**System Design**

# **Introduzione**

## **Scopo del sistema**

TomMASO è una piattaforma **semplice e user-friendly** che permette agli utenti registrati di creare un proprio spazio tramite l’utilizzo di un’intuitiva interfaccia grafica. La piattaforma permette agli utenti di creare blog, scrivere, modificare i propri articoli (tramite l’utilizzo del linguaggio **Markdown**) e di interagire con gli altri utenti registrati.

* 1. **Obiettivi del sistema**

La piattaforma è stata progettata con i seguenti obiettivi:

* + 1. **Criteri di usabilità**

Il sistema deve essere fruibile al maggior numero di persone, pertanto si auspica l’utilizzo di un’interfaccia user-friendly e un editor che consenta anche a chi si approccia per la prima volta al linguaggio Markdown di creare i propri articoli sul blog. Inoltre, deve essere fornito agli utenti il manuale per l’utilizzo.

* + 1. **Criteri di performance**

Frequenti rallentamenti della piattaforma potrebbero scoraggiare l’utilizzo da parte di alcuni utenti, pertanto si vuole garantire che il servizio sia prestante e stabile per almeno 200 utenti contemporaneamente. Il server dovrebbe garantire un uptime di almeno il 95%.

* + 1. **Criteri di affidabilità**

Una failure durante la gestione di una richiesta effettuata da un client non deve compromettere il funzionamento del resto del sistema e continuare a servire gli altri utenti. Il sistema deve essere protetto dalle vulnerabilità più comuni quali XSS e SQL-injection in quanto potrebbero compromettere la sicurezza degli utenti e portare al furto di dati sensibili.

Le password non devono essere salvate in chiaro in quanto esistono utenti che riutilizzano le password su altre piattaforme e di conseguenza la loro sicurezza sarebbe compromessa anche su altri siti.

* + 1. **Criteri di portabilità**

Il sistema deve essere installabile su tutti i dispositivi con sistemi Windows, MacOS e Linux per una futura distribuzione del sistema, pertanto si deve utilizzare un sistema di virtualizzazione.

* 1. **Definizioni, acronimi e abbreviazioni**

**RAD** - Requirement Analysis Document

**MVC** - Model View Control

**JSP** - Java Servlet Page

* 1. **Riferimenti**

Requirement Analysis Document di TomMASO

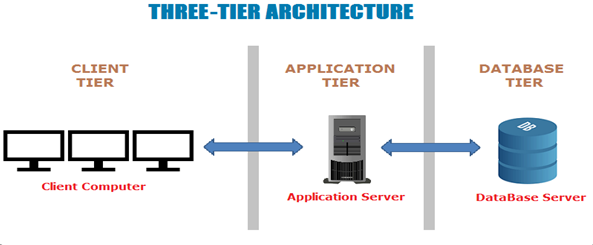
* 1. **Panoramica**

Il documento mostra i dettagli della progettazione della piattaforma TomMASO. Verranno analizzati nel dettaglio i sottosistemi che compongono

