|  |
| --- |
| http://www.fondazioneantoniodellamonica.com/images/logo_unisa.png  **Corso di Ingegneria del Software** |

****

**Docente:**

Andrea De Lucia

**Studenti:**

Raffaele Vitiello

Alessio Cuccurullo

Francesco Giuliano

**SYSTEM DESIGN DOCUMENT**

**SOMMARIO**

[**1.1** **Introduzione** 4](#_Toc534285054)

[**1.2 Obiettivi di design** 4](#_Toc534285055)

[**1.2.1 Criteri di performance** 5](#_Toc534285056)

[**1.2.2 Criteri di affidabilità** 5](#_Toc534285057)

[**1.2.3 Criteri di manutenzione** 6](#_Toc534285058)

[**1.2.4 Criteri per l’utente Finale** 6](#_Toc534285059)

[**1.3 Glossario** 6](#_Toc534285060)

[**1.4 Riferimenti** 6](#_Toc534285061)

[**1.5** **Panoramica** 6](#_Toc534285062)

[**2.** **Architettura del Software Corrente** 7](#_Toc534285063)

[**3.** **Architettura del Sistema Proposto** 8](#_Toc534285064)

[**3.1** **Panoramica** 8](#_Toc534285065)

[**3.2** **Decomposizione del sistema** 9](#_Toc534285066)

[**3.3** **Mapping Hardware/Software** 11](#_Toc534285067)

[**3.4** **Gestione dei dati persistenti** 13](#_Toc534285068)

[**3.5** **Controllo degli accessi e della sicurezza** 13](#_Toc534285069)

[**3.6** **Controllo del software globale** 13](#_Toc534285070)

[**3.7** **Condizioni boundary** 14](#_Toc534285071)

[**Scenari** 14](#_Toc534285072)

[**Casi d’uso** 15](#_Toc534285073)

[**4.** **Servizi dei Sottosistemi** 16](#_Toc534285074)

[**4.1.** **Gestione Utenti** 16](#_Toc534285075)

[**4.2.** **Gestione Storia** 17](#_Toc534285076)

[**4.3.** **Gestione Sessione** 18](#_Toc534285077)

[**5.** **Glossario** 19](#_Toc534285078)

# **Introduzione**

Riprendendo ciò che è stato riportato nel RAD:

Nei giochi di ruolo cartacei uno dei maggiori problemi per chi dirige il gioco è quello di dover appuntare con carta e penna ogni variazione che avviene durante lo sviluppo del gioco.

Il Moderatore, infatti, si trova molto spesso a improvvisare, vedendosi costretto a dover tener traccia di tutto ciò che viene detto e di ciò che ne consegue; da qui possono nascere problemi di incoerenza con quanto accaduto realmente nella sessione e con quello che sarebbe dovuto succedere: se non viene appuntato un cambiamento, si può andare incontro a dei lapsus potenzialmente pericolosi per il proseguirsi della storia. Di fatti spesso le sessioni di gioco vengono interrotte per poi essere riprese anche a giorni di distanza. Altro problema è la gestione dei numerosi personaggi non giocanti, la loro storia, le loro abilità eccetera.

Per quanto riguarda il Giocatore, uno dei problemi più gravi è, una volta iniziata la sessione di gioco, modificare la propria scheda personaggio, il quale si evolve di volta in volta diventando più (o meno) competente; infatti le schede cartacee subiscono nel tempo numerose modifiche che portano al facile deterioramento delle stesse. Un altro problema accade quando il Giocatore non modifica per niente la scheda per pigrizia o dimenticanza, in particolare per quanto riguarda l’inventario.

Per i giocatori novizi, la difficoltà maggiore è il primo approccio con la scheda del personaggio, in quanto è complicato capire intuitivamente come compilare la scheda sia la prima volta, sia per le successive.

## **1.2 Obiettivi di design**

Il sistema SineCharta deve poter essere il più efficiente ed intuitivo possibile. Tale efficienza sarà costruita attraverso rapidi tempi di risposta ad ogni genere di input ma anche differenti politiche di tolleranza all’errore. In più si punterà ad aver una buona manutenibilità attraverso il facile inserimento di nuove funzionalità. Sarà intuitivo attraverso agevoli interfacce. Tutto ciò ci porta a considerare quattro distinte classi: Performance, Affidabilità, Manutenzione ed Utente finale.

