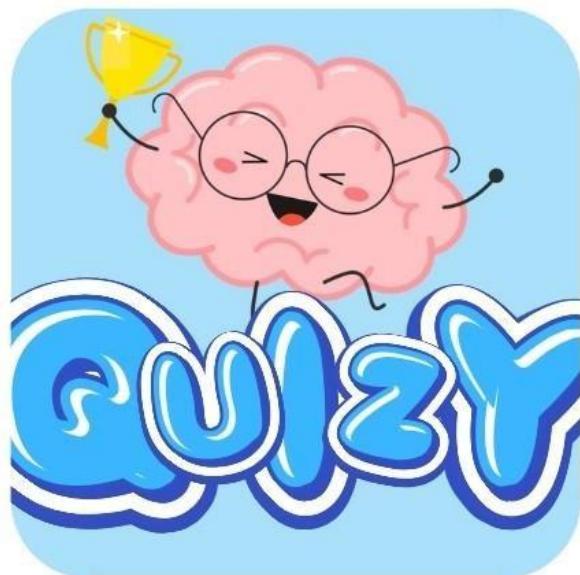


Università degli Studi di Salerno
Corso di Ingegneria del Software

Quizy
System Design Document
Versione 0.4



Data: 25/11/2025

Progetto:

	Versione: 0.4
Documento: System Design Document	Data: 25/11/2025

Coordinatore del progetto:

Nome	Matricola
------	-----------

Davide Nino Longobardi	0512119971

Partecipanti:

Nome	Matricola
Roberto Caiazza	0512116335
Fabiana Paciello	0512119578

Revision History

Data	Versione	Descrizione	Autore
17/11/2025	0.0	Stesura dell'indice e introduzione	Roberto Caiazza
18/11/2025	0.1	Stesura del System Decomposition	Davide Nino Longobardi
19/11/2025	0.2	Stesura del Persistent Data Management e overview	Fabiana Paciello
24/11/2025	0.3	Finitura dell'introduzione e Proposed software architecture	Davide Nino Longobardi, Fabiana Paciello, Roberto Caiazza
25/11/2025	0.4	Current Software Architecture e Glossario	Davide Nino Longobardi, Fabiana Paciello, Roberto Caiazza

INDICE

1. INTRODUZIONE	4
1.1. Purpose of the system.....	4
1.2. Design goals.....	4
1.3. Definitions, acronyms, and abbreviations	5
1.4. Overview	6
2. CURRENT SOFTWARE ARCHITECTURE	6
3. PROPOSED SOFTWARE ARCHITECTURE.....	6
3.1. Overview.....	6
3.2. Subsystem / software mapping.....	7
3.3. Hardware / software mapping.....	8
3.4. Persistent data managemet	8
3.5. Access control and security.....	9
3.5.1. Access Matrix.....	9
3.5.2. Security Strategy.....	10
3.6. Global software control.....	10
3.7. Boundary condition.....	11
4. GLOSSARIO.....	12

1. INTRODUZIONE

1.1. Purpose of the system

Lo scopo del presente sistema è quello di creare un applicativo per la somministrazione di Quiz a scelta multipla, avendo il totale controllo della piattaforma e degli utenti ad essa connessi. Questo semplifica la creazione dei test agevolandone la somministrazione agli utenti selezionati

1.2. Design Goals

1. **DG Privacy dei dati:** Il sistema deve garantire la privacy dei dati degli utenti, permettendo l'accesso solo a utenti autorizzati e impedendo le diffusione dei dati a terza parti.
2. **DG 2 Prestazioni:** Il sistema sarà in grado di gestire almeno 200 utenti con una latenza minima.
3. **DG 3 Tempi di risposta:** Il sistema deve rispondere alle richieste degli utenti in breve tempo, non superiore a 1 secondi
4. **DG 4 Usability:** Le funzionalità del sistema devono essere presentate in una interfaccia utente di facile comprensione.
5. **DG 5 Accuratezza dei dati:** Il sistema deve essere **accurato nel mantenere le risposte** senza scambiare quelle fornite.
6. **DG 6 Efficienza risorse:** L'applicazione deve essere **veloce** e non deve consumare eccessivamente la batteria del dispositivo su cui è installata.
7. **DG 13: Gestione Ruoli:** Il sistema deve garantire un alto livello di sicurezza dividendo la creazione dei quiz da chi li deve completare, con un sistema di **richieste di elevazione di ruolo** gestite da un Amministratore Gestore.

