Documentazione Belmopoly

# Glossario

1 Glossario 2

2 Introduzione 3

2.1 Informazioni sul progetto 3

2.2 Abstract 3

2.3 Scopo 3

3 Analisi 4

3.1 Analisi del dominio 4

3.2 Analisi e specifica dei requisiti 4

3.2.1 Spiegazione elementi tabella dei requisiti: 10

3.3 Use Case 11

3.3.1 Use Case Main 11

3.3.2 Use Case Gioco 12

3.4 Pianificazione 13

3.4.1 Progettazione 14

3.4.2 Implementazione 14

3.5 Analisi dei mezzi 15

3.5.1 Software 15

3.5.2 Hardware 15

4 Progettazione 16

4.1 Design dell’architettura del sistema 16

4.2 Design delle interfacce 16

4.2.1 Interfaccia Login/Sign Up 16

4.2.2 Interfaccia Main 17

4.2.3 Interfaccia Room 18

4.2.4 Interfaccia Friends 18

4.2.5 Interfaccia User 19

4.2.6 Interfaccia Notifications 19

4.2.7 Interface Character 20

4.2.8 Interfaccia Game 20

4.3 Design procedurale 21

4.3.1 Diagrammi di flusso 21

4.3.2 Diagramma ER 21

5 Implementazione 22

6 Test 22

6.1 Protocollo di test 22

6.2 Risultati test 23

6.3 Mancanze/limitazioni conosciute 23

7 Consuntivo 23

8 Conclusioni 23

8.1 Sviluppi futuri 23

8.2 Considerazioni personali 23

9 Glossario 23

10 Bibliografia 24

10.1 Bibliografia per articoli di riviste: 24

10.2 Bibliografia per libri 24

10.3 Sitografia 24

11 Allegati 24

# Introduzione

## Informazioni sul progetto

* **Allievo:** Alex Gilardi
* **Allievo:** Alan Gregorio
* **Allievo:** Mark Iljins
* **Docente:** Michel Palucci
* **Scuola e classe:** SAMT Sezione informatica, I3AA
* **Materia:** Modulo 306
* **Data inizio consegna:** 29.01.2025 – 28.05.2025

## Abstract

## Scopo

Lo scopo principale del progetto è quello di realizzare un’applicazione WEB che permette agli utenti che lo utilizzano di divertirsi e giocare al famoso gioco Monopoly. Ovviamente per rendere il nostro applicativo migliore oltre ad avere le funzionalità base del gioco, che tutti conoscono, abbiamo anche aggiunto delle funzionalità extra.

Un ulteriore scopo del progetto è sicuramente quello di migliorare le nostre competenze nei vari linguaggi che andremo a utilizzare. Infatti, questo progetto ci permetterà di andare a esplorare ulteriormente la creazione e la struttura di DataBase tramite il linguaggio SQL. Ci permetterà anche di migliorare le nostre abilità nei linguaggi WEB come PHP, Javascript, HTML.

Infine, essendo il nostro primo progetto a gruppi, ci permetterà di applicare e migliorare quello che abbiamo acquisito durante lo svolgimento del primo progetto, svolto nel primo semestre.

# Analisi

## Analisi del dominio

Belmopoly è un applicativo WEB user-friendly molto facile da utilizzare, infatti possiede un design molto semplice e intuitivo, e non necessita di nessun requisito per essere utilizzato. L’applicativo è pensato per tutti gli utenti che vogliono divertirsi in compagnia giocando a un gioco da tavola famosissimo, cioè Monopoly, ma rimanendo online.

Attualmente ci sono già abbastanza prototipi di Monopoly Online in circolazione, però di quelli presenti quasi tutti hanno dei difetti abbastanza significativi, che rovinano l’esperienza di gioco all’utente.

Oltre alle funzionalità base, gli utenti avranno anche la possibilità di, personalizzare la propria pedina (Testa, Busto, Gambe), di chattare con gli altri giocatori durante partita e infine la possibilità di mettere all’asta una proprietà, dove tutti i giocatori possono offrire la propria somma di denaro.

Grazie a queste funzionalità aggiuntive, rende il nostro applicativo unico, e migliore da giocare in quanto tutti quelli già presenti posseggono solo le funzionalità base.