### **1.2.1 Criteri di performance**

|  |  |
| --- | --- |
| Tempo di risposta | SineCharta deve assicurare una risposta abbastanza rapida alle richieste dell’utente (pochi secondi). |
| Throughput | Il sistema allo stato attuale non è distribuito. Pertanto non vi è possibilità di gestire differenti richieste simultanee o concorrenti. |
| Memoria | La quantità di memoria che verrà utilizzata da SineCharta non può essere stimata precisamente. In principio, il sistema dovrà essere sottoposto alla memorizzazione di almeno: 20 Moderatori, 60 giocatori. |

### **1.2.2 Criteri di affidabilità**

|  |  |
| --- | --- |
| Robustezza | SineCharta utilizzerà notifiche di errore run-time per gestire eventuali input errati, senza interrompere il funzionamento del sistema o il flusso interattivo dell’utente. |
| Disponibilità | SineCharta deve essere disponibile all’uso, 24 ore su 24, da parte degli utenti, grazie ad un server sempre attivo. |
| Tolleranza all’errore | SineCharta deve essere in grado di operare anche in presenza di condizioni di errore, gestendo eventuali malfunzionamenti. |
| Sicurezza | L’accesso al sistema è controllato da un sistema di autenticazione che categorizza gli utenti non permettendo l’accesso a parti del sistema non autorizzate. |

### **1.2.3 Criteri di manutenzione**

|  |  |
| --- | --- |
| Estendibilità | SineCharta per eventuali nuove funzionalità future, deve garantire l’estendibilità del sistema. Un’aggiunta non deve intaccare le varie funzionalità già presenti. Il codice scritto deve essere ben strutturato e di facile comprensione. |
| Modificabilità | In caso di comparsa di bug, il codice deve essere pienamente comprensibile, così da facilitare operazioni di modifica del sistema. |
| Leggibilità | Le parti del codice di SineCharta saranno facili da leggere ed interpretare grazie ad un’opportuna indentazione e aggiunta di commenti su ogni funzionalità. |
| Tracciabilità dei requisiti | Ad ogni requisito sarà dedicato un opportuno modulo, così da rendere i requisiti facili da tracciare ed estendere. |

### **1.2.4 Criteri per l’utente Finale**

|  |  |
| --- | --- |
| Usabilità | SineCharta permette all’utente di non avere nessuna difficoltà nell’interazione con il sistema, presentando delle interfacce semplici e molto intuitive anche per utenti novizi al gioco. Non mancheranno, inoltre, degli aiuti da parte del sistema per aiutare l’utente durante l’interazione con esso. |

## **1.3 Glossario**

* SineCharta : nome del sistema che si va a sviluppare.
* RAD : Requirement Analysis Document.

## **1.4 Riferimenti**

* Rad

# Bernd Bruegge, Allen H. Dutoit Object-Oriented Software Engineering: Using UML, Patterns and Java.

* 1. **Panoramica**

Prima di parlare dell’architettura, è importante fare un accenno alle attività di system design  
che costituiscono le fondamenta per l’architettura software del sistema.

* Decomposizione del sistema, in cui il sistema viene suddiviso in diversi sottosistemi  
  ognuno dei quali, a sua volta, è caratterizzato da servizi che offre ad altri sottosistemi.

L’insieme dei servizi sarà denominato Interfaccia.

* Mapping Hardware/Software, riguardante la scelta della configurazione hardware del  
  sistema, la comunicazione tra nodi, il come vengano incapsulati i servizi di un  
  sottosistema.
* Gestione dei dati persistenti, nel quale di individuano gli oggetti che devono essere  
  resi persistenti e quale genere di infrastruttura si deve usare per memorizzare tali  
  oggetti.
* Politiche di accesso e Sicurezza, che ci aiuta a rappresentare tramite delle tabelle le  
  operazioni ed informazioni utilizzabili da ogni singolo attore.
* Controllo del software globale, che ci guida su quali operazioni eseguire ed in che  
  ordine, per garantire il corretto flusso di controllo del sistema.
* Condizioni Boundary, che includono oltre l’avvio e lo shutdown anche la gestione dei  
  fallimenti dovuti all’invecchiamento del sistema, interruzione di corrente o anche a  
  errori di progettazione.

1. **Architettura del Software Corrente**

In commercio non è presente un sistema cosi concepito, dunque non è possibile descrivere l’architettura Software esistente.

1. **Architettura del Sistema Proposto**
   1. **Panoramica**

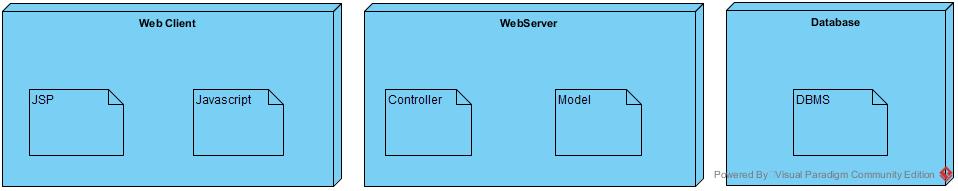
Il tipo di sistema proposto è un’applicazione web, composta da un architettura client/server. Tale sistema deve rispondere alle richieste effettuate da parte degli utenti, in base alla tipologia dell’utente che lo utilizza. Nel caso di un utente moderatore il sistema dovrà interagire con esso mostrando le storie e le sessioni di gioco attive, mentre per un utente giocatore il sistema interagirà con esso mostrando informazioni relative alle proprie sessioni di gioco e i personaggi completi di tutte le statistiche aggiornate.