1.3. Definitions, acronyms, and abbreviations

	Ingegneria del Software	Pagina 4 di 12
--	-------------------------	----------------

Campo	Formato corretto	Eccezione
Username	Alfanumerico. Massimo 12 caratteri.	Username troppo lungo. Username già presente nel sistema.
Password	Alfanumerico. Minimo 8 caratteri e obbligatoriamente 1 lettera maiuscola, 1 minuscola, una cifra e 1 carattere speciale.	
Descrizione del quiz	Stringa di sole lettere. Deve contenere massimo 20 caratteri.	Descrizione del quiz troppo lunga.
Titolo	Alfanumerico.	Titolo troppo lungo.
Punti risposta corretta	Valore reale. Deve essere positivo	Punteggio negativo.
Punti risposta sbagliata	Valore reale. Deve essere negativo o valere 0.0	Punteggio positivo.
Tempo limite	Valore intero. Deve essere positivo	Valore negativo.
Password per partecipare (al quiz)	Alfanumerico. Minimo 8 caratteri e obbligatoriamente 1 lettera maiuscola, 1 minuscola, una cifra e 1 carattere speciale	
Domanda	Stringa. Massimo 300 caratteri	Domanda troppo lunga.
Risposta	Stringa. Massimo 200 caratteri	Risposta troppo lunga.
Token	Stringa. Massimo 400 caratteri	Token troppo lunga.

1.4. Reference

> Università degli Studi di Salerno – Corso di Ingegneria del Software – “Proposta

Progetto Tasky”

- > Università degli Studi di Salerno – Corso di Ingegneria del Software – “Problem Statement Tasky”
- > Università degli Studi di Salerno – Corso di Ingegneria del Software – “Requirements Analysis Document Tasky”

2. CURRENT SOFTWARE ARCHITECTURE

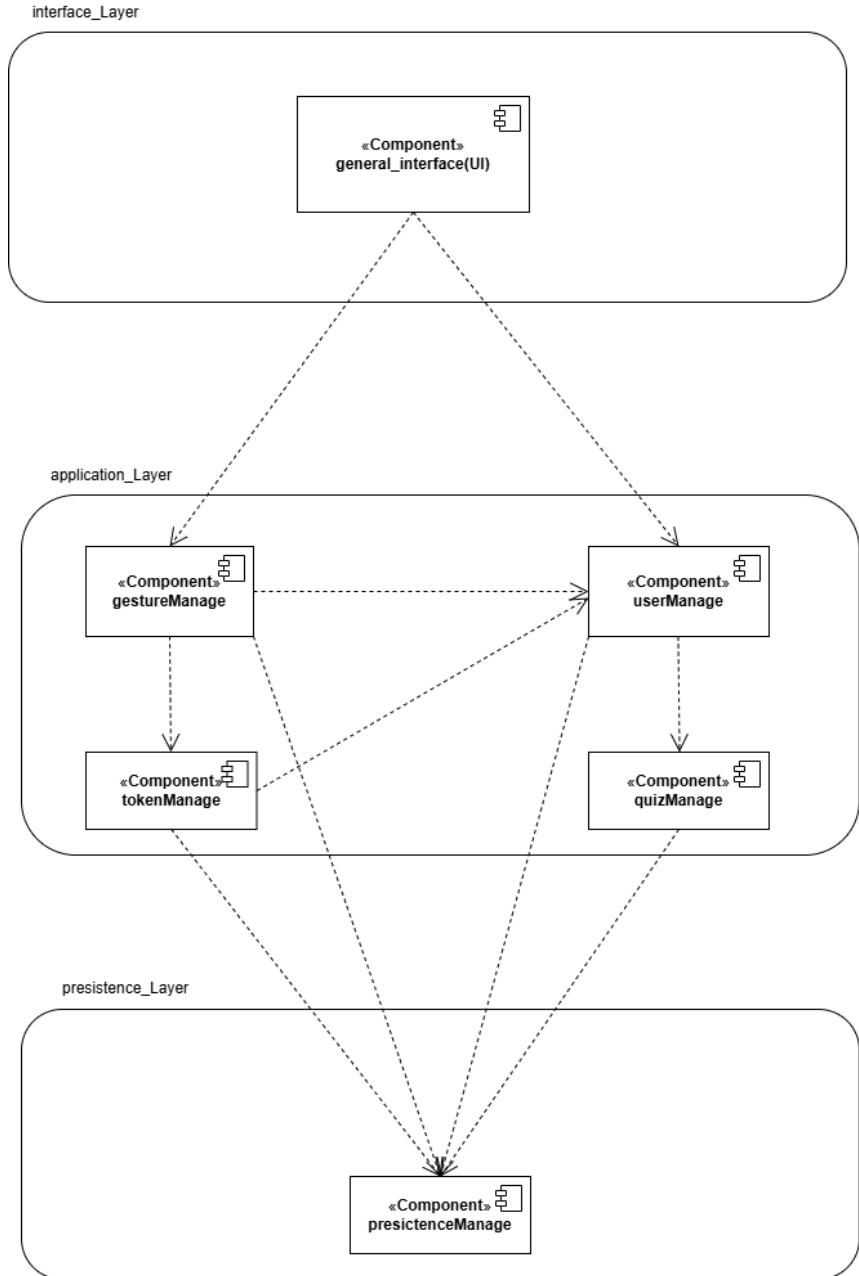
Il sistema attuale di riferimento per la somministrazione di quiz è identificato nella piattaforma **Kahoot!**. L'architettura è di tipo **Client-Server** basata su cloud (SaaS - Software as a Service), caratterizzata da una forte componente *real-time*. Sebbene questa architettura sia efficace per la gamification, presenta limiti che il sistema proposto (**Quizy**) intende superare.

