QUIZZY, apprendimento divertente

Obiettivo

Rendere l’apprendimento il più coinvolgente e divertente possibile.

Difficoltà incontrate

Progettazione architettura Client-Server e Sockets

Trovare una soluzione in gradi soddisfare i requisiti e che possa permettere una futura scalabilità

Trovare equilibrio tra Documentazione e Sviluppo

Adottando XP, è stato complicato gestire la quantità di documentazione rispetto allo svliluppo

Quizzy

Paradigma di programmazione

Programmazione ad oggetti

Linguaggi di programmazione

Java-17, JSP, SQL

Tools

Eclipse, Maven, DBeaver, JDepend, Emerge, StarUML

Diagrammi UML

- Use Case Diagram
- Class Diagram
- State Diagram
- Sequence Diagram
- Activity Diagram
- Component Diagram

Quizzy

Tools Utilizzati

Eclipse

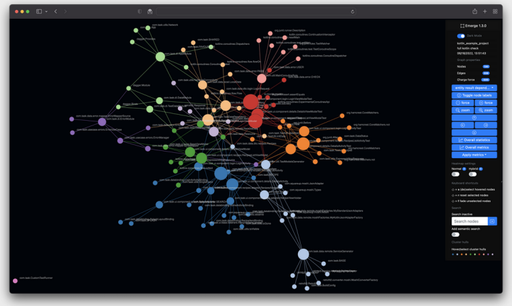
Maven

StarUML

DBeaver

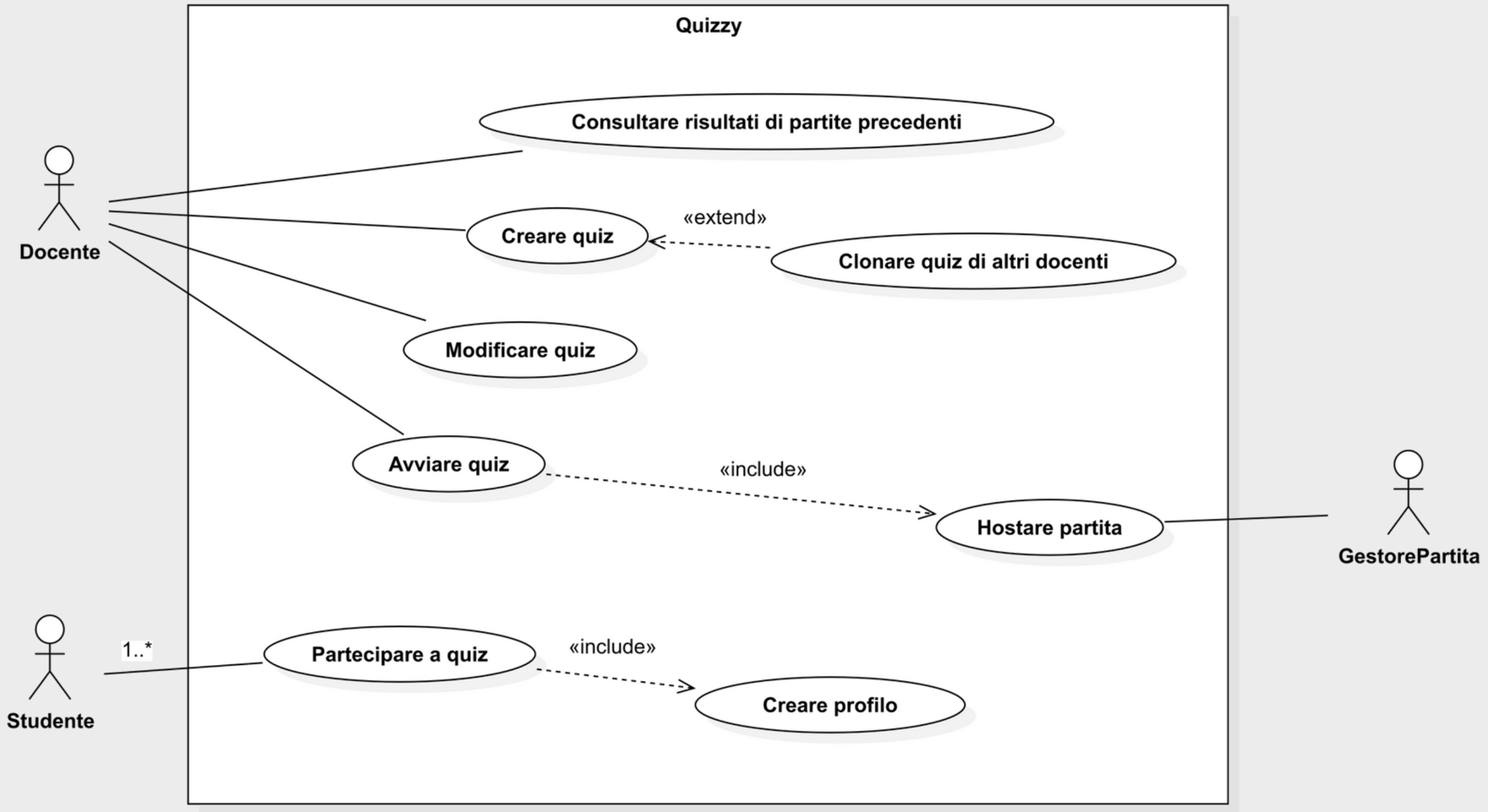
JDepend

Emerge



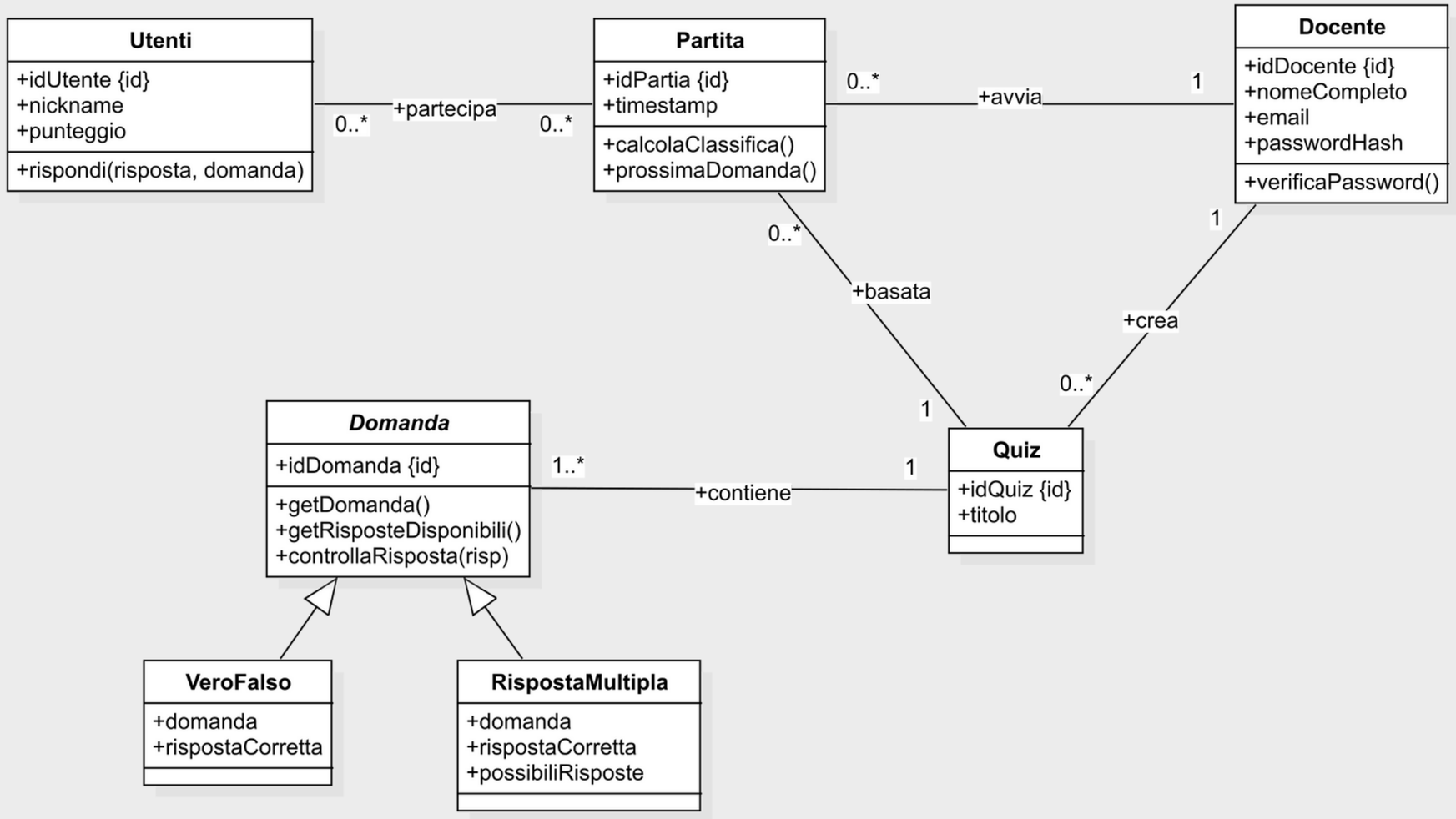
Quizzy

Use Case Diagram