Le motivazioni che hanno portato ad una scelta di un architettura client/server sono le seguenti:

* **Portabilità:** il sistema essendo un’applicazione web potrà essere utilizzata su diversi dispositivi e macchine differenti.
* **Performance:** il sistema garantisce tempi di risposta rapidi, ma comunque essendo un’applicazione web i tempi saranno in dipendenza della connessione di rete.
* **Scalabilità:** il sistema è in grado di supportare e gestire diverse richieste da parte di molti utenti contemporaneamente collegati all’applicazione.
* **Flessibilità:** il sistema, basato su due tipologie di utenza, visualizza due interfacce simili e diverse funzionalità specifiche per tipologia d’utente.
* **Affidabilità:** le componenti sia client che server garantiscono l’affidabilità anche in presenza di guasti e situazioni impreviste, quindi deve essere possibile effettuare dei backup periodici al database.

Per la realizzazione del sistema è stata utilizzata un’architettura MVC, poiché l’applicazione si occupa di gestire direttamente i dati, la logica e le funzionalità. I tre ruoli principali dell’applicazione sono:

* Web Client
* WebServer
* Database



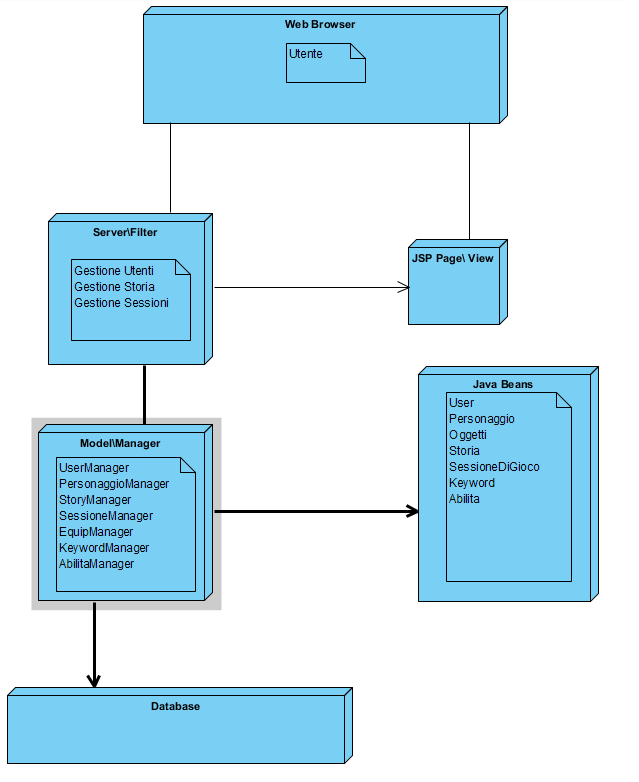
Il WebClient rappresenta l’interfaccia che permette all’utente di interagire con il sistema, ricoprendo il ruolo di client in quanto utilizza un browser per richiedere pagine web al WebServer.

Il WebServer ha il compito di elaborare i dati da inviare al client. Spesso interroga il database, tramite il Database per accedere ai dati persistenti.

Il Database mantiene i dati sensibili del sistema, utilizzando un DBMS, riceve inoltre le varie richieste dal WebServer inoltrandole al DBMS e restituendo i dati richiesti.

* 1. **Decomposizione del sistema**

Viene di seguito riportato un diagramma generale e la descrizione di ogni modulo:

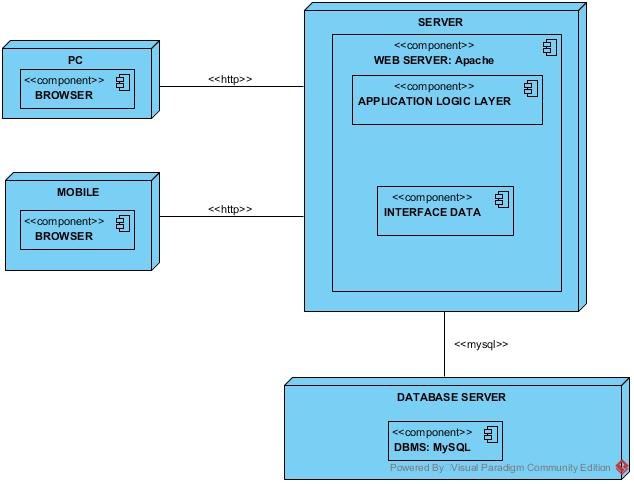
****

|  |  |
| --- | --- |
| Web Browser | |
| UtenteModeratore | Modulo che si occupa di gestire le funzionalità del sistema riservate ad UtentiModeratori, come la scrittura delle storie, la creazione delle sessioni, gestire le sessioni di gioco, invitare giocatori ad una storia, gestire gli scontri e NPC. |
| UtenteGiocatore | Modulo che si occupa di gestire le funzionalità del sistema riservate ad UtentiGiocatori, come creare un PG, accettare inviti ricevuti, partecipare ad una storia e partecipare ad una sessione attiva. |