## Analisi e specifica dei requisiti

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-01 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Interfaccia Login Utente | | | | |
| **Note** | L’utente potrà effettuare l’accesso al suo account dell’applicativo. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Textbox per Password | | | | |
| **002** | Textbox per Email | | | | |
| **003** | Tasto Sign Up | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-02 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Interfaccia Sign Up Utente | | | | |
| **Note** | L’utente se non possiede già un account potrà crearne uno da questa interfaccia. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Textbox per Username | | | | |
| **002** | Textbox per Email | | | | |
| **003** | Textbox per Password | | | | |
| **004** | Tasto Login | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-03 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Interfaccia Main | | | | |
| **Note** | L’utente potrà visualizzare una sezione per creare una room, continuare una partita, e visualizzare una lista di amici + gli inviti | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Tasto per creare la Room | | | | |
| **002** | Tasto per continuare una partita | | | | |
| **003** | Tasto dove visualizzare amici e inviti | | | | |
| **004** | Tasto per logout | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-04 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Interfaccia Creazione Room | | | | |
| **Note** | L’utente potrà visualizzare una pagina dove poter creare una nuova room privata, dove poter invitare a giocare i vari amici. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | TextBox per il nome della room | | | | |
| **002** | Tasto per avviare la partita | | | | |
| **003** | Tasto per tornare alla MainPage4 | | | | |
| **004** | 4 Tasti per aggiungere i vari amici (basta anche solo 2 giocatori) | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-05 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Interfaccia con le Room | | | | |
| **Note** | L’utente potrà visualizzare una pagina dove poter continuare la partita che vuole. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Zona con tutte le partite da continuare | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-06 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Interfaccia Gioco | | | | |
| **Note** | L’utente potrà visualizzare tutti gli elementi necessari per giocare al Monopoli | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Zona a destra riguardante i vari player con il loro patrimonio e proprietà | | | | |
| **002** | Zona al centro dell’interfaccia la tavola di gioco con tutte le caselle. | | | | |
| **003** | Pulsante per il lancio dei dadi. | | | | |
| **004** | Pulsante per aprire la chat. | | | | |
| **005** | Pulsante per richiedere un trade. | | | | |
| **006** | Pulsante per richiedere di uscire dalla partita. | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-07 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Interfaccia Popup Casella | | | | |
| **Note** | L’utente quando finisce su una casella, essa verrà mostrata all’utente in mezzo all’interfaccia di gioco modi popup. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Pulsante Compra | | | | |
| **002** | Pulsante Asta | | | | |
| **003** | Zona con le informazioni riguardanti la proprietà | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-08 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Algoritmo Login | | | | |
| **Note** | L’utente può effettuare il login | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Gestione dell’utente. | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-09 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Algoritmo Logout | | | | |
| **Note** | L’utente può effettuare il logout | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Gestione dell’utente. | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-10 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Algoritmo Creazione Account | | | | |
| **Note** | L’utente può creare un nuovo utente se già non ne possiede uno | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Gestione nome utente | | | | |
| **002** | Gestione email | | | | |
| **003** | Gestione password | | | | |
| **004** | Gestione DB | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-11 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Algoritmo Eliminazione Account | | | | |
| **Note** | L’utente può eliminare il proprio account | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Gestione DB | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-12 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Algoritmo Gestione Soldi | | | | |
| **Note** | L’utente possiede un conto con un certo quantitativo di soldi con cui può comprare le varie proprietà o pagare gli altri in caso cada su una casella non sua. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Gestione soldi sufficienti | | | | |
| **002** | Gestione soldi no negativi | | | | |
| **003** | Gestione + 200 ogni giro | | | | |
| **004** | Gestione Bancarotta | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-13 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Algoritmo Tiro Dadi | | | | |
| **Note** | L’utente quando è il suo turno può tirare i dadi | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Gestione lancio | | | | |
| **002** | Zona dove ci sono i dadi | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-14 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Algoritmo Gestione Proprietà | | | | |
| **Note** | L’utente può comprare nuove proprietà, possederle e scambiare | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Gestione dell’acquisto della proprietà | | | | |
| **002** | Gestione del possesso della proprietà | | | | |
| **003** | Gestione per lo scambio di proprietà | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-15 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Algoritmo Gestione Prigione | | | | |
| **Note** | L’utente quando cade sulla cella “Go To Jail”, viene automaticamente portato alla cella della prigione senza passare per il via. Se possiede una carta “Jail Free” può uscire di prigione gratis. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Gestione cella “Go To Jail” | | | | |
| **002** | Gestione turni prigione | | | | |
| **003** | Gestione uscire di prigione | | | | |
| **004** | Gestione carta “Jail Free” | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-16 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Algoritmo Gestione Carte Eventi | | | | |
| **Note** | L’utente se cade su Imprevisti o Probabilità, gli verrà mostrata a schermo una carta di quel tipo. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Gestione carte imprevisti | | | | |
| **002** | Gestione carte probabilità | | | | |
| **003** | Gestione evento da eseguire | | | | |
| **004** | Gestione carte inventario | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-17 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Algoritmo Gestione Lista Amici | | | | |
| **Note** | L’utente può mandare una richiesta di amicizia a un altro utente e può anche rifiutare eventuali richieste. Può anche rimuovere dalla lista un amico. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Gestione invio richiesta amicizia | | | | |
| **002** | Gestione rifiuto richiesta amicizia | | | | |
| **003** | Gestione rimuovi amico | | | | |
| **004** | Gestione apri lista | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-18 | **Priorità** | 1 | **Versione** | 1.0 |
| **Nome** | Algoritmo Gestione Invita Amico | | | | |
| **Note** | L’utente mentre crea una partita può invitare a giocare i propri amici che ha nella lista amici. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Gestione invito a giocare | | | | |
| **002** | Gestione accetta invito a giocare | | | | |
| **003** | Gestione rifiuto invito a giocare | | | | |
| **004** | Zona nella pagina di creazione room dove poter mandare l’invito | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-19 | **Priorità** | 2 | **Versione** | 1.0 |
| **Nome** | Algoritmo Gestione Asta | | | | |
| **Note** | L’utente quando cade su una nuova proprietà, se decide di non comprarla, essa verrà messa all’asta, dove tutti i giocatori possono offrire la propria somma per comprarla. | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Gestione blocca partita | | | | |
| **002** | Gestione offerte player | | | | |
| **003** | Gestione timer offerta maggiore | | | | |
| **004** | Gestione aumento offerta | | | | |
| **005** | Gestione vincita asta | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-20 | **Priorità** | 3 | **Versione** | 1.0 |
| **Nome** | Algoritmo Gestione Chat Istantanea | | | | |
| **Note** | L’utente durante la partita può scriversi con gli altri giocatori in modo istantaneo nella chat situata in basso a destra | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Gestione invio messaggio | | | | |
| **002** | Gestione ricezioni messaggi | | | | |
| **003** | TextBox per scrivere il messaggio | | | | |
| **004** | Gestione apri chat | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisito** | REQ-21 | **Priorità** | 3 | **Versione** | 1.0 |
| **Nome** | Algoritmo Gestione Pedina Personalizzata | | | | |
| **Note** | L’utente può personalizzare la propria pedina prima di iniziare a giocare (Testa, Busto, Gambe) | | | | |
| **Sotto requisiti** | | | | | |
| **001** | Gestione selezione testa | | | | |
| **002** | Gestione selezione busto | | | | |
| **003** | Gestione selezione gambe | | | | |
| **004** | Salvataggio personaggio | | | | |

### Spiegazione elementi tabella dei requisiti:

**ID**: Identificativo univoco del requisito

**Nome**: Breve descrizione del requisito

**Priorità**: Indica l’importanza di un requisito nell’insieme del progetto, definita assieme al

committente. Ad esempio, poter regolare la qualità audio ha priorità minore rispetto al fatto di poter convertire il formato del file audio. Solitamente si definiscono al massimo di 2-3 livelli di priorità.

**Versione**: Indica la versione del requisito. Ogni modifica del requisito avrà una versione aggiornata.

**Note**: Eventuali osservazioni importanti o riferimenti ad altri requisiti.

**Sotto requisiti**: Elementi che compongono il requisito.

## Use Case

### Use Case Main

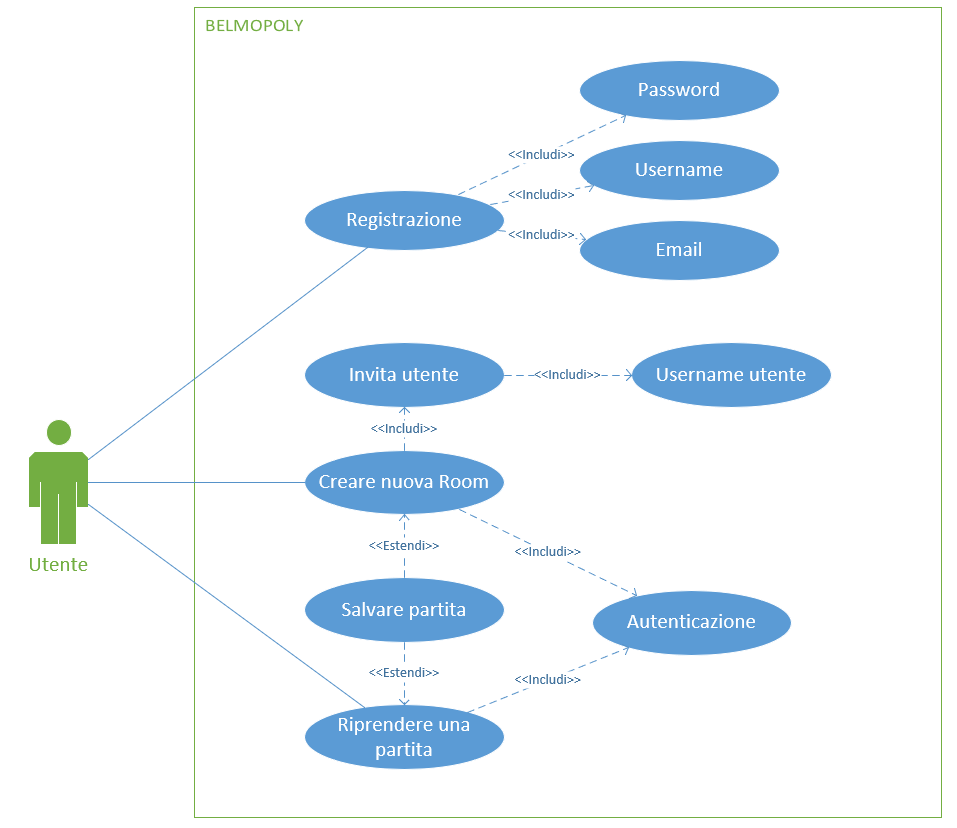


Figura 1 - Use Case Main

Questo è lo Use Case Main dell’applicativo WEB Belmpoly. Esso descrive il funzionante dell’applicazione senza la parte di gioco. Come si vede c’è solo un attore che utilizza l’applicativo WEB:

1. **Utente**: L’utente potrà svolgere principalmente tre azioni. La prima di tutte che andrà a fare se non possiede già un account nell’applicazione, è la registrazione. Ovviamente questa operazione non può essere completata se non si inserisce password, username e email.