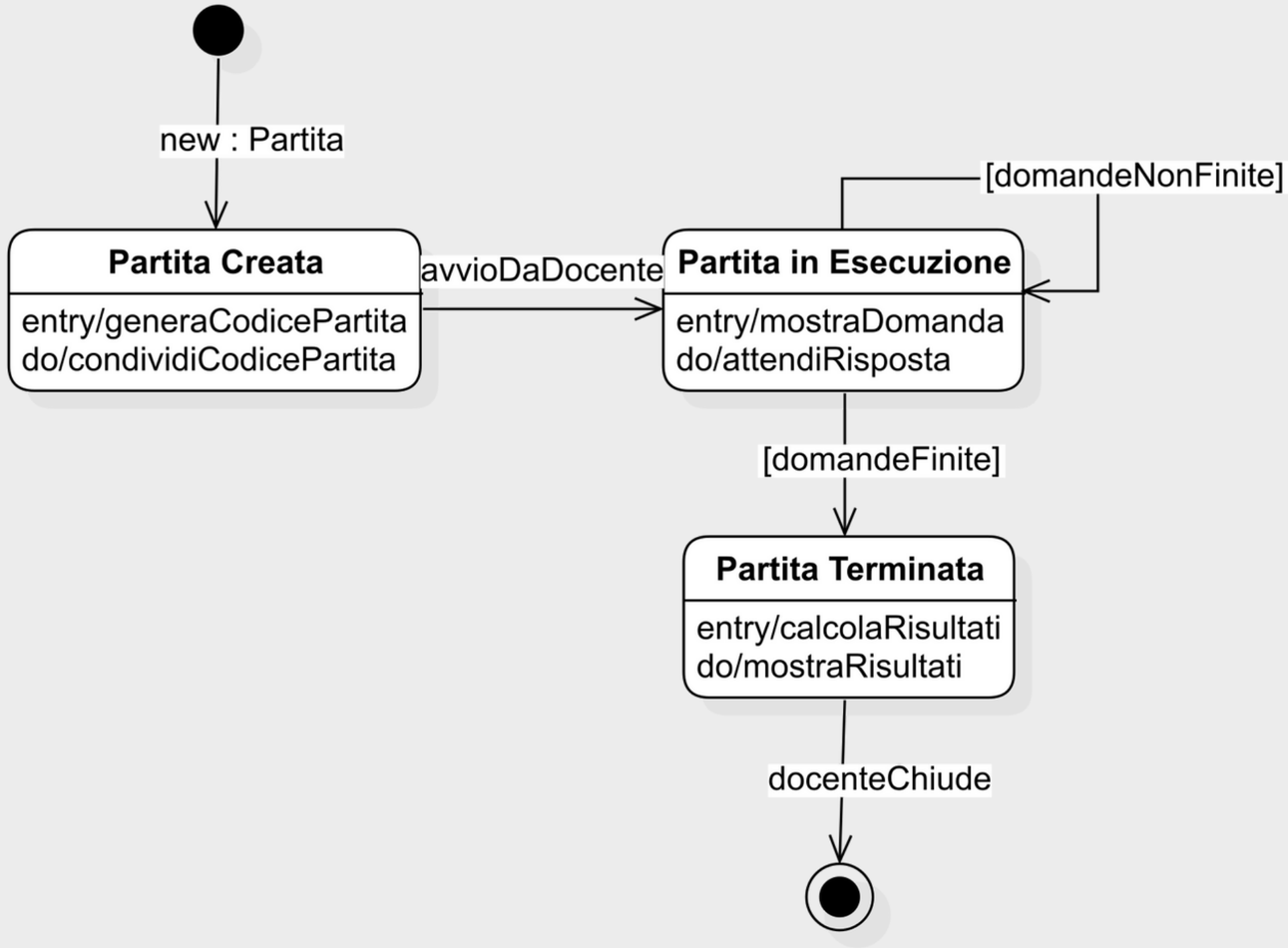
Quizzy

Class Diagram



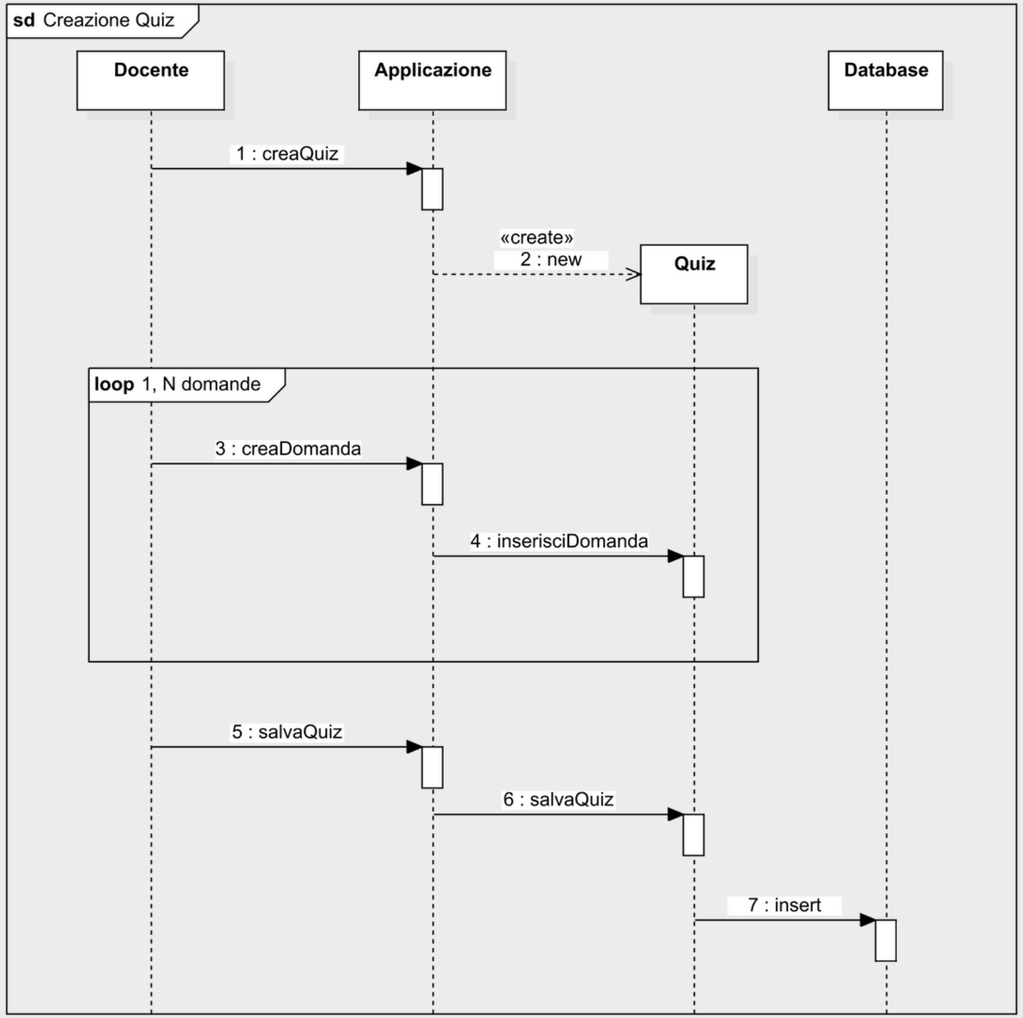
Quizzy

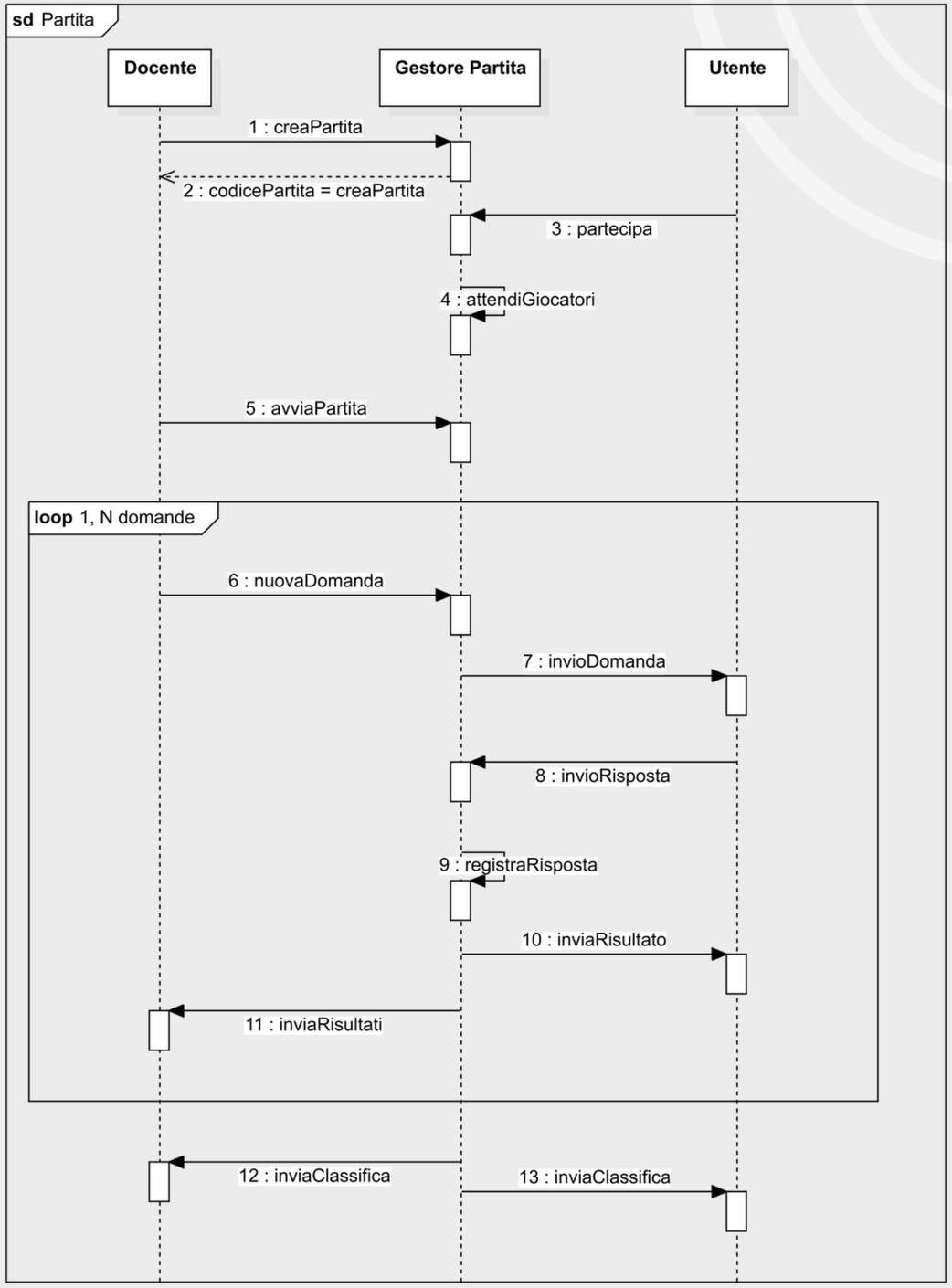
State Diagram



Quizzy

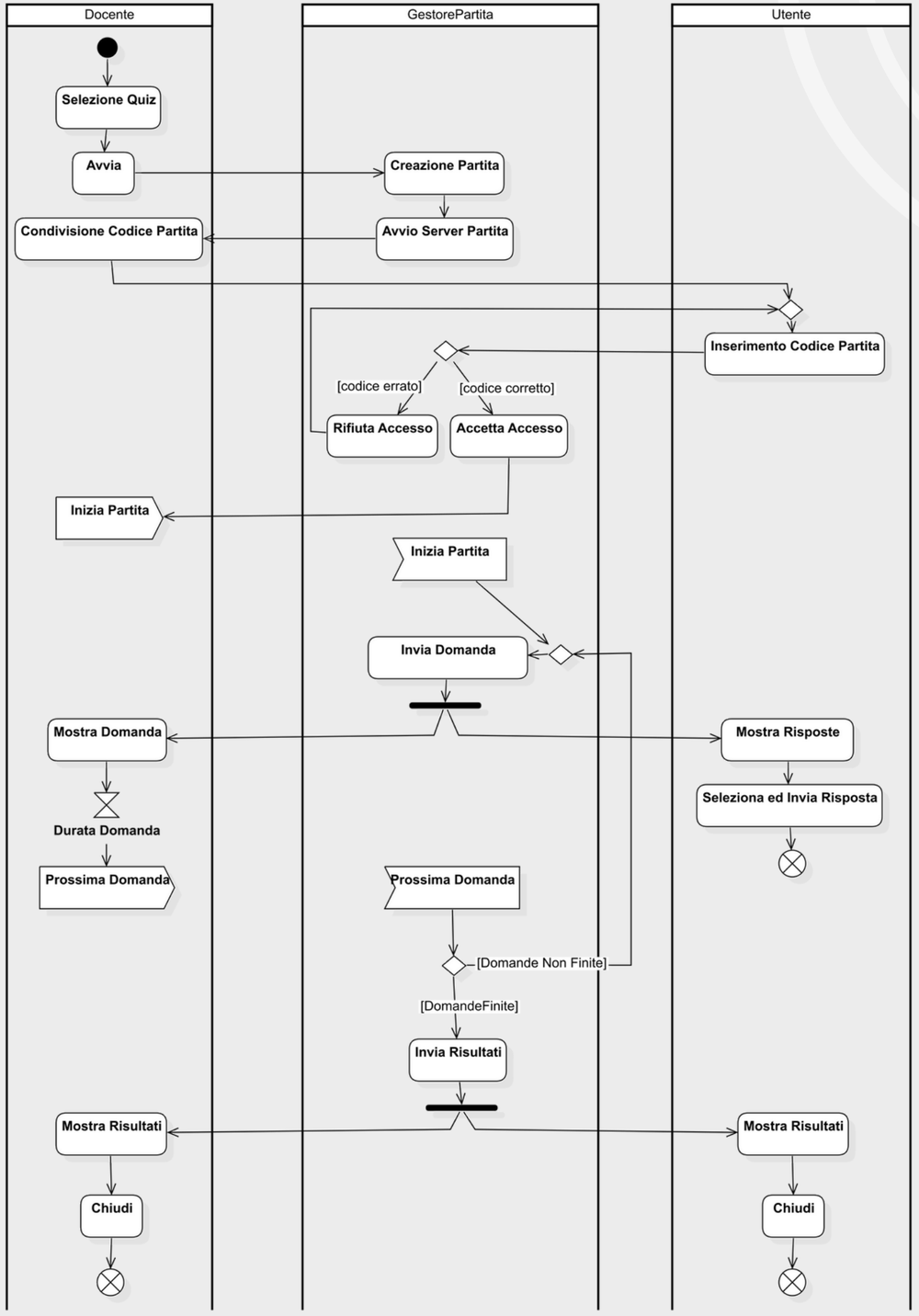
Sequence Diagram





