizzazione di punti d’interesse su di una mappa, ma non ha le funzionalità del sistema LetsMeet proposto.

**3. Architettura del software proposto**

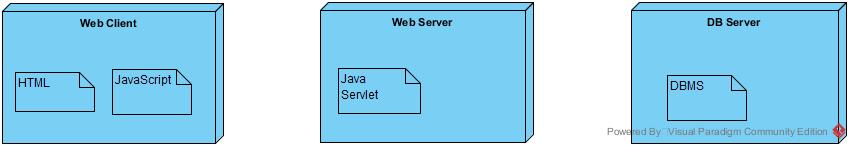
Il sistema proposto è una fusione di servizi che offrono Google ed altri sistemi di condivisione di eventi.

**3.1 Panoramica**

Il sistema proposto è un software web-based per la gestione dei propri eventi online. Le tipologie di utenti si dividono in: utente, moderatore e super-admin. L’utente usufruirà della quasi totalità dei servizi offerti dalla piattaforma; il moderatore accederà alle stesse funzionalità di un utente e in più potrà accedere alle funzionalità che si occupano di gestire gli utenti, dando la possibilità di poter visionare le eventuali segnalazione di commenti o eventi e poter sospendere dalla piattaforma gli utenti segnalati; infine il super-admin potrà gestire solamente l’eventuale trasformazione di un utente in moderatore, poi potrà accedere come utente alla piattaforma.

L’architettura del sistema si presenta come una architettura Client/Server per i seguenti motivi:

* **Portabilità:** il sistema (essendo web-based) potrà essere utilizzato su una varietà di  
  macchine e sistemi operativi.
* **Trasparenza:** il sistema nonostante sia distribuito è in grado di fornire i propri servizi al  
  singolo utente senza interferire con gli altri utenti del sistema.
* **Performance:** il sistema garantirà reattività per i task degli utenti collegati ma essendo  
  web-based molto dipenderà dalla qualità della connessione online.
* **Scalabilità:** il sistema sarà in grado di gestire un grosso numero di utenti connessi  
  contemporaneamente che effettuano le operazioni in contemporanea.
* **Flessibilità:** Il sistema fornirà un’interfaccia grafica intuitiva e con funzionalità  
  specifiche per il ruolo dell’utente che effettuerà l’accesso.
* **Affidabilità:** Entrambi i componenti client e server devono essere affidabili ed essere  
  in grado di mantenere i propri dati anche in seguito a guasti, quindi deve essere  
  possibile effettuare dei backup periodici al database.



**3.2 Decomposizione del sistema**

Per realizzare la piattaforma LetsMeet è stato utilizzato lo stile architetturale three-tier in versione  
Client/server. L’architettura three-tier ("a tre strati") indica una particolare architettura software  
di tipo multi-tier per l’esecuzione di un'applicazione web che prevede la suddivisione  
dell'applicazione in tre strati dedicati rispettivamente alla interfaccia utente, alla logica  
funzionale e alla gestione dei dati persistenti.

In particolare, i tre strati si occupano di differenti funzionalità del sistema di seguito descritte:

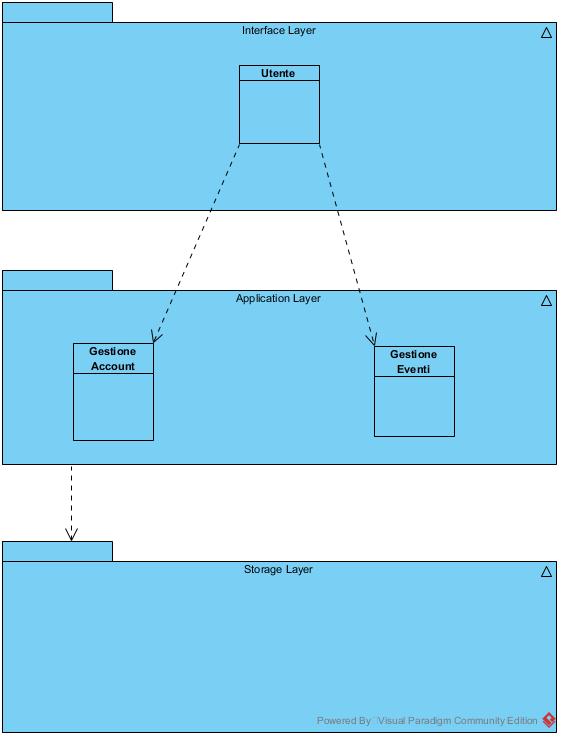
* **Interface Layer:** Include tutte le interfacce grafiche e in generale i boundaryobjects, come le form con cui interagisce l'utente. L’ interfaccia verso l’utente è rappresentata da un Web server e da eventuali contenuti statici (es. pagine HTML).
* **Application Layer** include tutti gli oggetti relativi al controllo e all’elaborazione dei  
  dati. Questo avviene interrogando il database tramite lo storage layer per generare  
  contenuti dinamici e accedere a dati persistenti
* **Storage Layer** effettua la memorizzazione, il recupero e l'interrogazione degli  
  oggetti persistenti. I dati, i quali possono essere acceduti dall’application layer, sono  
  depositati in maniera persistente su un database tramite DBMS.

Riportiamo una breve descrizione delle gestioni del sistema:

* **Gestione account:** Questa funzionalità raccoglie tutte le informazioni per gestire l’autenticazione degli utenti su LetsMeet, la trasformazione di un profilo utente in moderatore e la ricerca/visualizzazione dei profili utenti.
* **Gestione Eventi:** Questa funzionalità raccoglie tutte le informazioni per la gestione degli eventi sulla piattaforma compresa la loro ricerca e l’invio dei messaggi inerenti a quegli eventi.
* **Gestione Segnalazione:** Questa funzionalità raccoglie tutte le informazioni per la gestione delle segnalazioni degli utenti.

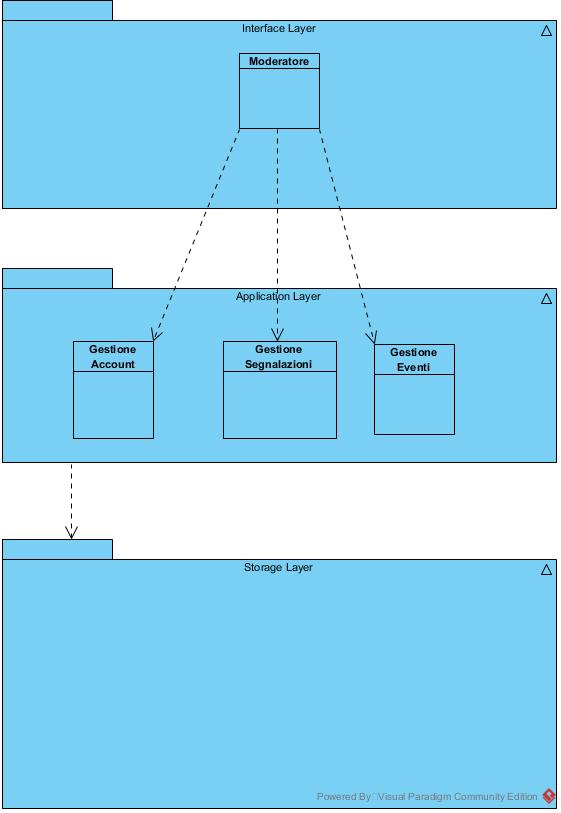
In seguito, verranno mostrati ogni singolo layer.