|  |  |
| --- | --- |
| Servlet\Filter | |
| Gestione Utenti | Modulo che si occupa di gestire le funzionalità del sistema riservate alla gestione degli utenti;  permette quindi di registrare un nuovo utente, consente ad un utente di cambiare la propria e-mail o la propria password, di recuperare le proprie credenziali di accesso, permette infine l’autenticazione dell’utente nel sistema. |
| Gestione Storie | Modulo che si occupa di gestire le funzionalità del sistema riservate alla gestione delle Storie;  consente la creazione di una Storia, gestire gli inviti per i giocatori, permette la creazione delle Keyword e la creazione del personaggio. Permette agli utenti giocatori la creazione dei PG e infine, consente ad un UtenteGiocatore di partecipare ad una Storia. |
| Gestione Sessioni | Modulo che si occupa di gestire tutte le funzionalità riservate alla gestione della Sessione di gioco; permette ad un UtenteModeratore di creare ed avviare una sessione, consultare le sessioni attive, rivedere le Storie scritte in precedenza, consultare le keyword, modificare la storia corrente, visualizzare gli ordini di chiamata dei giocatori, estrarre i tarocchi e la gestione degli NPC. Mentre un UtenteGiocatore può gestire la propria scheda PG ed estrarre i tarocchi durante le Sessioni di gioco. |

|  |  |
| --- | --- |
| Database | |
| Dati persistenti | Modulo che si occupa di memorizzare dati in  memoria, in modo da poter essere prelevati e  modificati in modo corretto. |

* 1. **Mapping Hardware/Software**

****

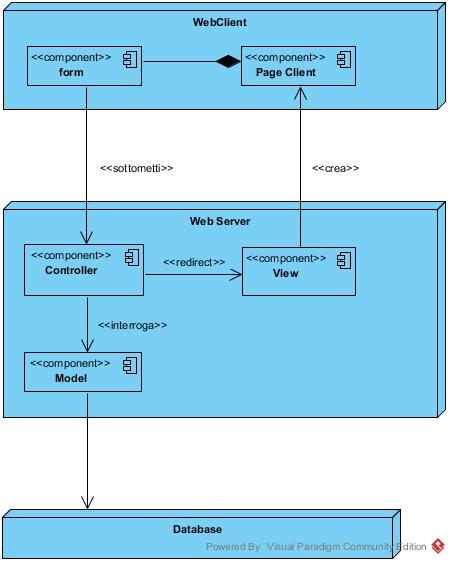
La struttura del sistema di Sine Charta è formata da un server centrale e dal browser di qualsiasi sistema operativo

**Web Server**Il server utilizzato è Apache Tomcat versione 9.0.

**Application Logic layer**Il sistema, e quindi le sue funzionalità, sono implementate tramite linguaggio Java e tramite l’utilizzo di servlet, il view è preposto alla visualizzazione della pagina HTML tramite JSP.

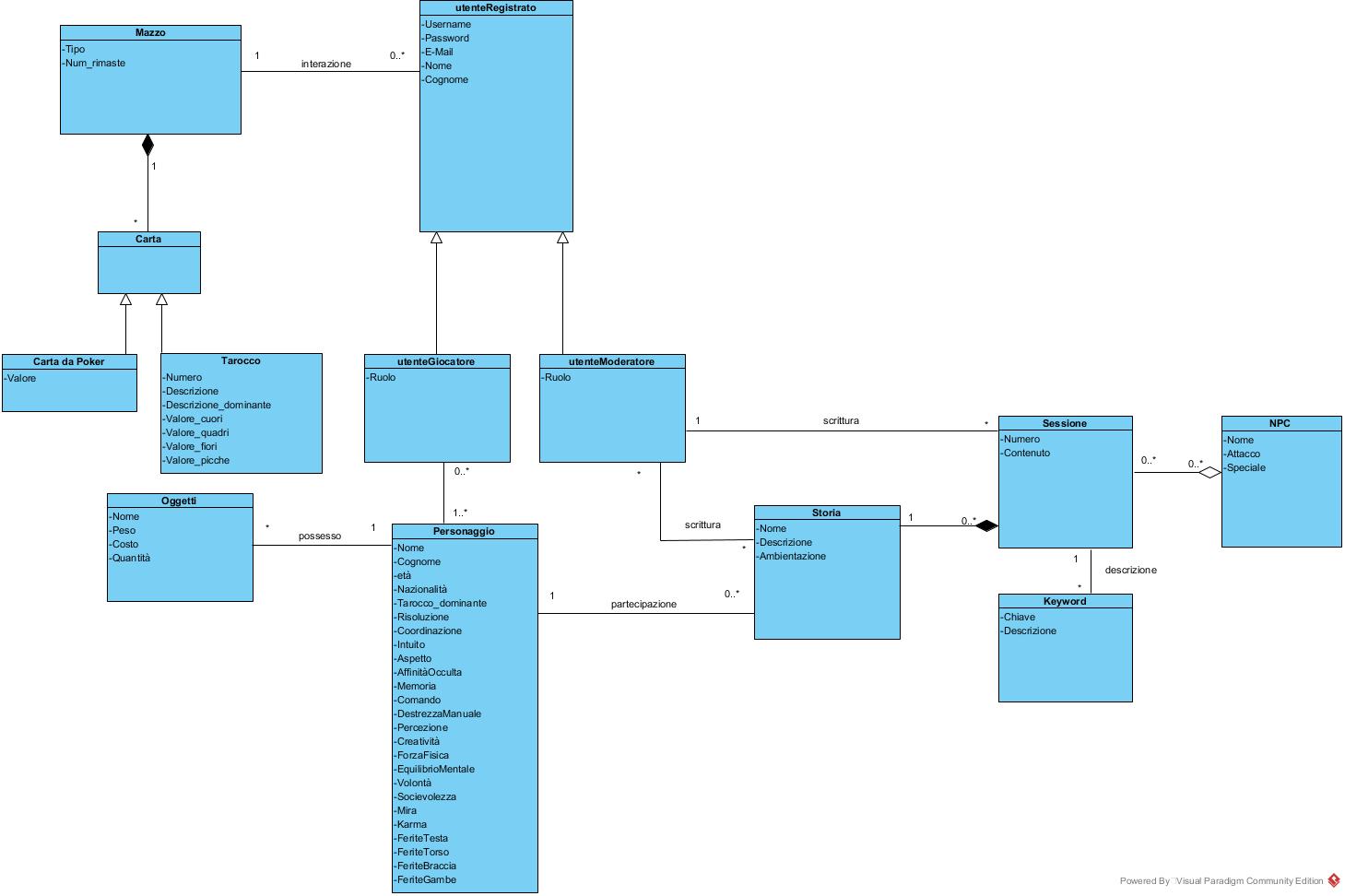
**Storage layer**Rappresenta il collegamento con il server da parte del sistema e si occupa di tutte le richieste  
di accesso e modifiche sui dati permanenti presenti nel database.

