***Youth Club***

***Requirements Elicitation   
Versione 1.1***

****

***DATA: 27/12/2018***

**Coordinatore del progetto:**

|  |  |
| --- | --- |
| Nome | Matricola |
| Francesco Truono | 0512104830 |

**Partecipanti:**

|  |  |
| --- | --- |
| Nome | Matricola |
| Vincenzo Liguorino | 0512104692 |
| Aniello Petrosino | 0512104680 |
| Cristiana Elena Lazar | 0512104662 |

|  |  |
| --- | --- |
| **Scritto da:** | Tutto il team |

Indice

1. [Identify Actors and Use Cases](#Actor)
2. [Identify Relationships among Use Cases](#relati)
3. [Identify nonfunctional requirements](#nonfunc)
4. **IDENTIFY ACTOR AND USE CASES**

**Introduzione**

Abbiamo identificato i seguenti attori: l’applicazione (sia client che server) , i servizi esterni di ricerca(Google, Yelp e Foursquare) e l’utente.

L’applicazione si occupa di verificare i dati immessi dall’utente e impara dalle ricerche di questi ultimi mediante l’utilizzo dei servizi esterni di ricerca.

L’utente invece rappresenta colui che sta utilizzando l’applicazione e risulta identificato dal sistema.

Generalizzando gli scenari in casi d’uso possiamo sviluppare passi di situazioni concrete e considerando casi generali. Analizzando gli scenari di ricerca abbiamo individuato una suddivisione in più specializzazioni quali la scelta del metodo di ricerca

Abbiamo deciso di dividere le funzionalità relative all’utente in 3 casi d’uso : ricerca per nome, per località e per posizione (con l’ausilio del gps) tutte inizializzate dall’utente.

Nel caso di ricerca del nome è stato aggiunto un attore , applicazione , che si preoccupa di verificare il pattern della stringa di ricerca.

Partendo dagli scenari relative alle recensioni quali la stesura e la modifica delle recensioni, abbiamo diviso i compiti dell’utente specializzandosi in 3 ulteriori casi d’uso: stesura di una recensione, modifica della recensione dalla lista delle recensioni e dalla pagina del locale.

**Use Cases**

|  |  |
| --- | --- |
| **Nome Caso d’uso** | Seleziona tipo di ricerca |
| **Attori** | Utente |
| **Flusso di eventi** | 1. L’utente apre l’applicazione 2. All’utente si presenta una pagina contenente dei pulsanti che permettono di selezionare il tipo di ricerca che vuole effettuare 3. L’utente seleziona il tipo di ricerca |
| **Entry Condition** | * L’utente avvia l’applicazione |
| **Exit Conditions** | * L’utente ha selezionato il tipo di ricerca |

|  |  |
| --- | --- |
| **Nome Caso d’uso** | Stesura recensione |
| **Attori** | Utente |
| **Flusso di eventi** | 1. Il sistema mostra all’utente oltre alle informazioni del locale la possibilità di scrivere una recensione; 2. L’utente tramite la schermata di recensione compila la form con titolo e testo della recensione;    * Inserimento valutazione locale (1 pessimo, 5 ottimo);    * Valutazione qualità del servizio;    * Valutazione qualità del cibo/bevanda; 3. Il sistema valida la recensione e si accerta che non ci sono parole offensive\*; |
| **Entry Condition** | * L’utente ha selezionato un locale |
| **Exit Conditions** | * Il sistema accetta la recensione |
| **Exception** | * Nel punto 3 del flusso di eventi il sistema controlla se ogni campo della form è compilato correttamente, nel caso in cui questo non accade mostra all’utente i campi sbagliati marcandoli in rosso. |