***Utente***



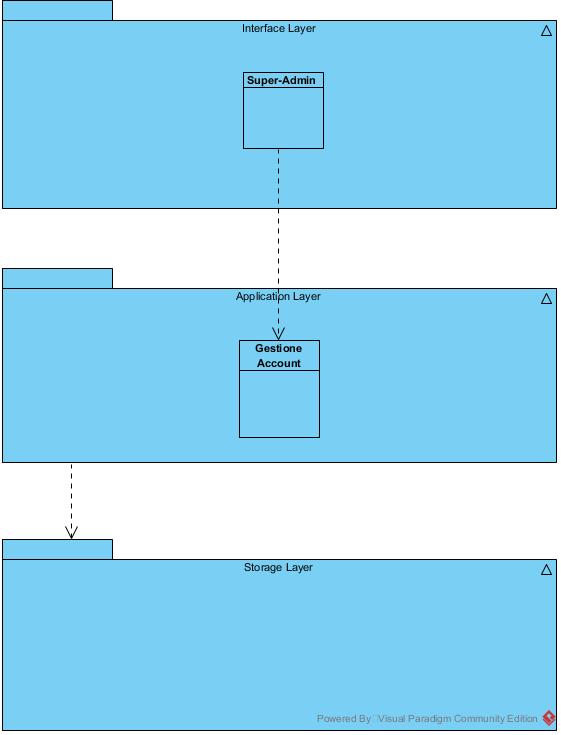
* **Gestione Account:** L’Utente potrà godere delle funzioni riguardanti la propria autentificazione all’interno della piattaforma, la visualizzazione del proprio profilo personale e quello degli altri utenti con eventuale ricerca.
* **Gestione Eventi:** L’Utente potrà godere delle funzionalità di creazione, partecipazione, verifica partecipazione di un evento e la possibilità di ricerca degli eventi con visualizzazione delle eventuali pagine di informazioni.

***Moderatore***



* **Gestione Account:** Il Moderatore potrà godere delle funzioni riguardanti la propria autentificazione all’interno della piattaforma, la visualizzazione del proprio profilo personale e quello degli altri utenti con eventuale ricerca.
* **Gestione Eventi:** Il Moderatore potrà godere delle funzionalità di creazione, partecipazione, verifica partecipazione di un evento e la possibilità di ricerca degli eventi con visualizzazione delle eventuali pagine di informazioni.
* **Gestione Segnalazioni:** Il Moderatore potrà godere delle funzioni di visualizzazione ed accettazione, compreso scarto, di segnalazioni effettuate dagli altri profili della piattaforma, sospendendo gli eventuali utenti segnalati.

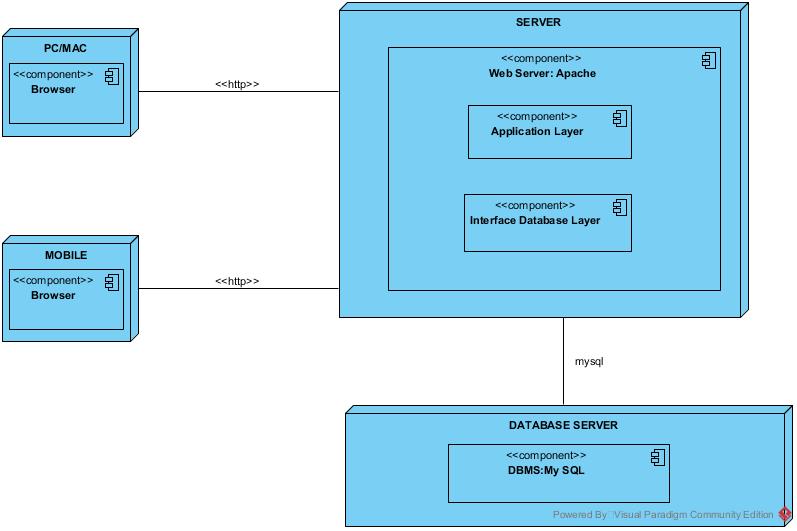
***Super-Admin***



* **Gestione Account:** Il Super-admin potrà godere della funzione di trasformazione degli utenti in moderatori.