3. PROPOSED SOFTWARE ARCHITECTURE

3.1. Overview

Quizy è una piattaforma per la creazione e somministrazione di quiz basata su un'architettura **Three-Tier** implementata in **Java**. La persistenza è garantita da **MySQL** e transazioni **ACID**.

3.2. Subsystem Decomposition



I sottosistemi sono organizzati in 3 livelli (Layer) separati:

- **Interface_Layer:** livello dedicato **all'interfaccia (UI)** usata dall'utente per comunicare col sistema attraverso il proprio dispositivo mobile.
- **Application_Layer:** livello dedicato alla **logica di business**, gestisce le richieste utente e la logica interna
- **Persistence_Layer:** livello dedicato alla **persistenza dei dati** e alla logica di mantenimento di questi ultimi

I nostri sottosistemi utilizzano diversi componenti divisi e organizzati nei loro livelli specifici, in particolare per l'application_Layer abbiamo:

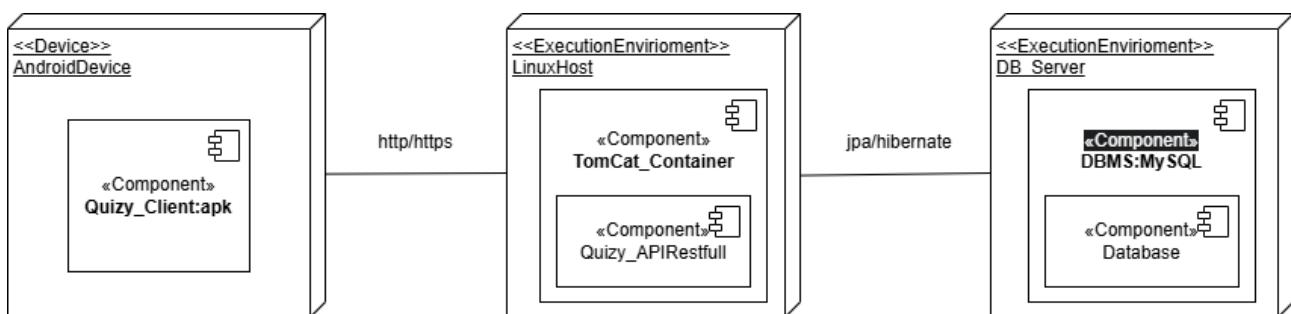
- **GestureManage:** si occupa della parte di gestione degli account da parte del **gestore account** e della autenticazione di quest'ultimo, permette anche logout e recupero password
- **UserManage:** componente che si occupa della gestione dell'accesso utente e della gestione

- account in generale (disponibile solo per gestore account), permette login, logout e recupero password.
- **TokenManage:** componente per la gestione dei token di richiesta per cambi ruolo o supporto tecnico, gestisce sia creazione che stato.
- **QuizManage:** componente che gestisce i quiz e la loro creazione, permette anche di eliminare, completare e visualizzare un quiz esistente.