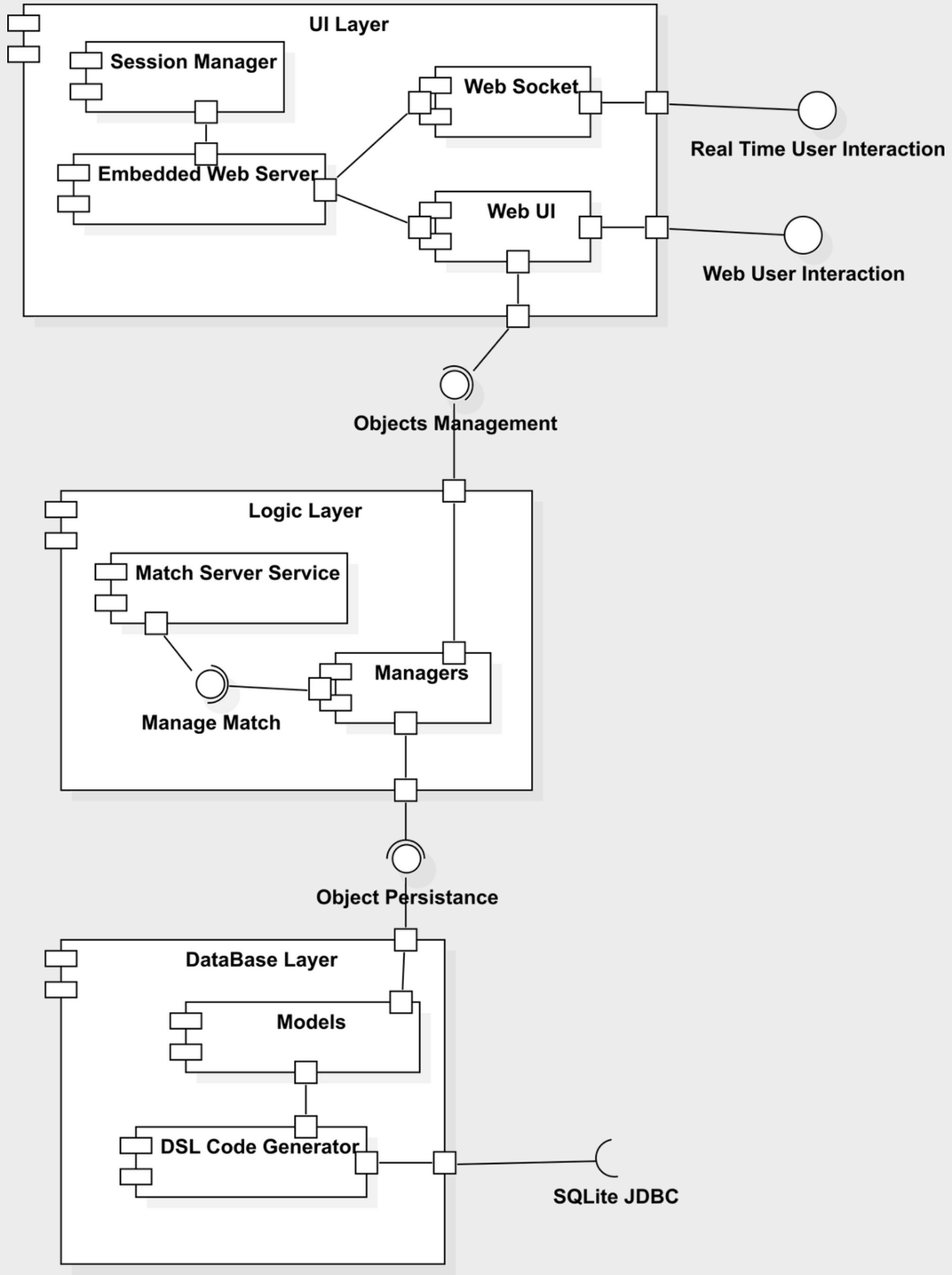
Quizzy

Activity Diagram



Quizzy

Component Diagram



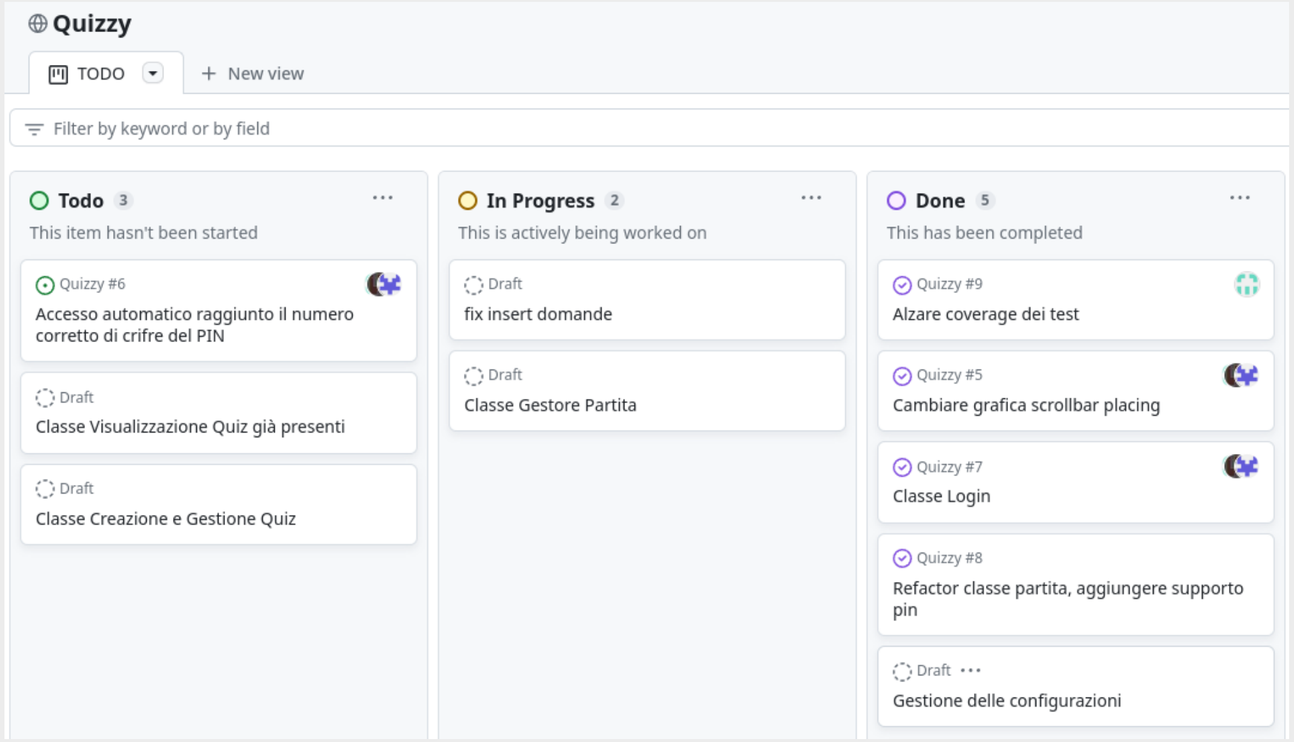
Quizzy

Software Management

Github

Issues & Labels

Kanban Board



Quizzy

Software life cycle