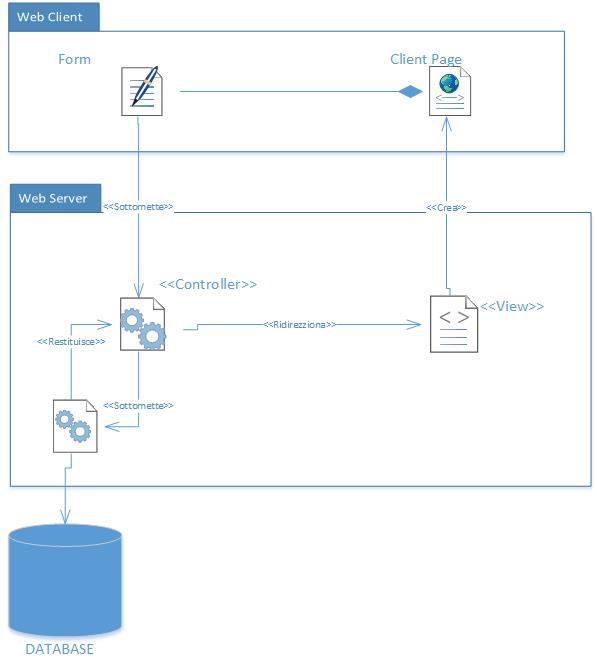
**3.3 Mapping Hardware /Software**

La struttura hardware proposta è costituita da un server centrale e dai client cioè un qualsiasi  
computer al quale un utente può collegarsi per sfruttare il browser per accedere al sistema  
LetsMeet. Al server si collegano i client ed il database, il tipo di utente è determinato in fase di  
autenticazione controllando nel database il tipo di utente che corrisponde allo Username  
inserito in quel client. I client dovranno effettuare richieste al server per eseguire le operazioni  
legate alle loro rispettive funzionalità. II client ed il server saranno connesse tramite una rete  
che utilizzerà il protocollo TCP/IP.



**Interface Layer**

L’utente utilizza il sistema mediante un Browser installato all’interno del suo calcolatore (ad es.  
Opera, Firefox, Chrome).  
**Application layer**Il sistema, e quindi le funzionalità, sono implementate in linguaggio JAVA. Il codice in JAVA  
verrà tradotto in linguaggio HTML e il codice risultante viene inviato al browser del client.  
**Storage layer**Rappresenta il collegamento con il server da parte del sistema e si occupa di tutte le richieste  
di accesso e modifiche sui dati permanenti presenti nel database.  
**Database Server***Il DBMS usato* è MySQL il quale presenta molte API che permettono l’interazione tra sistema  
e database.



A questo punto si fanno i diagrammi Run-Time, sincer n aggia e capit, arrop vrim.

**3.4 Dati persistenti**

Vedere il file (LetsMeet\_Dati\_Persistenti.pdf) allegato con la pubblicazione dei dati persistenti.

**3.5 Controllo degli accessi e della sicurezza**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Oggetti | Entità | Utente | Moderatore | Super-Admin |
| Account | | * Login * Logout * Registrazione * Ricerca\_Utenti * Visualizza\_Profilo\_PS | * Login * Logout * Registrazione * Ricerca\_Utenti * Visualizza\_Profilo\_PS | * Registrazione\_Mod |
| Eventi | | * Visualiazza\_Eventi * Creazione\_Evento * Rating\_Evento * Partecipa\_Evento * Verifica\_Partecipazione * Ricerca\_Eventi * Visualizza\_Evento * Scrittura\_Commento * VisualizzaCommenti | * Visualiazza\_Eventi * Creazione\_Evento * Rating\_Evento * Partecipa\_Evento * Verifica\_Partecipazione * Ricerca\_Eventi * Visualizza\_Evento * Scrittura\_Commento * VisualizzaCommenti |  |
| Segnalazioni | | * Segnala\_Evento * Segnala\_Commento | * Segnala\_Evento * Segnala\_Commento * Accetta\_Segnalazione * Rifiuta\_Seganalazione |  |

**3.6 Controllo del software globale**

Il controllo del flusso software viene gestito da classi java che interagendo con il client, il quale  
si interfaccia tramite un web browser, svolgendo le varie operazioni. Il server smista ogni nuova  
richiesta alla classe java adeguata, inoltrando poi la risposta al client.