# **Architettura Software Proposta**

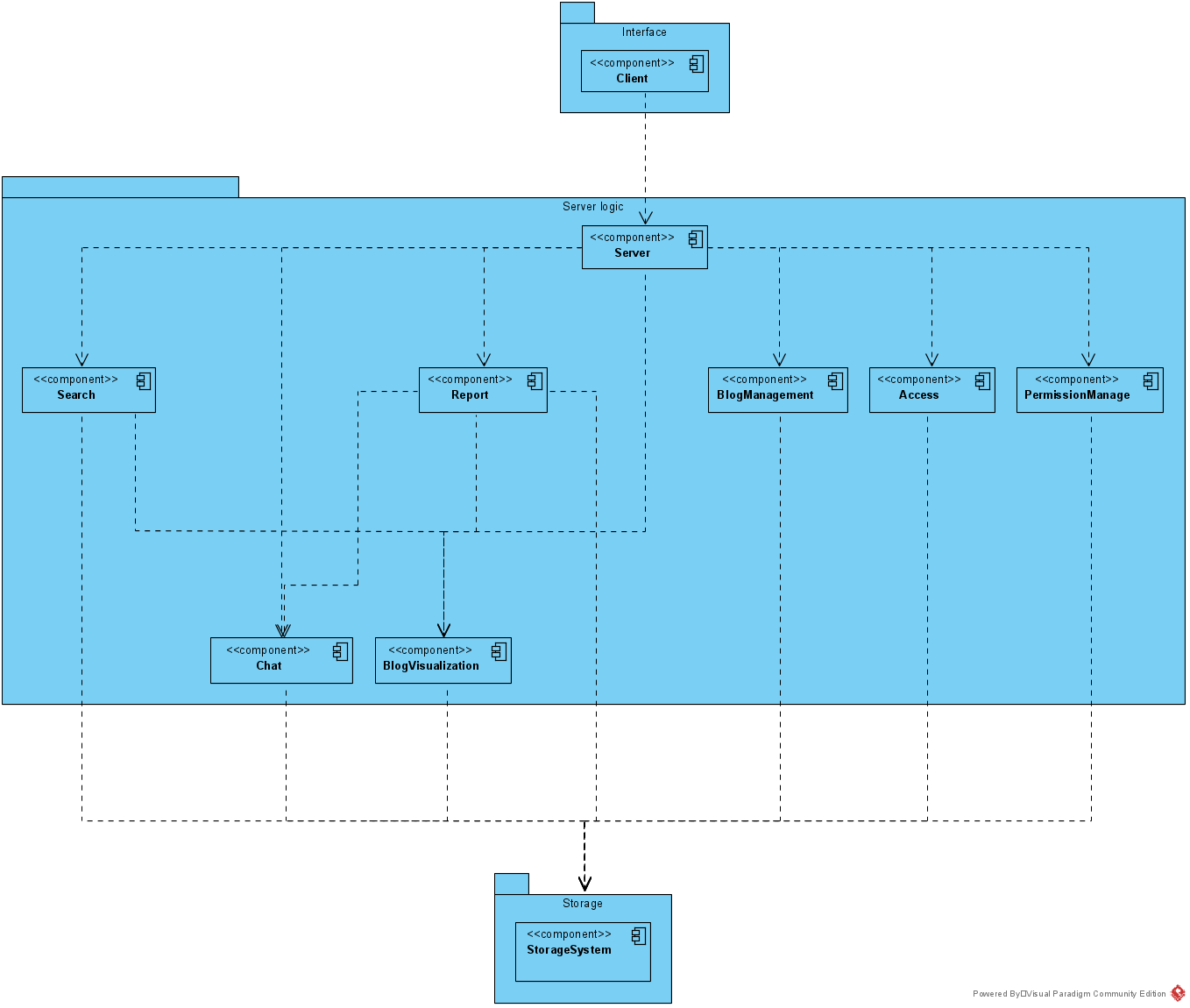
# **Panoramica**

Per il sistema si è proposta un’**architettura a tre livelli**.