Pair Programming

TDD

Refactoring

Simple Design

Small Releases

RUP

Kanban Board

Quizzy

Requisiti

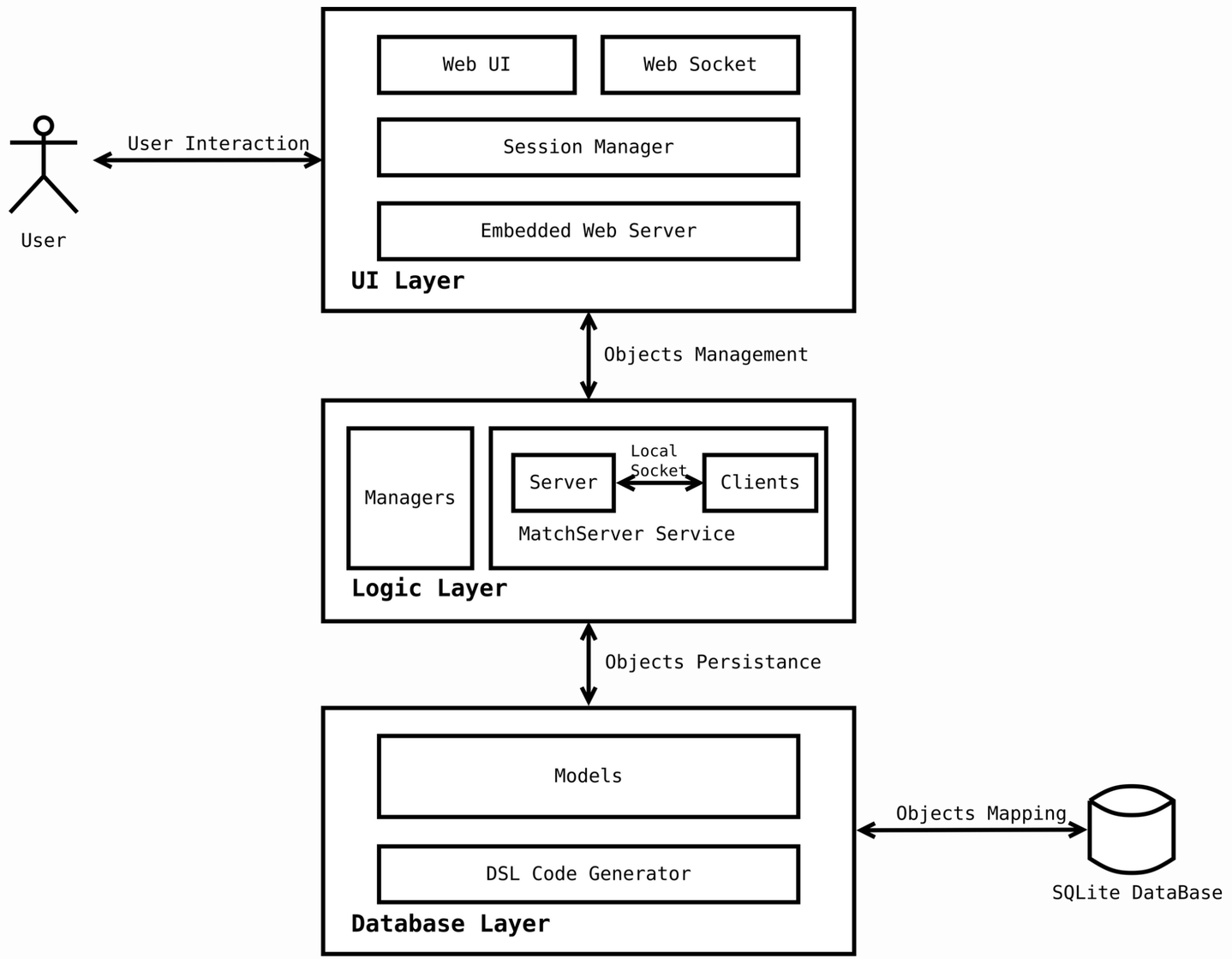
Sono stati individuati immedesimandosi nell'utente ed ispirandoci ad applicazioni simili già esistenti.