**3.7 Condizioni Boundary**

Le condizioni limite riguardano l’accensione e lo spegnimento del sistema per quanto riguarda  
il lato Server. Dal lato Client si riferiscono agli errori di connessione al server.

**3.7.1 Avvio del sistema**Il sistema dopo essersi avviato presenta un’interfaccia ai client. Dopo aver effettuato l’autenticazione ogni utente può accedere alle funzionalità disponibili.

**3.7.2 Terminazione del sistema**È possibile terminare il sistema se e solo se tutti i sottosistemi sono stati disattivati in precedenza. Prima della disattivazione totale del database-server e dell’application-server verranno disconnessi tutti i client connessi al sistema; ad ognuno di loro arriverà una notifica prima della terminazione del sistema.  
La terminazione del sistema avviene solo nel caso in cui tutti i sottosistemi siano stati disattivati: nel caso in cui due sottosistemi siano ancora in esecuzione, il sistema rimane attivo. Per non incorrere in problemi - che possano scoraggiare il cliente nell’uso del sistema - prima di disattivare l’application server e il database server, è consigliabile disattivare prima tutti i client.  
Da qualsiasi postazione attraverso la funzione di logout è possibile disattivare ogni sottosistema. La disattivazione di un sottosistema da una qualsiasi postazione client ha come effetto la chiusura del suo terminale; sul database-server e sull’application-server tale funzione comporta la disattivazione di tali sottosistemi.

**3.7.3 Fallimento del sistema**Nel caso si verifichi un errore dovuto all’hardware o al software si cercherà di ripristinare una configurazione del sistema precedente allo stato d’errore.  
Poiché i dati sono gestiti dal DBMS non c’è alcun rischio di perderli. Tuttavia, non è da escludere la perdita dei dati, se si verifica un guasto al supporto di memorizzazione dei dati nel database-server.  
Per minimizzare questo rischio, si eseguiranno periodicamente dei backup del database del sistema e periodicamente il sistema hardware verrà sottoposto a controlli.

**Casi d’uso Boundary Condition**

|  |  |  |
| --- | --- | --- |
| Nome caso d’uso | Shutdown | |
| ID | UCSHUTDOWN | |
| Partecipanti | Amministratore | |
| Flusso di eventi: | UTENTE  1. Amministratore attiva la funzione ‘Spegni’ | SISTEMA  2. Il sistema controlla se ci sono eventuali client da disconnettere dopo di che avvia la procedura d’arresto. Notifica il successo dell’operazione. |
| Condizioni di entrata | Amministratore accede al sistema | |
| Eccezioni |  | |
| Condizioni di uscita | Il sistema si arresta con successo. | |
| Requisiti di funzionalità | Il sistema si arresta entro 1 minuto. | |

|  |  |  |
| --- | --- | --- |
| Nome caso d’uso | StartUp | |
| ID | UCSTARTUP | |
| Partecipanti | Amministratore | |
| Flusso di eventi: | UTENTE  1. Amministratore attiva la funzione ‘Accensione’ | SISTEMA  2. Il sistema attiva i propri servizi rendendosi disponibile in tutte le sue funzionalità e notifica il successo dell’operazione. |
| Condizioni di entrata | Amministratore accede al sistema | |
| Eccezioni |  | |
| Condizioni di uscita | Il sistema si attiva con successo. | |
| Requisiti di funzionalità | Il sistema si attiva entro 3 minuto. | |