Mentre la seconda azione che può svolgere, è quella di poter creare una nuova Room (partita). Ovviamente la Room non può essere creata se prima non si fa l’autenticazione, e inoltre bisogna per forza invitare almeno un giocatore, altrimenti non si può procedere con la creazione. L’invito del giocatore sarà svolto tramite username. In più l’utente ha anche la possibilità di salare la partita, esso non è obbligatorio.

L’ultima azione che può svolgere è quella di riprendere una partita già cominciata, come per la creazione, prima bisogna svolgere l’autenticazione, altrimenti non si può riprendere. Anche qui c’è la possibilità di salvare la partita, non obbligatorio.

### Use Case Gioco

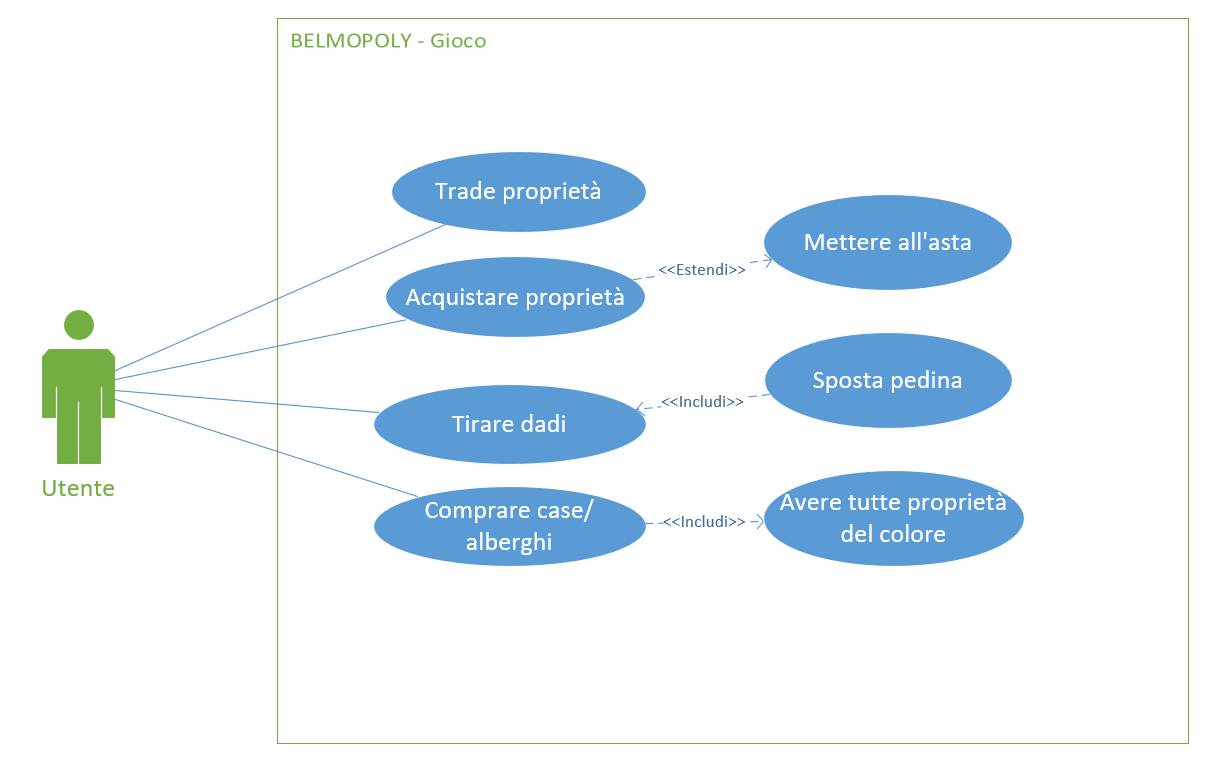


Figura 2 - Use Case Gioco

Questo è lo Use Case Gioco dell’applicativo WEB Belmpoly. Esso descrive la parte di gioco dell’applicazione. Come prima c’è solo un attore che utilizza l’applicativo WEB:

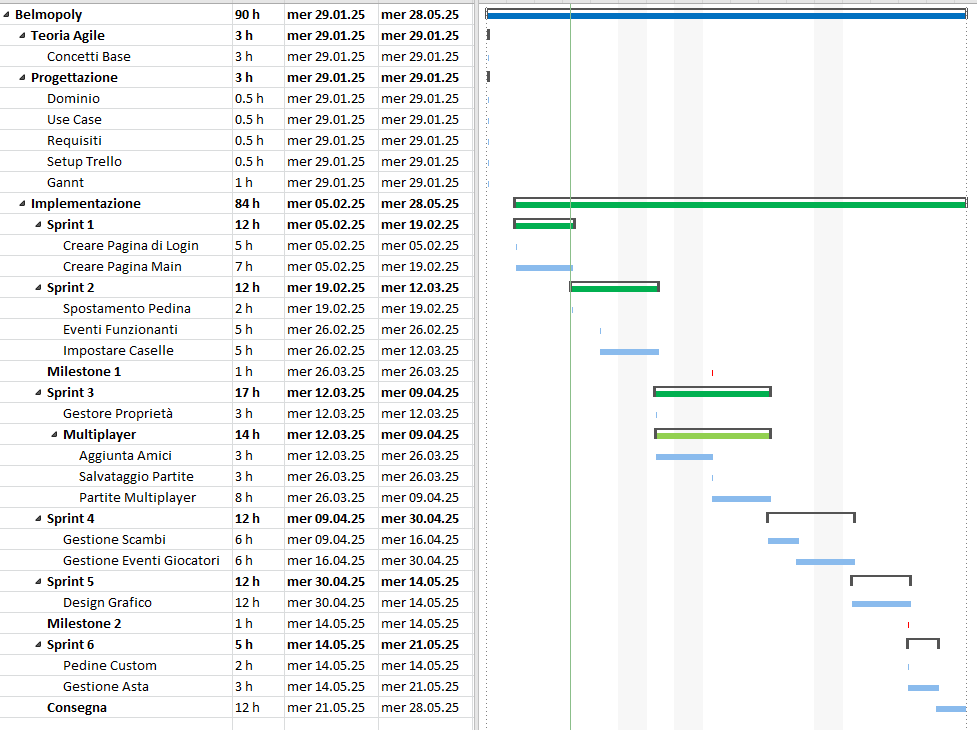
1. **Utente**: L’utente in questo caso può svolgere 4 azioni principali. La prima è quella di poter tirare i dadi, che automaticamente innesca l’attività secondaria cioè quella del movimento della pedina.