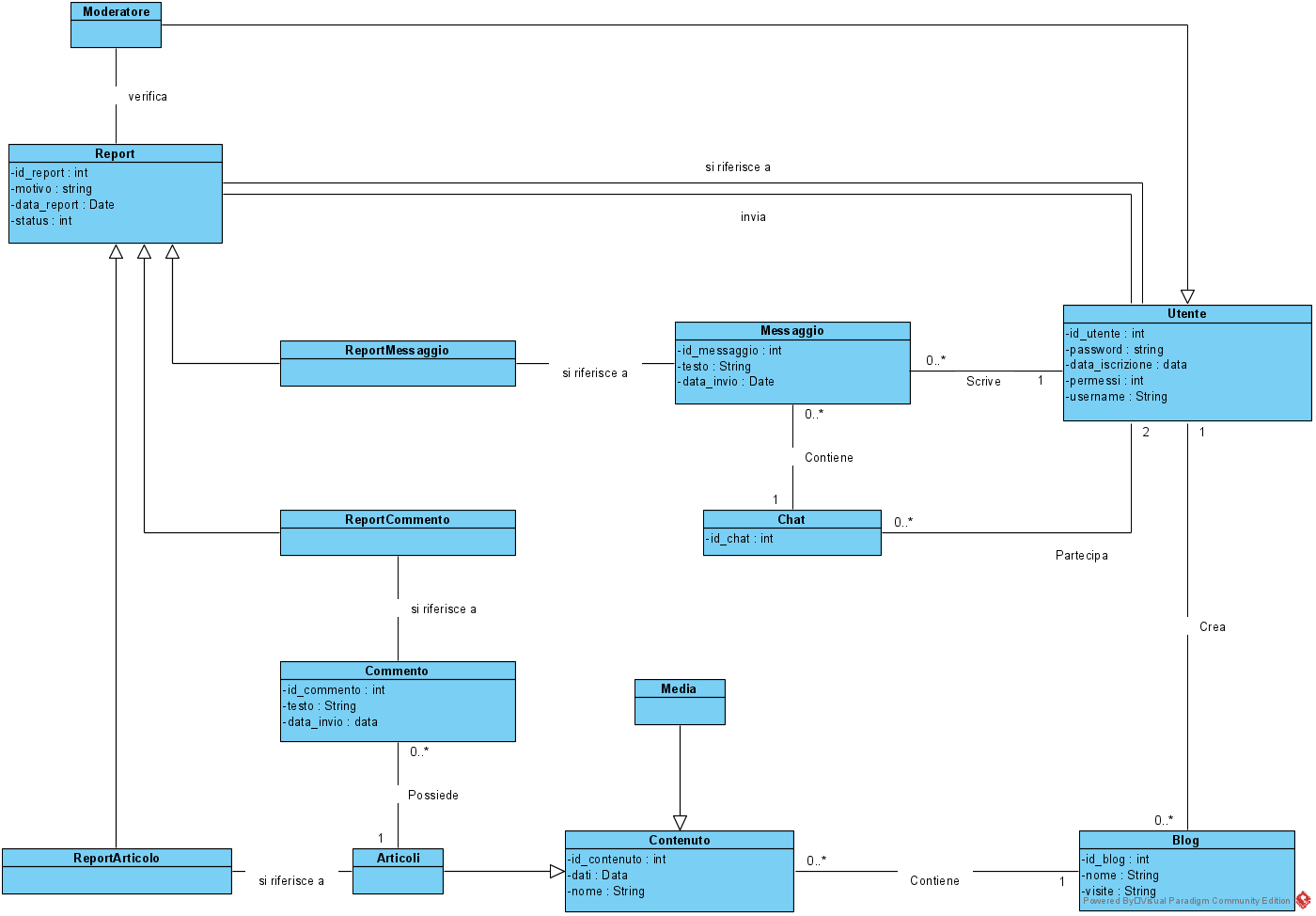
# **Decomposizione in sottosistemi:**

* Il sottosistema **Chat** è responsabile della creazione delle chat tra gli utenti, di inviare messaggi al loro interno e mostrare le chat già avviate
* Il sottosistema **Server** gestisce il controllo degli accessi e della concorrenza
* Il sottosistema **Client** fornisce un frontend per l’utente per inizializzare tutti i casi d’uso
* Il sottosistema **StorageSystem** è responsabile della memorizzazione di tutti gli oggetti persistenti
* Il sottosistema **PermissionManage** si occupa del controllo degli accessi
* Il sottosistema **Access** si occupa della registrazione e dell’autenticazione dell’utente
* Il sottosistema **BlogContentManagement** si occupa del recupero dei file di un blog, del caricamento di un file relativo ad un blog, della creazione di un articolo di un blog e della cancellazione di un file o blog
* Il sottosistema **BlogVisualization** si occupa di visualizzare i contenuti dei blog con la corretta formattazione rendendo e della gestione dei commenti relativi agli articoli
* Il sottosistema **Report** gestisce le modalità di segnalazione, controllo e verifica report
* Il sottosistema **Search** è responsabile della ricerca di utenti e blog, della visualizzazione del profilo utente e del recupero dei blog con più visualizzazioni



## **Gestione dei dati persistenti**

Il sistema si connetterà ad un database gestito con MariaDB, dove saranno memorizzati i dati personali degli utenti, le chat, i report e le informazioni relative ai blog. Poiché i file caricati dagli utenti potrebbero essere di dimensioni molto grandi si è scelto di memorizzarli sul file system del database server.



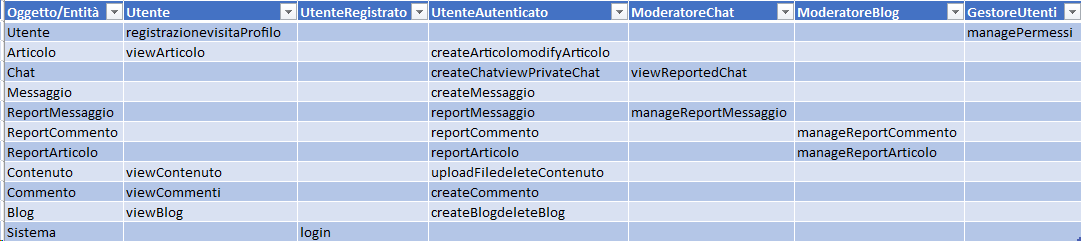
# **Mappatura Hardware/Software**

Il sistema utilizza un’architettura Client/Server. Il Web Server è rappresentato da Apache Tomcat 9 ed è situato su una singola macchina, la logica del sistema è costituita da Java Servlet mentre l’interfaccia utente è realizzata utilizzando pagine JSP (Java Servlet Page). Il Client è rappresentato dal Web Browser utilizzato dall’utente. La comunicazione tra i nodi è rappresentata da richieste e risposte http tra client e server, e da query in JDBC tra server e database.

1. **Controllo degli accessi e sicurezza**

Gli utenti potranno accedere al loro spazio tramite login con email e password. Le password verranno crittografate prima di essere memorizzate sul DB tramite l’utilizzo di salt e SHA2 a 512 bit. Il sistema dovrà garantire che i file caricati dagli utenti non compromettano la sicurezza e sanificare tutti i dati immessi in input per evitare l’esecuzione di codice arbitrario.

Per documentare i diritti di accesso succintamente usiamo la matrice di controllo degli accessi



È possibile vedere la tabella con una formattazione migliore in MatriceAccessi