**Database Server**Il DBMS usato è MySQL il quale presenta molte API che permettono l’interazione tra sistema  
e database.

****

* 1. **Gestione dei dati persistenti**

La memorizzazione dei dati è stata gestita con l’uso di un DataBase di tipo relazionale, in quanto, oltre allo spazio di archiviazione richiesto, i database consentono di ottenere un veloce tempo di risposta (query di ricerca), garantendo una gestione multiutente ed una gestione tramite DBMS.  
Questa scelta è stata fortemente pensata data la sicurezza offerta da un DBMS di ultima generazione, insieme ad una maggiore affidabilità, c’è la garanzia di coerenza, facilità di gestione, nonché dalla velocità di accesso e trasmissione dei dati.  
Riportiamo in seguito lo schema generale e successivamente le singole tabelle con i relativi campi ed una breve descrizione. Infine, prima di questi vi è il class diagram generale riproposto nel RAD con una breve descrizione delle corrispondenze che hanno portato alla creazione del database.

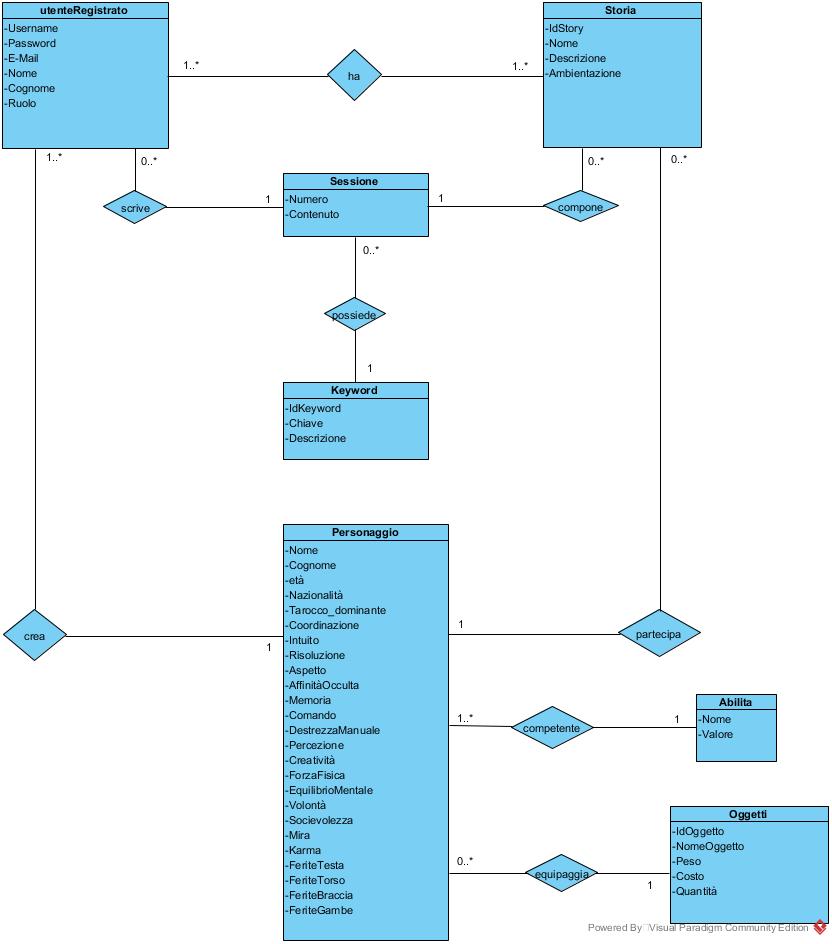


* La classe utente, che può rappresentare o un utenteGiocatore o un utenteModeratore, è rappresentato dalla tabella utenteRegistrato, dove il ruolo ne chiarisce la tipologia. Nella tabella sono. memorizzare le informazioni dell’utente relative a username, password, e-mail, nome e cognome; queste sono memorizzate nella stessa tabella utenteRegistrato associato ad ogni utenteGiocatore o utenteModeratore.
* Una Storia è scritta da un utenteModeratore la quale è composta da diverse Sessioni e da alcune keyword. Essa è composta da un nome, da una descrizione e dall’ambientazione in cui viene svolta la Storia. Le informazioni relative ad una Storia sono mantenute all’interno della tabella Storia.
* Un utenteModeratore scrive anche le Sessioni, le quali sono identificate da un numero progressivo e dal contenuto che fornisce la descrizione di tutto quello che succede all’interno della sessione. Ad ogni sessione può far parte un certo numero di Nemici e ad ogni sessione può essere collegata uno o più keyword. Queste informazioni sono contenute all’interno della tabella Sessioni.
* Un Nemico che prende parte ad una Sessione è identificato da un nome, ha un certo valore di attacco, una descrizione e un potere speciale. Tutte queste informazioni sono all’interno della tabella Nemico.
* Un utenteGiocatore per poter partecipare ad una Storia deve necessariamente creare un Personaggio, abbreviato PG. Il PG è identificato dalle sue Generalità, ha un Tarocco dominante, delle caratteristiche, delle abilità, una risoluzione e quando partecipa alla Sessione, può subire delle ferite durante uno scontro. Dunque, queste informazioni sono mantenute all’interno della tabella Personaggio.