**4.0 Servizi dei sottosistemi**

**4.1 Gestione Account**

|  |  |
| --- | --- |
| **Sottosistema** | Gestione Account |
| **Descrizione** | Sottosistema che gestisce l’autentificazione degli utenti, tutte le operazioni riguardanti la registrazione degli Utenti o Moderatori e le operazioni di ricerca e visualizzazione di account |
| **Servizi offerti** | **Descrizione** |
| Login | Questa funzionalità permette di effettuare l’accesso al sistema, tramite le proprie credenziali, per poi sfruttare le funzionalità che esso offre |
| Logout | Questa funzionalità permette di uscire dal sistema. |
| Registrazione Utente | Questa funzionalità permette di registrare un nuovo account della piattaforma |
| Registrazione Moderatore | Questa funzionalità permette di registrare un Moderatore della piattaforma |
| Ricerca Utente | Questa funzionalità permette di poter cercare un Utente che abbia un profilo sulla piattaforma |
| Visualizza Profilo Utente | Questa funzionalità permette di visualizzare la pagina profilo di un Utente, che si la propria o di un altro Utente |

**4.2 Gestione Eventi**

|  |  |
| --- | --- |
| **Sottosistema** | Gestione Eventi |
| **Descrizione** | Sottosistema che gestisce tutte le operazioni riguardante gli eventi sulla piattaforma, come la ricerca, la visualizzazione, la partecipazione, verifica della partecipazione e la scrittura dei commenti sugli eventi |
| **Servizi offerti** | **Descrizione** |
| Visualizzazione Eventi | Questa funzionalità permette di visualizzare gli eventi limitrofi alla posizione dell’utente |
| Creazione Evento | Questa funzionalità permette di creare un evento |
| Ratine Evento | Questa funzionalità permette di votare ad un evento a cui si è partecipati |
| Partecipa Evento | Questa funzionalità permette di indicare la propria partecipazione ad un evento |
| Verifica partecipazione Evento | Questa funzionalità permette di poter verificare la propria partecipazione ad un evento |
| Visualizza info Evento | Questa funzionalità permette di visualizzare le informazioni di un evento |
| Ricerca Evento | Questa funzionalità permette di ricercare un eventuale evento creato sulla piattaforma |
| Scrittura Commento | Questa funzionalità permette di scrivere un commento allegato ad un evento |
| Visualizzazione Evento | Questa funzionalità permette di visualizzare i commenti scritti inerenti ad un evento in precedenza |

**4.3 Gestione Segnalazione**

|  |  |
| --- | --- |
| **Sottosistema** | Gestione Segnalazione |
| **Descrizione** | Sottosistema che gestisce le segnalazioni inviate ai moderatori da parte degli utenti |
| **Servizi offerti** | **Descrizione** |
| Segnalazione Evento | Questa funzionalità permette di poter segnalare un evento creato sulla piattaforma |
| Segnalazione Commento | Questa funzionalità permette di poter segnalare un commento scritto sulla piattaforma inerente ad un evento |
| Accettazione segnalazione | Questa funzionalità permette ad un moderatore di accettare la segnalazione cancellando l’item segnalato e sospendendo l’utente |
| Rifiuto segnalazione | Questa funzionalità permette ad un moderatore di rifiutare una segnalazione |

**5.0 Glossario**

**LetsMeet:** Piattaforma che verrà sviluppata.

**Web-Based:** Il termine identifica un sistema basato sul web, quindi accessibile simultaneamente da più postazioni.

**Utente:** Qualsiasi attore che può beneficiare dei servizi del sistema.

**Moderatori:** Utenti speciali che possono che possono godere delle funzioni legate alla gestione delle segnalazioni.

**Amministratore:** Un attore speciale che po' godere di funzionalità limitate come Startup e Shutdown del sistema.

**RAD:** Documento di Analisi dei Requisiti.

**DBMS:** Sistema di gestione delle basi di dati.

**Database:** Insieme organizzato di dati persistenti.