La componente **presistenceManage** (**presistence_Layer**) gestisce l'interazione col **DBMS** e le operazioni da fare sulla base dati.

3.3. Hardware/software mapping

Quizy verrà implementato attraverso un'architettura a **tre livelli (Tree-Tier Architecture)**, che prevede più nodi per la distribuzione che dividono le implementazioni e permettono la scalabilità della applicazione.



I tre nodi principali:

- **L'Interface_Layer** è l'**AndroidDevice** che contiene **Quizy_Client:apk** ovvero un'applicazione Androidi scaricabile, è possibile adattare questa componente a qualsiasi dispositivo anche non Android essendo divisa dalla parte applicativa.
- **L'Application_Layer** è **LinuxHost** nodo che ospita il **TomCat_Container** che gestisce **Quizy_APIRestfull** ovvero un applicativo java che espone i servizi del sistema.
- **Il Persistence_Layer** è **DB_server** nodo contenente il **DBMS MYSQL** è il database del sistema

Il sistema impone all'applicativo Android di comunicare solo con i servizi offerti da **Quizy_APIRestfull**, norma fondamentale in questo tipo di architettura.

3.4. Persistent data management

Il Data Tier cruciale per la **manutenibilità, sicurezza e affidabilità** dell'applicazione, garantendo che le informazioni sui quiz, le risposte e le sessioni degli utenti siano conservate in modo strutturato e coerente.

La nostra applicazione si basa sull'utilizzo di un **database relazionale** per la persistenza dei dati di sistema. Il DBMS selezionato è **MySQL**, scelto per le seguenti motivazioni:

- **Robustezza e Open-Source:** È un DBMS robusto e la sua natura open-source ne facilita l'adozione e la manutenibilità a lungo termine.
- **Integrità Transazionale (ACID):** Supporta pienamente le **transazioni ACID** (Atomicità, Consistenza, Isolamento, Durabilità), requisito fondamentale per le

operazioni critiche come l'invio finale di un quiz e, soprattutto, per il **salvataggio progressivo** in caso di interruzioni.

- **Prestazioni e Sicurezza:** Offre ottime prestazioni per carichi di lavoro misti (CRUD operations) e garantisce l'integrità referenziale richiesta dal modello.
- **Compatibilità:** È ampiamente compatibile con l'ecosistema tecnologico moderno utilizzato per lo sviluppo del backend.

La persistenza sarà gestita mappando le entità logiche del sistema in **tabelle relazionali**, preservando le associazioni tramite chiavi primarie ed esterne.

Le entità logiche del database sono:

- **Utente:** caratterizzato da un nome, cognome, username e password, può essere di tipo '**Creatore**' (abilitato a costruire quiz) o di tipo '**User**' (autorizzato esclusivamente a sostenere i quiz);
- **Quiz:** caratterizzato da un tempo, difficoltà, titolo, descrizione, numero domande e data di creazione;
- **Domanda:** caratterizzata da punti dati in caso di risposta corretta, e punti dati in caso di risposta sbagliata;
- **Risposta:** caratterizzata da affermazione e una flag in caso di risposta corretta;
- **Ticket:** caratterizzato da una descrizione del ticket, tipo di richiesta e una descrizione della richiesta, rappresenta una richiesta al gestore;
- **Risponde:** La sua funzione primaria è garantire la durabilità dei dati di sessione (salvataggio progressivo) in tempo reale, permettendo il recupero della compilazione in corso in caso di spegnimenti o interruzioni.

3.5. Access control and security

3.5.1 Access Matrix

Quizy implementa meccanismi di autenticazione e protezione:

- **Autenticazione:** L'accesso iniziale si basa sull'accoppiata Username/Password, che garantisce l'accesso alle operazioni permesse. Sono definiti i seguenti ruoli principali, che determinano l'accesso ai sottosistemi:
- **Player:** Il Player è colui che può partecipare alle sessioni di quiz e visualizzare i propri risultati storici.
- **Quiz Creator:** Il Quiz Creator detiene il controllo sui quiz. Può creare, modificare ed eliminare i quiz nel sottosistema Model.
- **Gestore Account:** Gestore account è l'entità che effettua controlli e fornisce lo status a un utente di "Quiz Creator"