* Ogni PG è in possesso di un numero imprecisato di Oggetti, identificati da un Id e definiti da nome, che li identifica, il peso e il costo. Gli oggetti possono essere delle armi, definite dal tipo, modello, dal danno che fanno, dall’abilità, dal numero di munizioni e dalla ricarica.
* Un utenteRegistrato, sia come nella figura di utenteModeratore che di utenteGiocatore, può interagire con dei Mazzi, composti da Carte, le quali possono essere Tarocchi o Carte da Poker, il Mazzo è descritto dal tipo e dal numero di carte rimaste; le Carte da Poker sono caratterizzate dal loro valore, mentre i Tarocchi sono definiti dalla descrizione, dal valore delle carte e dalla descrizione dominate. Tutte queste informazioni sono contenute nelle tabelle: Mazzo, Carta, Carta da Poker e Tarocco.

## **3.5 Schema Entity-Relationship**



In fasi di ristrutturazione abbiamo risolto tutte le generalizzazioni, accorpando le classi “utenteModeratore” e “utenteGiocatore” nella tabella “utenteRegistrato” la quale si differenzierà tramite l’attributo ruolo, che è un enumeratore che può essere “utenteModeratore” o “utenteGiocatore”. Inoltre, abbiamo deciso di rendere l’attributo della classe “Personaggio” ‘attributo’, una tabella, poiché abbiamo bisogno di tenere traccia delle abilità del Personaggio in maniera consistente ed efficace. Di seguito l’ER creato con Visual Paradigm versione definitiva.



|  |  |
| --- | --- |
| UtenteRegistrato | |
| Campo | **Vincoli** |
| Username | Lunghezza massima: 15 caratteri, unique  Chiave primaria |
| Password | Lunghezza massima: 15 caratteri, not null |
| E-Mail | Lunghezza massima: 20 caratteri, not null, unique |
| Nome | Not null |
| Cognome | not null |
| Ruolo | Enumeratore: utenteModeratore/utenteGiocatore |

|  |  |
| --- | --- |
| Personaggio | |
| Campo | **Vincoli** |
| Nome | Lunghezza massima: 30 caratteri, |
| Cognome | Lunghezza massima: 30 caratteri, |
| Età | Not null |
| Nazionalità | Not null |
| Tarocco\_dominante | Lunghezza massima: 15, not null |
| Intuito | Intero not null |
| Aspetto | Intero not null |
| Coordinazione | Intero not null |
| AffinitàOcculta | Intero not null |
| Memoria | Intero not null |
| Comando | Intero not null |
| DistanzaDallaMorte | Intero not null |
| DestrezzaManuale | Intero not null |
| Percezione | Intero not null |
| Creatività | Intero not null |
| ForzaFisica | Intero not null |
| EquilibrioMentale | Intero not null |
| Volontà | Intero not null |
| Socievolezza | Intero not null |
| Mira | Intero not null |
| Karma | Intero not null |
| Risoluzione | Intero not null |
| Salute | Intero not null |
| FeriteTesta | Lunghezza massima: 5 caratteri,not null |
| FeriteTorso | Lunghezza massima: 5 caratteri,not null |
| FeriteBraccia | Lunghezza massima: 5 caratteri,not null |
| FeriteGambe | Lunghezza massima: 5 caratteri,not null |
| Username (chiave primaria di utenteRegistrato) | Lunghezza massima: 15 caratteri, unique  Chiave primaria |
| IdStory (chiave primaria di Storia) | Intero, chiave primaria |