|  |  |
| --- | --- |
| **Nome Caso d’uso** | Ricerca per nome |
| **Attori** | Utente |
| **Flusso di eventi** | 1. L’utente scrive il nome del locale nell’apposita area di testo 2. L’applicazione verifica se è stato rispettato il formato: *[Nome locale, Città]* 3. Il sistema verifica l’esistenza della città sul database. 4. Recupera la lista dei locali associati a quel nome 5. Restituisce il locale (potrebbero essere presenti più locali con lo stesso nome e si trovano nella stessa città |
| **Entry Condition** | * L’utente ha selezionato il tipo di ricerca per nome |
| **Exit Conditions** | * Il sistema ritorna il risultato della ricerca |
| **Eccezione** | * Nel punto 3 in caso di riscontro negativo , esegue caso d’uso “Apprendimento” |

|  |  |
| --- | --- |
| **Nome Caso d’uso** | Ricerca per posizione |
| **Attori** | Utente, GPS |
| **Flusso di eventi** | 1. L’utente chiede la posizione al Gps che gli ritorna la latitudine e longitudine, che verranno scritte nell’apposita area di testo. 1.1 L’utente seleziona una o più categorie : Enoteche, Discoteche , Bar o Pub 2. Il sistema verifica se il luogo è già presente sul database 3. Restituisce la lista dei locali in quella zona |
| **Entry Condition** | * L’utente ha selezionato il tipo di ricerca per posizione |
| **Exit Conditions** | * Il sistema ritorna il risultato della ricerca |
| **Exception** | * Nel punto 1 del flusso di eventi se il GPS è disattivato, il sistema chiede all’utente di attivarlo * Nel punto 2 in caso di riscontro negativo , esegue caso d’uso “Apprendimento” |

|  |  |
| --- | --- |
| **Nome Caso d’uso** | Apprendimento |
| **Attori** | Google API  Foursquare API  Yelp API |
| **Flusso di eventi** | 1. In caso di riscontro negativo, interroga Yelp, Google e Foursquare 2. Esegue il merge dei tre risultati ed esclude i duplicati 3. Salva le informazioni sul database |
| **Entry Condition** | * Il sistema verifica se il luogo è presente nel database |
| **Exit Conditions** | * Aggiornamento del database e restituzione dati |

|  |  |
| --- | --- |
| **Nome Caso d’uso** | Modifica della recensione dalla pagina del locale |
| **Attori** | Utente |
| **Flusso di eventi** | 1. L’utente accede alla pagina del locale utilizzando il caso d’uso di ricerca 2. Il sistema ordina le recensioni in modo tale da mostrare come prima recensione quella del utente 3. L’utente chiede al sistema di modificare la recensione desiderata. 4. L’utente apporta le modifiche desiderate. 5. Il sistema valida la recensione e si accerta che non ci siano parole offensive. |
| **Entry Condition** | * L’utente ha selezionato una recensione |
| **Exit Condition** | * Il sistema aggiorna la recensione |
| **Exception** | * il sistema può rifiutare la recensione o mantenere la precedente |

|  |  |
| --- | --- |
| **Nome Caso d’uso** | Modifica della recensione dalla lista delle recensioni |
| **Attori** | Utente |
| **Flusso di eventi** | 1. L’utente accede alla recensione dall’apposita area della lista di recensioni. 2. L’utente chiede al Sistema di modificare la recensione desiderata. 3. L’utente apporta le modifiche desiderate. 4. Il sistema valida la recensione e si accerta che non ci sono parole offensive |
| **Entry Condition** | * L’utente ha selezionato la recensione dalla lista delle recensioni |
| **Exit Condition** | * Il sistema aggiorna la recensione |
| **Exception** | * Il sistema può rifiutare la recensione o mantenere la precedente |