Ruolo	Visualizza quiz	Partecipa al quiz	Visualizza risultati	Crea nuovi quiz	Modifica quiz	Elimina quiz	Gestire domande e risposte	Visualizzare le segnalazioni/cambi ruolo
Player	✓	✓	✗	✗	✗	✗	✓	✗
Quiz Creator	✓	✓	✓	✓	✓	✓	✓	✗
			Ingegneria del Software			Pagina 9 di 12		

Gestore account	✓	✗	✗	✗	✗	✗	✗	✓

Il sistema di Quizy implementa un modello di accesso alle risorse basato su ruoli (**Role-Based Access Control**), ciò garantisce che le operazioni sugli oggetti siano limitate in base al ruolo assegnato.

3.5.2 *Security Strategy*

In accordo con i requisiti non funzionali quizy implementa una serie di strategie atte a mitigare i principali problemi noti.

1. Data Security:

- a. **Query injection protection:** per prevenire possibili manipolazioni illecite utilizzeremo white list di controllo (per query native e non), PreparedStatement di JDBC nativo e JPQL che impone query sicure.
- b. **Password Hashing:** per sicurezza di accesso il sistema conserva nel Database tutte le password con una funzione di hashing rendendo difficile la decriptazione di quest'ultima.

2. Access Security:

- a. **Token block:** Tramite appositi token cifrati blocchiamo accessi anomali ai ruoli al di fuori delle possibilità dell'account usati in tutta la comunicazione col sistema (al di fuori dell'autenticazione)
- b. **Access Filter:** attraverso ad una gestione dei ruoli e ad un mantenimento dei ruoli designati nella persistenza, in modo da garantire l'accesso al ruolo corretto.

3.6 *Global software control*

Il sistema utilizza un modello di controllo event-driven per gestire le comunicazioni e le interazioni tra i vari sottosistemi. Le richieste sono inviate dall'interfaccia utente e da eventi interni al sistema, garantendo un flusso coerente e sincronizzato tra i diversi componenti. Questo tipo di approccio assicura che le operazioni siano gestite in modo efficiente e che i dati siano coerenti, anche in presenza di operazioni concorrenti.

3.7 *Boundary condition*

> Inializzazione

Nome caso d'uso: Start System

Attori: Tecnico

Flusso di eventi:

- Il tecnico installa il sistema sulla macchina di lavoro
- Poi viene configurato il container
- All'avvio il sistema di Quizy si collega al database da remoto

Terminazione: Il sistema è operativo (Online) e pronto ad accettare richieste.

> Terminazione

Nome caso d'uso: Shut down

Attori: Sistemista

Flusso di eventi:

- Il Sistemista isola un modulo (ad esempio un'istanza di servizio).
- Il modulo isolato blocca l'accettazione di nuove richieste.
- Il modulo rilascia le risorse e si spegne.

Terminazione: Prima di terminare, il sottosistema deve assicurarsi di **committare** (rendere permanenti) tutte le modifiche

> Fallimento

Nome caso d'uso: Failure

Attori: Sistema di riavvio automatico

Flusso di eventi:

- Un modulo fallisce.
- Il sistema effettua la chiusura di tutti i servizi gestiti in start up.
- Le sessioni utente attive vengono automaticamente reindirizzate alla nuova istanza operativa.

Terminazione: Il modulo specificato viene disattivato senza interrompere il servizio globale per gli utenti rimanenti.

4. GLOSSARIO

- **Quiz:** Serie di domande a risposta multipla, finalizzate alla valutazione delle conoscenze dell'utente.
- **Utente:** Qualsiasi persona registra nel sistema, che può accedere all'app per partecipare ai quiz o visualizzare i risultati

- **API:** interfaccia software che gestisce la comunicazione tra l'app client e il server dell'applicazione.
- **Modulo:** Con questa parola che appare nel punto 3.7 ci riferiamo a una istanza del servizio, ovvero, un oggetto in esecuzione su un server ma traslato al livello di un sottosistema.
- **JDBC (Java Data Base Connector):** è un' API standard che permette all'applicazione scritta in Java di connettersi e interagire con il database.
- **JPA (Java Persistence API):** connettore per il DB basato su JDBC.
- **JPQL (Java Persistence Query Language):** Query formattate in linguaggio java.