|  |  |
| --- | --- |
| Storia | |
| Campo | **Vincoli** |
| IdStory | Intero, chiave primaria |
| Titolo | Lunghezza massima: 50 caratteri, unique  Chiave primaria |
| Descrizione | Lunghezza massima: 500 caratteri, not null |
| Ambientazione | Enumeratore: Terre perdute/Quarto Reich/Soviet/Sanctum Imperum , not null |

|  |  |
| --- | --- |
| Sessione | |
| Campo | **Vincoli** |
| Numero | Intero, unique, Chiave primaria |
| Contenuto | not null |
| Username (chiave primaria di utenteRegistrato) | not null, chiave primaria |
| IdStory (chiave primaria di Storia) | Intero, Chiave primaria |

|  |  |
| --- | --- |
| Keyword | |
| Campo | **Vincoli** |
| Id | Intero, unique, Chiave primaria |
| Chiave | Lunghezza massima: 50 caratteri |
| Descrizione | Lunghezza massima:500 caratteri |
| Numero (chiave primaria di Sessione) | Intero, unique, Chiave esterna |

|  |  |
| --- | --- |
| Oggetti | |
| Campo | **Vincoli** |
| IdOggetto | Intero not null |
| NomeOggetto | Lunghezza massima: 30 caratteri, unique  Chiave primaria |
| Peso | Double not null |
| Costo | Double not null |
| Quantità | Not null |
| Username (chiave primaria di utenteRegistrato) | Lunghezza massima: 15 caratteri, unique  Chiave esterna |
| IdStory (chiave primaria di Storia) | Intero, chiave esterna |

|  |  |
| --- | --- |
| Abilità | |
| Campo | **Vincoli** |
| Nome | Lunghezza massima: 15 caratteri, unique  Chiave primaria |
| Valore | Intero |
| Username (chiave primaria di utenteRegistrato) | Lunghezza massima: 15 caratteri, unique  Chiave esterna |
| IdStory (chiave primaria di Storia) | Intero, chiave esterna |

|  |  |
| --- | --- |
| Ha (relazione tra utenteRegistrato e Storia) | |
| Campo | **Vincoli** |
| Username (chiave primaria di utenteRegistrato) | Not null, chiave esterna |
| IdStory (chiave primaria di Storia) | Intero, unique, Chiave esterna |
| Flag | TinyInt (0,1) |

* 1. **Controllo degli accessi e della sicurezza**

SineCharta è un sistema multi-utente, ci sono diversi attori che hanno il permesso di eseguire diverse operazioni su vari insiemi di oggetti. Per schematizzare al meglio il controllo degli accessi abbiamo suddiviso per tipologia di utente le azioni consentite, al fine di ottenere una visione più compatta e dettagliata grazie ad una matrice degli accessi riportata di seguito:

|  |  |  |
| --- | --- | --- |
| Attori  Oggetti | utentModeratore | utenteGiocatore |
| Autenticazione | ✓ LogIn  ✓ LogOut  ✓ Visualizza\_Profilo  ✓ Modifica\_Dati\_Personali  ✓ Recupera\_Password  ✓ Recupera\_Username | ✓ LogIn  ✓ LogOut  ✓ Visualizza\_Profilo  ✓ Modifica\_Dati\_Personali  ✓ Recupera\_Password  ✓ Recupera\_Username |
| Registrazione | ✓ Registrazione | ✓ Registrazione |
| Storie | ✓ Editor\_Storia  ✓ Invia\_Inviti  ✓ Accetta\_Inviti  ✓ Crea\_PG  ✓ Gioca | X Editor\_Storia  X Invia\_Inviti  ✓ Accetta\_Inviti  ✓ Crea\_PG  ✓ Gioca |
| Sessioni | ✓ Gestione\_Mazzo  ✓ Gestione\_PG  ✓ Editor\_Sessione  ✓ Gestione\_Sessione  ✓ Gestione\_NPC | ✓ Gestione\_Mazzo  ✓ Gestione\_PG  X Editor\_Sessione  X Gestione\_Sessione  X Gestione\_NPC |

* 1. **Controllo del software globale**

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

* 1. **Condizioni boundary**

Le condizioni limite hanno a che vedere con l’accensione e lo spegnimento del sistema per quanto riguarda il lato Server, mentre dal lato Client si riferiscono agli errori di connessione al server.

