**Controllo degli accessi e della sicurezza**

L’applicativo mobile YouthClub permette l’accesso a un solo tipo di persona cioè l’utente.

Ad ogni utente sono associati un id, utilizzato come credenziale per essere riconosciuto dal sistema, e una lista di recensioni dove possiamo risalire alle recensioni già effettuate.

Ogniqualvolta un utente abbia intenzione di utilizzare una funzionalità contenuta all’interno dell’applicazione, e quindi di avviare una nuova sessione, dovrà effettuare l’accesso all’app che in automatico risalirà al suo id tramite il codice IMEE del cellulare.

La sessione di utilizzo verrà terminata alla chiusura dell’app.

Il sistema, di conseguenza, può essere utilizzato da qualsiasi utente, le cui funzionalità disponibili sono descritte all’interno della matrice degli accessi.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Oggetto**  **Attori** | Gestione | | | |
| **Ricerca** | **Mappa** | **Lista Locali** | **Recensioni** |
| Utente | Visualizzazione;  Inserimento valori chiave;  Aggiornamento;  Possibilità di scelta;  Annullamento | Visualizzazione;  Annullamento; | Visualizzazione;  Interazione; | Visualizzazione;  Modifica;  Completamento;  Feedback utente;  Annullamento |

**Controllo del flusso globale del sistema**

Il controllo del flusso del software è garantito attraverso la realizzazione di classi Java che fanno da  
ricevitore per gli eventi rispondendo alle attivazioni dei client.  
A generare le richieste è il client, tale richiesta genera un evento e classe preposta a gestirlo si occupa di  
inizializzare le richieste e di inoltrarle per lo svolgimento dell’operazione che il client desidera portare a  
compimento. Una volta terminata l’operazione verrà prodotto un risultato.  
Il sistema software YouthClub è ibrido in quando sfrutta un modello client-server per quando riguarda le richieste inviate ed elaborate dal server , ma il server viene suddiviso in 3 layer Model-View-Control.

Il lato client si occupa di eseguire le richieste e decodificare e mostrare opportunamente le risposte nell’applicazione.

Il lato server presenta invece la logica e la conoscenza del dominio applicativo, il concetto di view diventa più generico e si occupa solo della formattazione dei dati in risposta alle richieste del client.

**Condizioni boundary**

**Avvio del sistema**Distinguiamo l’avvio dell’app “YouthClub” in lato server e in lato client:  
−Server: l’avvio avviene nel momento in cui viene avviato il server Tomcat dalla macchina su cui  
risiede;  
−Client: l’avvio avviene ogni volta che l’Utente accede all’app. Quando ciò avviene viene presentata un’interfaccia che permette all’utente di accedere alle funzionalità offerte dal sistema.

**Terminazione del sistema**−Client: la terminazione avviene al momento della chiusura dell’app da parte dello stesso. Quando ciò avviene vengono interrotte tutte le operazioni non concluse o pendenti che il Client stava tentando di eseguire , per mantenere la consistenza dei dati si concluderanno inevitabilmente con un fallimento.

**Fallimento del sistema**  
−Server: Il fallimento del sistema può avvenire per molteplici ragioni elencate di seguito:

1. Il sistema non conosce il luogo ma in quel momento nessuna delle 3 api esterne può fornire un risultato per down temporanei del servizio , ovviamente le probabilità che nessuno dei 3 servizi funzionano sono estremamente esigue.
2. Il sistema conosce il luogo ma MySQL smette di funzionare.
3. Altre problematiche potrebbero essere causate dalla rottura di qualche componente hardware e/o a sbalzi di corrente che causerebbero a una mancanza del servizio in maniera temporanea.

**Use Case Avvio**

|  |  |
| --- | --- |
| **Nome Use Case:** | Avvio |
| **Condizioni di entrata:** | Il ServerAdministrator accede al server |
| **Flusso di eventi:** | 1. Dopo aver effettuato correttamente l’accesso viene eseguito dal ServerAdministrator lo start-Up del server 2. Il sistema riceve la richiesta e attiva tutti i servizi necessari al corretto funzionamento del sistema. Fatto ciò sarà possibile usare il sistema. |
| **Condizioni di uscita:** | * Il Server è attivo e riceve una richiesta da parte del Client |

**Use Case Terminazione**

|  |  |
| --- | --- |
| **Nome Use Case:** | Terminazione |
| **Condizioni di entrata:** | Il ServerAdministrator esegue l’ operazione per arrestare il server |
| **Flusso di eventi:** | 1. Viene eseguito dal ServerAdministrator lo ShutDown del server 2. Il server riceve la richiesta e disattiva tutti i servizi usati dal sistema, inoltre esegue tutte le operazioni necessarie per una corretta terminazione |
| **Condizioni di uscita:** | * Il Server si arresta |

**Use Case Fallimento**

|  |  |
| --- | --- |
| **Nome Use Case:** | Failure luogo |
| **Condizioni di entrata:** | L’utente fa la richiesta |
| **Flusso di eventi:** | 1. Il sistema controlla se il luogo esiste già 2. Se non esiste effettua richiesta esterna al sistema |
| **Condizioni di uscita:** | * Risposta negativa da parte delle Api |

|  |  |
| --- | --- |
| **Nome Use Case:** | Failure database |
| **Condizioni di entrata:** | L’utente fa la richiesta |
| **Flusso di eventi:** | 1. L’utente ricerca il luogo |
| **Condizioni di uscita:** | * MySQL smette di funzionare ed emette un eccezione |