La seconda azione che può fare è quella di comprare nuove proprietà, questa azione viene estesa dal poter mettere all’asta la proprietà se si decide di non comprarla.

La terza attività che può andare a svolgere è il tiro dei due dadi, essa ovviamente include l’azione dello spostamento della pedina.

Mentre l’ultima azione che può andare a fare è quella di comprare case o alberghi per la proprietà in questione, ovviamente questa azione deve essere inclusa dal possedere tutte le proprietà dello stesso colore, altrimenti non può comprare nessuna delle due.

## Pianificazione



### Progettazione

La fase di progettazione è la fase più importante a inizio progetto, essa comprende tutte le attività necessarie per poi avviare la fase di implementazione. All’interno di questa fase abbiamo inserito 5 attività principali, e abbiamo previsto che ci porteranno via in totale 3 ore, cioè mezza giornata.   
In questa fase andremo a definire tutti i requisiti principali con la loro priorità, inoltre andremo a settare il setup di Trello[[1]](#footnote-1) con tutte le varie task da svolgere. Altre due attività che realizzeremo sono il Gannt stesso e lo Use Case dell’applicativo WEB. In fine andremo a realizzare tutti i diagrammi ER e di flusso.

### Implementazione

La fase di implementazione è la fase principale, infatti i porterà via all’incirca 84 ore, è quella che contiene tutte le attività per la realizzazione effettiva dell’applicativo WEB. Questa fase è suddivisa in 6 Sprint, che a loro volta al loro interno hanno attività differenti. In questa fase troviamo anche 2 Milestone cioè due punti importanti e fondamentali per il progetto.

#### Sprint 1

Lo sprint 1 sarà della durata di 12 ore, in questa fase andremo a creare la pagina di autenticazione e login, con già tutti i vari controlli e il salvataggio dei dati su DB. Inoltre andremo a creare la pagina principale dell’applicativo più la tavola di gioco.

#### Sprint 2

Lo sprint 2 è anche lui della durata di circa 12 ore, in questa fase andremo a completare le attività che non abbiamo finito nello sprint1, e che reputiamo necessarie finirle. In più andremo a svolgere le prime attività importanti per il funzionamento dell’applicativo, cioè lo spostamento della pedina all’interno della tavola, un piccolo evento per quando si capita su una cella, e in fine completare definitivamente la tavola da gioco con tutti i nomi e prezzi.

#### Sprint 3

Lo sprint 3 è leggermente più lungo degl’altri tre, dalla durata di circa 17 ore. Qui andremo a sviluppare l’algoritmo per la gestione delle proprietà, cioè tutte quelle funzionalità che permetto di comprare le proprietà, e di far pagare gli altri giocatori qualvolta ci finiscano sopra. In più andremo a sviluppare il multiplayer, cioè l’aggiunta di amici all’interno della partita e in fine il salvataggio della partita stessa su DB.

#### Sprint 4

Lo sprint 4 durerà all’incirca 12 ore, in questo tempo andremo a svolgere due attività molto importanti. La prima attività sarà quella di realizzare tutti gli algoritmi che permetteranno gli scambi di proprietà tra giocatori. Mentre la seconda cosa che andremo a fare saranno tutti gli algoritmi riguardante i vari eventi che può svolgere il giocatore, come il lancio dei dadi, la visualizzazione delle proprie proprietà ecc.

#### Sprint 5

Lo sprint 5 è quello che contiene meno attività, esso durerà sempre circa 12 ore. In questa fase andremo a migliorare e finire definitivamente il design dell’applicativo WEB. Esso verrà sviluppato basandosi sullo stile “Cyberpunk”, infatti ogni nome proprietà ha un nome riguardante questo tema.

#### Sprint 6

Lo sprint 6 è l’ultimo che andremo a svolgere, infatti sarò quello meno importante, in quanto contiene due attività di proprietà 3. In questa fase, se riusciremo con le tempistiche, andremo a implementare gli algoritmi per la personalizzazione della pedina e gli algoritmi per la gestione delle aste.

## Analisi dei mezzi

Per la realizzazione dell’applicativo WEB Belmopoly abbiamo utilizzato i PC fornito dalla Scuola Arti e Mestieri di Trevano con i software già installati al suo interno.

### Software

SDK, librerie, tools utilizzati per la realizzazione del progetto e eventuali dipendenze.

### Hardware

Su quale piattaforma dovrà essere eseguito il prodotto? Che hardware particolare è coinvolto nel progetto? Che particolarità e limitazioni presenta? Che HW sarà disponibile durante lo sviluppo?

# Progettazione

## Design dell’architettura del sistema

Descrive:

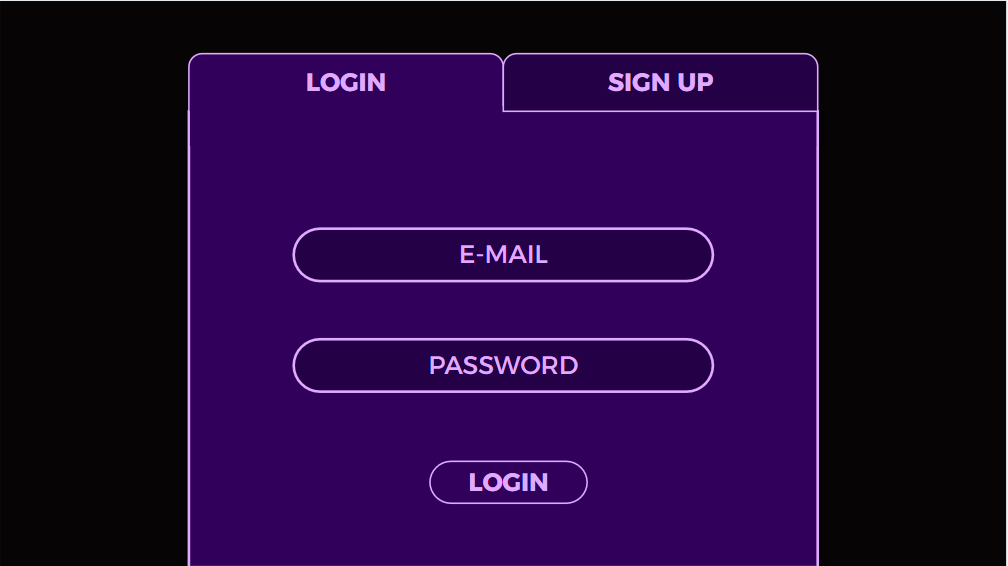
* La struttura del programma/sistema lo schema di rete...
* Gli oggetti/moduli/componenti che lo compongono.
* I flussi di informazione in ingresso ed in uscita e le relative elaborazioni. Può utilizzare *diagrammi di flusso dei dati* (DFD).
* Eventuale sitemap