**Scenari**

|  |  |
| --- | --- |
| Nome Scenario | Startup Server |
| Istanze di Attori Partecipanti | Antonio: Admin |
| Flusso di eventi | 1. Antonio decide di voler avviare il sistema e quindi clicca sul pulsante “Avvia”. 2. Il sistema, con le opportune procedure di avvio, attiva i server e i relativi servizi in remoto rendendosi disponibile ad eventuali rischiste. 3. Il sistema notifica il successo della procedura. |

|  |  |
| --- | --- |
| Nome Scenario | Shutdown Server |
| Istanze di Attori Partecipanti | Antonio: Admin |
| Flusso di eventi | 1. Antonio decide di voler arrestare il sistema, quindi accede alla pagina dedicata e clicca sul pulsante “Arresta”. 2. Il sistema effettua una scansione per verificare se ci sono ancora richieste in sospeso. 3. Il sistema porta a termine le eventuali richieste in sospeso. 4. Tramite le opportune procedure di arresto il sistema disattiva i servizi in remoto e il server. 5. Il sistema notifica il successo della procedura. |

**Casi d’uso**

|  |  |
| --- | --- |
| ID | UC\_Startup |
| Nome Caso Uso | Startup Server |
| Attori Pertecipanti | Admin |
| Entry Condition | L’amministratore accede al sistema |
| Flusso di eventi | UTENTE SISTEMA   1. Admin accede al sistema e clicca sul pulsante “Avvia” 2. SineCharta accende il server e attiva i servizi in remoto rendendosi disponibile per le richieste notificando il successo dell’operazione all’utente. |
| Exit Condition | Il server è attivo e i relativi servizi sono disponibili |
| Eccezioni | Errore Startup |

|  |  |
| --- | --- |
| ID | UC\_Shutdown |
| Nome Caso Uso | Shutdown Server |
| Attori Pertecipanti | Admin |
| Entry Condition | L’amministratore accede al sistema |
| Flusso di eventi | UTENTE SISTEMA   1. Admin accede al sistema e clicca sul pulsante “Spegni” 2. SineCharta effettua una scansione per verificare se ci sono eventuali richieste in sospeso, porta a termine le richieste e avvia le procedure di arresto. Il sistema notifica il successo dell’operazione all’utente. |
| Exit Condition | Il server si spegne correttamente |
| Eccezioni | Errore Shutdown |

1. **Servizi dei Sottosistemi**
   1. **Gestione Utenti**

|  |  |
| --- | --- |
| Sottosistema | Gestione Utenti |
| Descrizione | Sottosistema che gestisce la registrazione di un utente a SineCharta, l’autenticazione di tutti gli utenti e le operazioni necessarie alla loro gestione. |
| Servizi offerti | |
| Servizio | **Descrizione** |
| Registra Utente | Permette di inserire un utente nel database. |
| LogIn | Permette ad un utente di poter effettuare l’accesso al sistema. |
| LogOut | Permette ad un utente di uscire dal sistema. |
| Recupero Credenziali | Permette ad un utente di poter recuperare il proprio username e la propria password. |
| Modificare informazioni personali | Permette ad utente di modificare la propria password e la propria email. |

* 1. **Gestione Storia**

|  |  |
| --- | --- |
| Sottosistema | Gestione Storia |
| Descrizione | Sottosistema che gestisce la creazione delle storie, la loro modifica, la creazione della keyword, la creazione di un personaggio, l’invio degli inviti e la partecipazione degli utentiGiocatori alle storie. Tutto ciò che viene creato o modificato deve essere poi salvato all’interno del database |
| Servizi offerti | |
| Servizio | **Descrizione** |
| Editor Storia | Permette la creazione e la modifica di una storia e di salvarla all’interno del database. |
| Invitare utentiGiocatori alla Storia | Permette ad utentiModeratori di invitare utentiGiocatori alla loro storia. |
| Accettare un invito | Permette ad utentiGiocatori di accettare gli inviti ricevuti. |
| Creare un PG | Permette ad utentiGiocatori di creare un PG e di salvarlo all’interno del database. |
| Gioca | Permette ad un utenteGiocatore di iniziare a giocare. |

* 1. **Gestione Sessione**

|  |  |
| --- | --- |
| Sottosistema | Gestione Sessione |
| Descrizione | Sottosistema che gestisce la creazione della sessione, la modifica, consultare le keyword, gestire gli npc e i PG, visualizzare l’ordine di chiamata degli scontri ed estrarre i tarocchi. |
| Servizi offerti | |
| Servizio | **Descrizione** |
| Editor Sessione | Permette la creazione e la modifica di una Sessione e il suo salvataggio all’interno del database. Inoltre, permette ad un utenteModeratore di aggiungere una keyword. |
| Gestione Sessione | Permette di avviare una sessione e visualizzarne il contenuto, keyword. Inoltre, carica anche i dati dello scontro e l’ordine di chiamata. |
| Gestione Mazzo | Permette ad un utenteModeratore o ad un utenteGiocatore di estrarre Carte o di mischiare un mazzo. |
| Gestione NPC | Permette ad un utenteModeratore di gestire un NPC. |
| Gestione PG | Permette ad un utenteGiocatore di gestire un PG. |