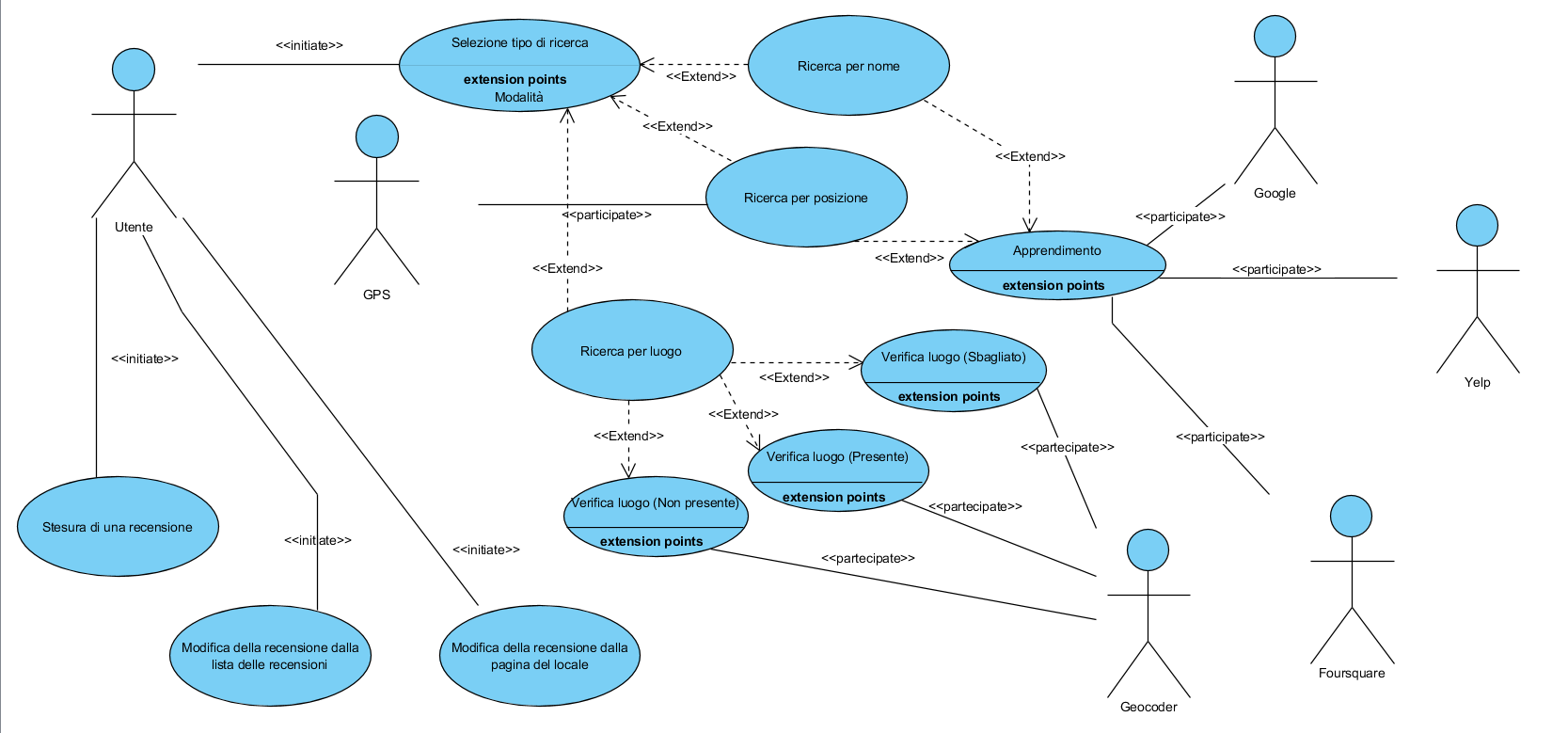
|  |  |
| --- | --- |
| **Nome Caso d’uso** | Ricerca per luogo |
| **Attori** | Utente, GPS |
| **Flusso di eventi** | 1. L’utente inserisce nell’apposita area di testo il luogo interessato. 1.1 L’utente seleziona una o più categorie : Enoteche, Discoteche , Bar o Pub 2. Il sistema verifica se il luogo esiste 3. Restituisce la lista dei locali in quella zona |
| **Entry Condition** | * L’utente ha selezionato il tipo di ricerca per luogo |
| **Exit Conditions** | * Il sistema ritorna il risultato della ricerca |
| **Exception** | * Nel punto 2 in caso di riscontro positivo che il luogo è già stato ricercato, esegue caso d’uso “Verifica luogo(Presente)” * Nel punto 2 in caso di riscontro negativo che il luogo inserito è sbagliato, esegue caso d’uso “Verifica luogo(Sbagliato)” * Nel punto 2 in caso di riscontro positivo che il luogo ancora non è stato ricercato, esegue caso d’uso “Verifica luogo(Non Presente)” |