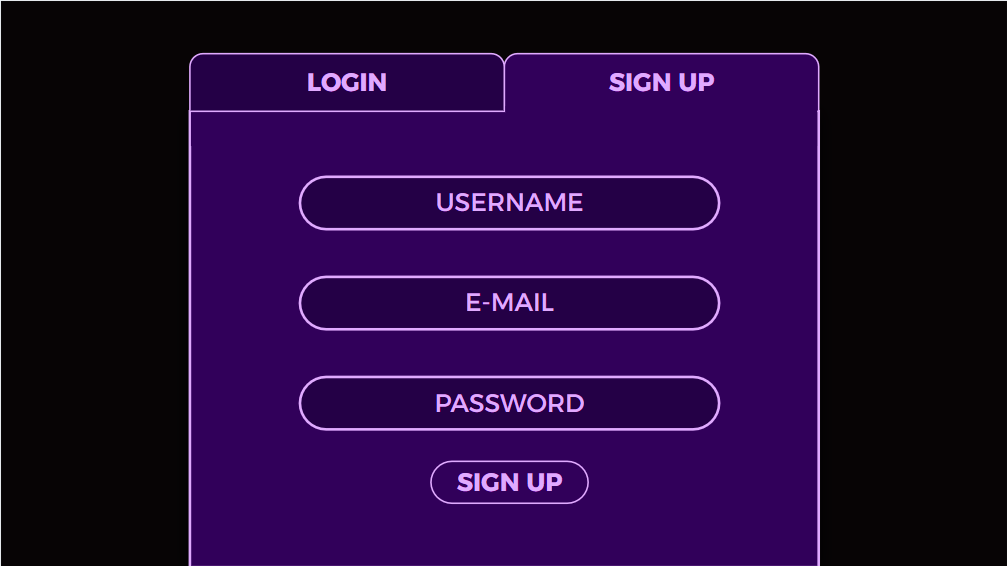
## Design delle interfacce

La progettazione delle interfacce è stata realizzata basandosi sulle informazioni ricavate durante la fase di analisi e le abbiamo progettate tramite il sito moqups[[2]](#footnote-2) .

Tutte le interfacce sono state pensate per essere utilizzate su Browser Web.

### Interfaccia Login/Sign Up

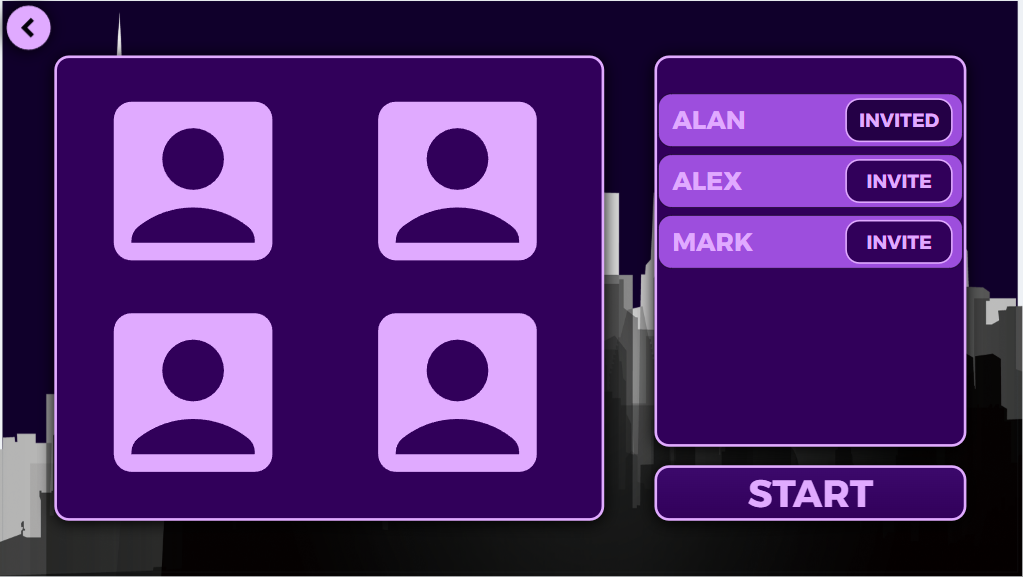




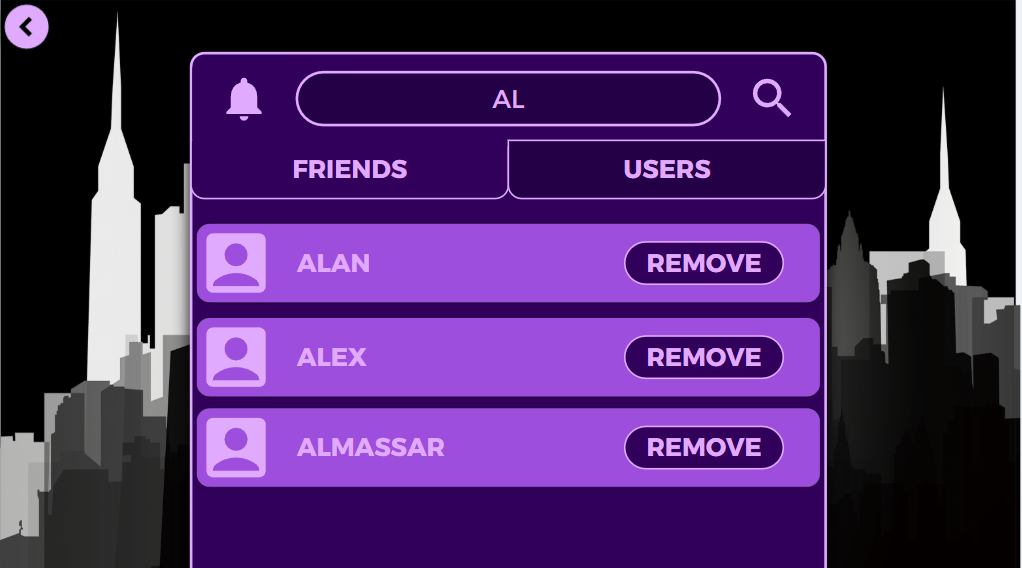
### Interfaccia Main



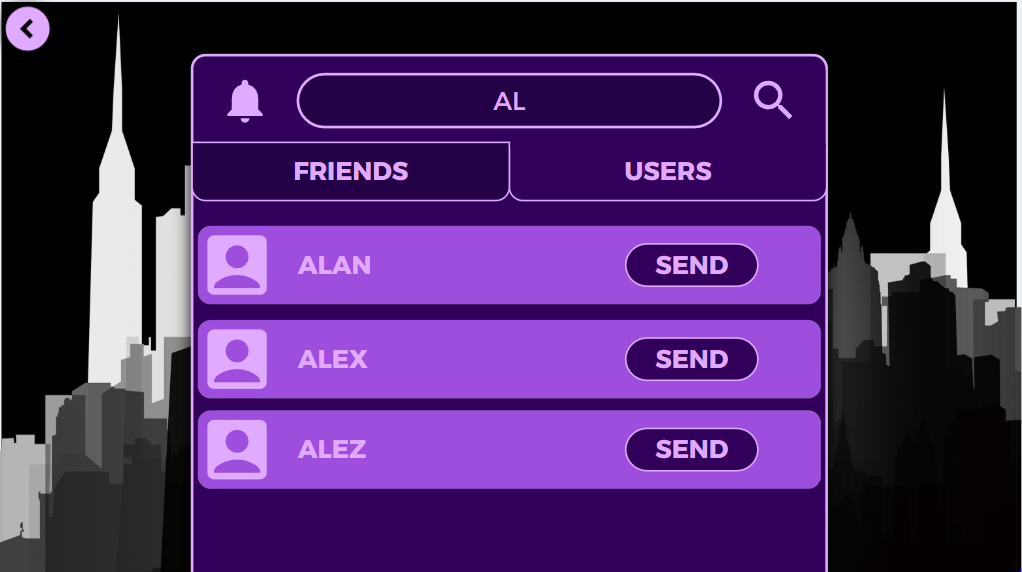
### Interfaccia Room



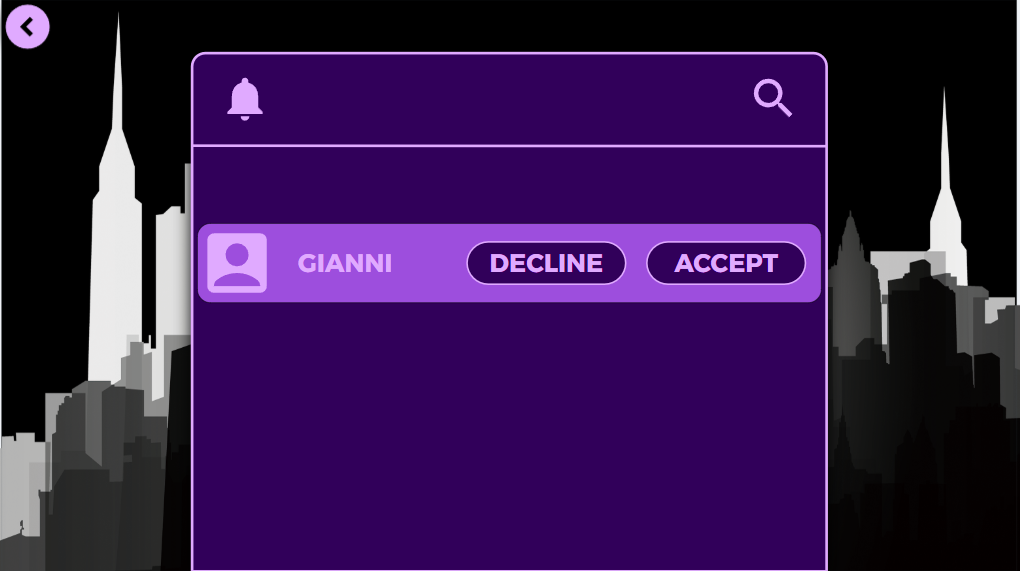
### Interfaccia Friends



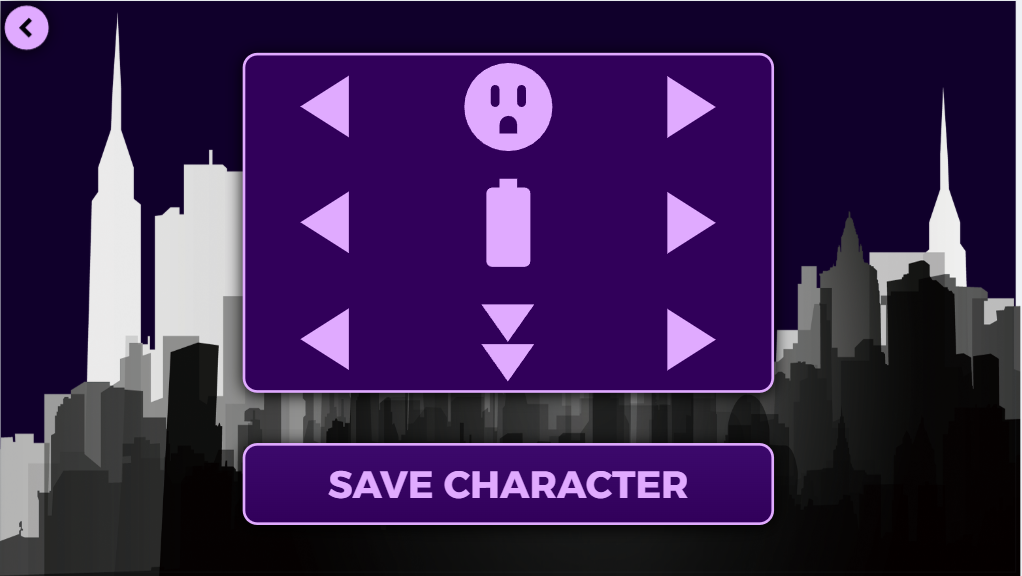
### Interfaccia User



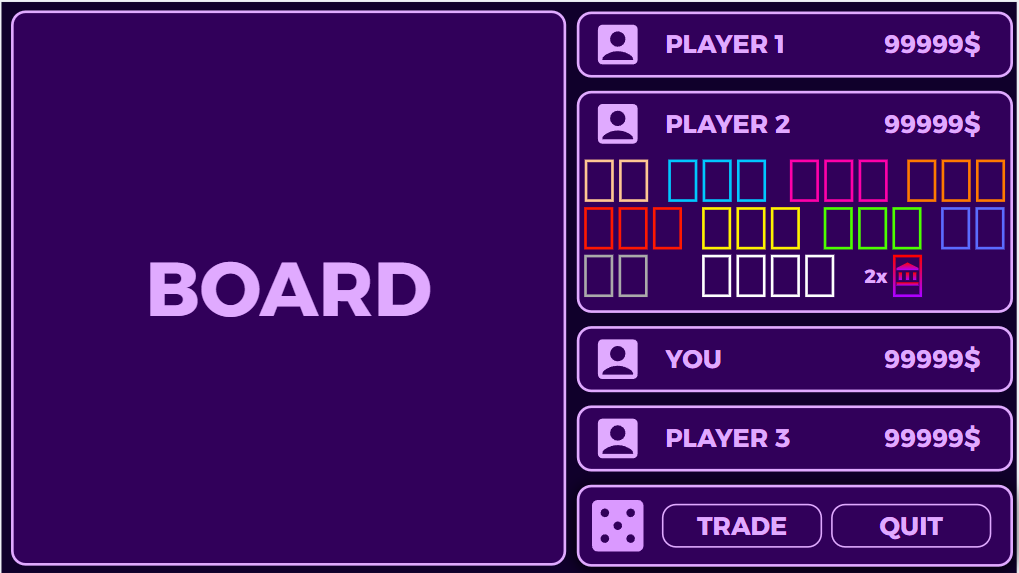
### Interfaccia Notifications



### Interface Character



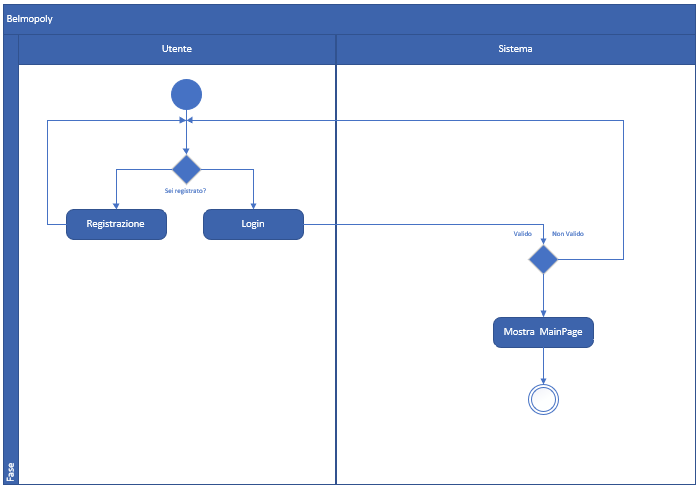
### Interfaccia Game



## Design procedurale

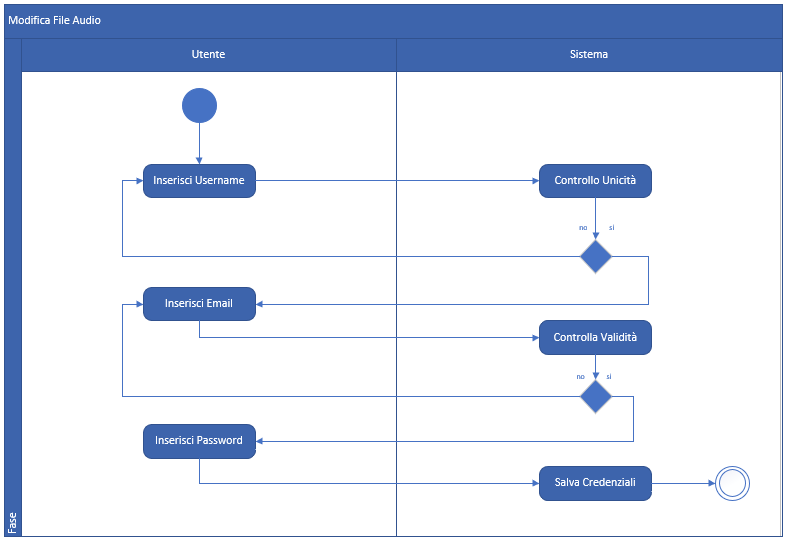
### Activity Diagram

#### Login



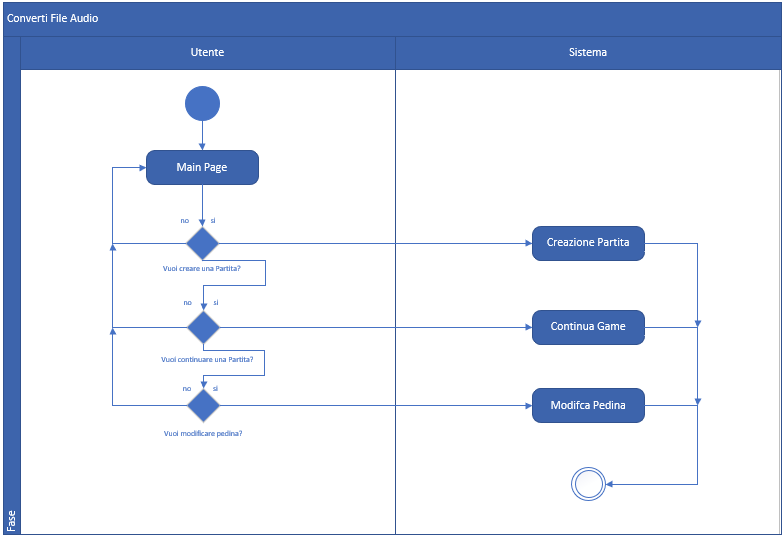
Questo è l’activity diagram per il login dell’applicazione WEB Belmopoly. Come si può vedere ha una struttura estremamente semplice, infatti inizialmente si chiede all’utente se è già registrato, se si va verso il login, altrimenti va verso la registrazione che poi riporterà al punto iniziale. Una volta passato il login il sistema controllerà se i dati inseriti sono validi.