|  |  |
| --- | --- |
| **Nome Caso d’uso** | Verifica luogo (Presente) |
| **Attori** | Geocoder API |
| **Flusso di eventi** | 1. L’utente in caso di riscontro positivo, interroga Geocoder API 2. Memorizza il luogo |
| **Entry Condition** | * Presa in input di un luogo |
| **Exit Condition** | * Il sistema scrive il luogo sul database |

|  |  |
| --- | --- |
| **Nome Caso d’uso** | Verifica luogo (Sbagliato) |
| **Attori** | Geocoder API |
| **Flusso di eventi** | 1. L’utente in caso di riscontro negativo, interroga Geocoder API 2. Restituisce che il luogo inserito è sbagliato |
| **Entry Condition** | * Presa in input di un luogo |
| **Exit Condition** | * Il sistema restituisce che è scorretto |

|  |  |
| --- | --- |
| **Nome Caso d’uso** | Verifica luogo (Non Presente) |
| **Attori** | Geocoder API |
| **Flusso di eventi** | 1. L’utente in caso di riscontro negativo, interroga Geocoder API 2. Trasforma il luogo in latitudine e longitudine 3. Salva informazioni sul database |
| **Entry Condition** | * Presa in input di un luogo |
| **Exit Condition** | * Il sistema memorizza i dati |

1. Identify Relationships among Use Cases



|  |  |
| --- | --- |
| **Selezione tipo di ricerca** | L’utente accedono all’applicazione ha la possibilità di scegliere il tipo di ricerca |
| **Stesura di una recensione** | L’utente ha la possibilità di scrivere una recensione sulla pagina del locale. |
| **Ricerca per nome** | L’utente può ricercare il locale scrivendo il nome e la città sfruttando il caso d’uso di “selezione tipo di ricerca” |
| **Ricerca per luogo** | L’utente può ricercare tutti i locali in una zona sfruttando Selezione tipo di ricerca |
| **Ricerca per posizione** | L’utente può ricercare tutti i locali vicino alla sua posizione sfruttando Selezione tipo di ricerca |
| **Apprendimento** | Il sistema impara nuovi locali |
| **Modifica della recensione dalla lista delle recensioni** | L’utente può modificare una recensione da lui scritta selezionando quella desiderata tra le presenti nella lista |
| **Modifica della recensione dalla pagina del locale** | L’utente può modificare la sua recensione, scritta in precedenza, sulla pagina del locale. |
| **Verifica luogo(Presente)** | Il sistema controlla il luogo |
| **Verifica luogo(Sbagliato)** | Il sistema controlla il luogo |
| **Verifica luogo(Non presente)** | Il sistema controlla il luogo |

1. Identify nonfunctional requirements

**Requisiti**

**Category Nonfunctional requirements**

**Usability** - Gli utenti per utilizzare l’applicativo possono registrarsi senza conoscenza preliminare del funzionamento di quest’ultimo.

**Reliability** -Quando un’azione viene interrotta a causa di un crash, il sistema dovrebbe essere in grado di riportare l’utente all’azione precedente al crash. Al massimo una ricerca potrebbe essere persa.

**Performance** -Il sistema deve essere in grado di parallelizzare le ricerche su tre diversi sistemi e memorizzarle per apprendere dalle precedenti ricerche degli utenti.

-L’applicazione deve avere tempi di risposta ristretti.

**Supportability** - (mantenibilità e adattabilità)

**Implementation** -YouthClub gira su tutti i dispositivi Android.  
 -Il server necessita di Java e Tomcat.