#### Registrazione



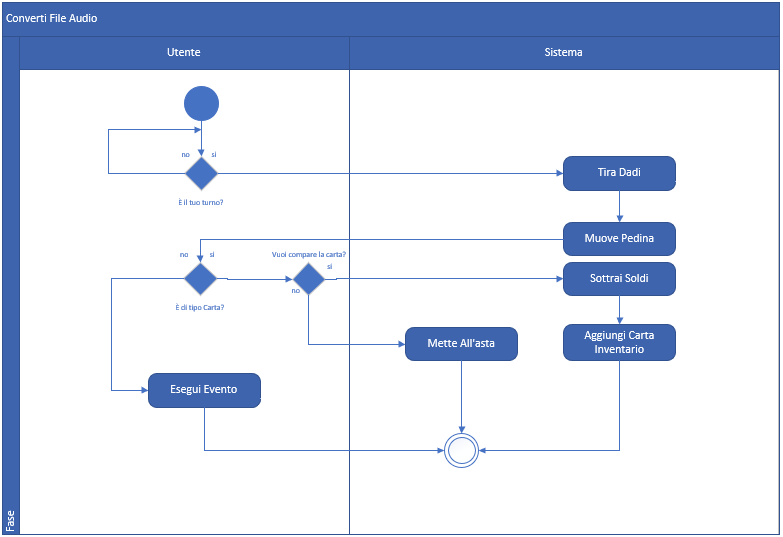
Questo è l’activity diagram per registrarsi sull’applicazione. Rispetto al diagramma di login esso ha una struttura leggermente più complessa, ma il concetto di funzionamento rimane semplice e intuitivo. Infatti vengono eseguite semplicemente due tipi di azione. Inserire il dato, che viene effettuata dall’utente, e controllare il dato inserito, che viene svolto dal sistema.

#### Globale



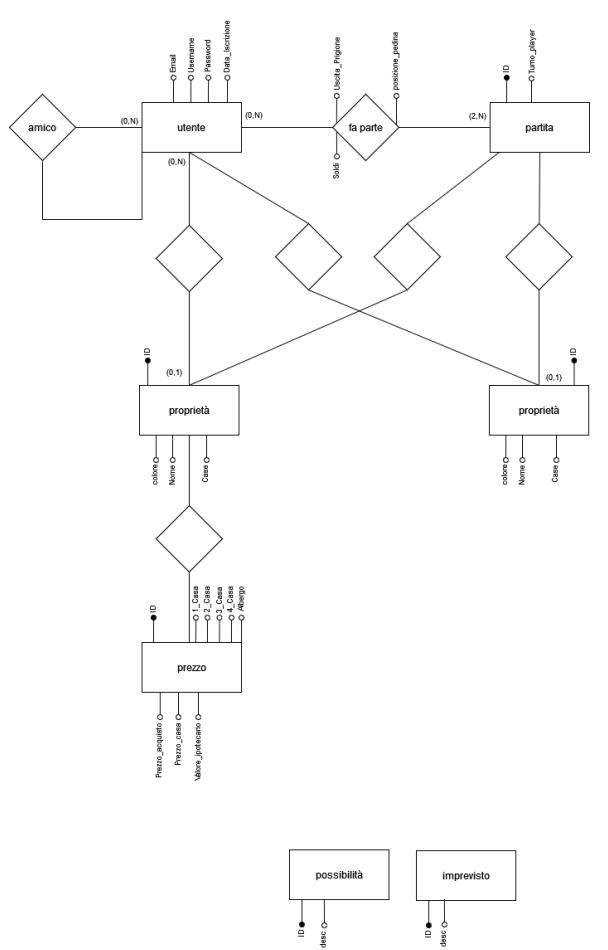
Questo è l’activity diagram che rappresenta tutte le azioni che si possono eseguire prima di iniziare una partita all’interno dell’applicazione. Come si può vedere è una struttura abbastanza complessa, però il concetto rimane semplice e intuitivo. L’utente una volta che si trova nella Main Page può decidere se creare una partita, continuare una partita, oppure modificare la propria pedina.

#### Gioco Effettivo



Questo è l’activity diagram che rappresenta il funzionamento del gioco effettivo, cioè quando l’utente si trova in partita. Esso è il diagramma con la struttura più complessa e anche a livello di concetto è più impegnativo rispetto agli altri. Come si può vedere c’è un controllo iniziale che “chiede” all’utente se il suo turno. Dopodiché ci sono tutte le azioni che permettono di giocare al classico Monopoli.

### Diagramma ER



Questo è il diagramma ER, esso rappresenta la struttura del DataBase che ho lo scopo di contenere tutti i dati riguardante l’applicazione WEB Belmopoly. Come si vede il DataBase è formato da 5 tabelle principali.

La tabella **utente** serve per salvare al suo interno le informazioni degli utenti registrati, perciò il nome utente, l’email e la password crittografata.

La tabella **partita** server per salvare tutti i dati che riguardano una specifica partita che è stata slavata per poi continuarla in seguito.

La tabella **proprietà** a sinistra serve per salvare tutte le proprietà libere quando si salava una partita per poi continuarla in seguito. A essa è pure collega la tabella che definisce il prezzo della proprietà e i prezzi delle varie case e dell’albergo.

# Implementazione

In questo capitolo dovrà essere mostrato come è stato realizzato il lavoro. Questa parte può differenziarsi dalla progettazione in quanto il risultato ottenuto non per forza può essere come era stato progettato.

Sulla base di queste informazioni il lavoro svolto dovrà essere riproducibile.

In questa parte è richiesto l’inserimento di codice sorgente - Print Screen - di maschere solamente per quei passaggi particolarmente significativi e/o critici.

Inoltre, dovranno essere descritte eventuali varianti di soluzione o scelte di prodotti con motivazione delle scelte.

Non deve apparire nessuna forma di guida d’uso di librerie o di componenti utilizzati. Eventualmente questa va allegata.

Per eventuali dettagli si possono inserire riferimenti ai diari.

# Test

## Protocollo di test

Definire in modo accurato tutti i test che devono essere realizzati per garantire l’adempimento delle richieste formulate nei requisiti. I test fungono da garanzia di qualità del prodotto. Ogni test deve essere ripetibile alle stesse condizioni.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-001  REQ-012 | **Nome:** | Import a card with KIC, KID and KIK keys, but not shown with the GUI |
| **Descrizione:** | Import a card with KIC, KID and KIK keys with no obfuscation, but not shown with the GUI | | |
| **Prerequisiti:** | Store on local PC: Profile\_1.2.001.xml (appendix n\_n) and Cards\_1.2.001.txt (appendix n\_n).  PIN (OTA\_VIEW\_PIN\_PUK\_KEY) and ADM (OTA\_VIEW\_ADM\_KEY) user right not set. | | |
| **Procedura:** | 1. Go to “Cards manager” menu,  in main page click “Import Profiles” link, Select the “1.2.001.xml” file, Import the Profile 2. Go to “Cards manager” menu,  in main page click “Import Cards” link, Select the “1.2.001.txt” file, Delete the cards,  Select the “1.2.001.txt” file, Import the cards 3. Research the “41795924770” Card, Click the imsi card link Check the card details 4. Execute the SQL: SELECT imsi, dir, keyset, cntr, rawtohex(kickey), rawtohex(kidkey), rawtohex(kikkey), rawtohex(chv), rawtohex(dap)FROM otacardkey a where imsi='340041795924770' ORDER BY keyset; | | |
| **Risultati attesi:** | Keys visible in the DB (OtaCardKey) but not visible in the GUI (Card details) | | |

## Risultati test

Tabella riassuntiva in cui si inseriscono i test riusciti e non del prodotto finale. Se un test non riesce e viene corretto l’errore, questo dovrà risultare nel documento finale come riuscito (la procedura della correzione apparirà nel diario), altrimenti dovrà essere descritto l’errore con eventuali ipotesi di correzione.

## Mancanze/limitazioni conosciute

Descrizione con motivazione di eventuali elementi mancanti o non completamente implementati, al di fuori dei test case. Non devono essere riportati gli errori e i problemi riscontrati e poi risolti durante il progetto.

# Consuntivo

Consuntivo del tempo di lavoro effettivo e considerazioni riguardo le differenze rispetto alla pianificazione (cap. 1.7) (ad esempio Gantt consuntivo).

# Conclusioni

Quali sono le implicazioni della mia soluzione? Che impatto avrà? Cambierà il mondo? È un successo importante? È solo un’aggiunta marginale o è semplicemente servita per scoprire che questo percorso è stato una perdita di tempo? I risultati ottenuti sono generali, facilmente generalizzabili o sono specifici di un caso particolare? ecc.

## Sviluppi futuri

Migliorie o estensioni che possono essere sviluppate sul prodotto.

## Considerazioni personali

Cosa ho imparato in questo progetto? ecc.

# Glossario

Inserite una semplice tabella con due colonne che spieghi i termini specifici del progetto (lista dei termini in ordine alfabetico A-Z)

Esempio:

|  |  |
| --- | --- |
| **Termine** | **Descrizione** |
| AJAX | **Asynchronous JavaScript And XML**: una tecnica che permette di eseguire richieste ed ottenere dati da una pagina web in modo asincrono. |
| CSS | **Cascading Style Sheets**: linguaggio che permette di definire il layout e la grafica di una pagina web. |

# Bibliografia

## Bibliografia per articoli di riviste:

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo dell’articolo (tra virgolette),
3. Titolo della rivista (in italico),
4. Anno e numero
5. Pagina iniziale dell’articolo.

## Bibliografia per libri

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo del libro (in italico),
3. ev. Numero di edizione,
4. Nome dell’editore,
5. Anno di pubblicazione,
6. ISBN.

## Sitografia

1. URL del sito (se troppo lungo solo dominio, evt completo nel diario),
2. Eventuale titolo della pagina (in italico),
3. Data di consultazione (GG-MM-AAAA).

**Esempio:**

* http://standards.ieee.org/guides/style/section7.html, *IEEE Standards Style Manual*, 07-06-2008.

# Allegati

Elenco degli allegati, esempio:

* Diari di lavoro
* Codici sorgente/documentazione macchine virtuali
* Istruzioni di installazione del prodotto (con credenziali di accesso) e/o di eventuali prodotti terzi
* Documentazione di prodotti di terzi
* Eventuali guide utente / Manuali di utilizzo
* Mandato e/o QdC
* Prodotto
* …

1. <https://trello.com/> [↑](#footnote-ref-1)
2. <https://app.moqups.com/> [↑](#footnote-ref-2)