Sono riportati all'interno del documento dell’analisi dei requisiti.

Quizzy

Architettura

Abbiamo utilizzato un’architettura a 3 livelli



Quizzy

Software Design

Design Pattern

Wrapper

```java
@ServerEndpoint(value = "/join/{sessionId}")
public class WebSocketClientUtenteWrapper {

    ClientUtente client;

    @OnOpen
    public void onOpen(Session session, EndpointConfig config, @PathParam("sessionId") String sessionId) {

        HttpSession httpSession=null;
        try {
            httpSession = SessionManager.getSession(sessionId).getSession();
        } catch (IOException e) {
            e.printStackTrace();
        }

        try {
            if (httpSession != null) {
                UtenteManager um = (UtenteManager) httpSession.getAttribute("um");

                if (um != null) {
                    this.client=new ClientUtente(um.getPartitaPort(), um.getSessioneUtente(), (String message) -> {
                        try {
                            session.getBasicRemote().sendText(message);
                        } catch (IOException e) {
                            e.printStackTrace();
                        }
                        return null;
                    });
                }else {
                    session.close();
                }
            } else {
                session.close();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }

    }
```

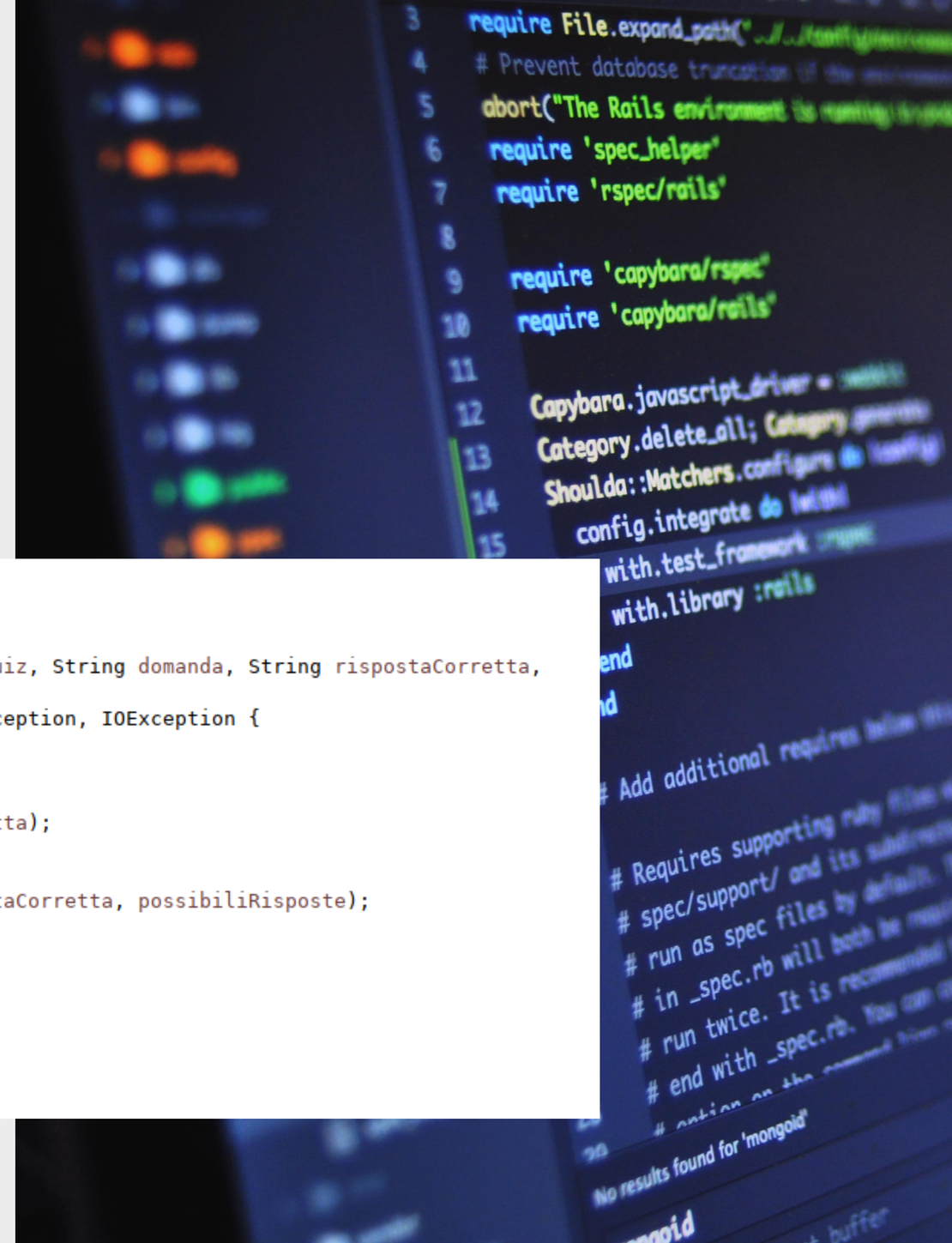
Quizzy

Software Design

Design Pattern

Factory

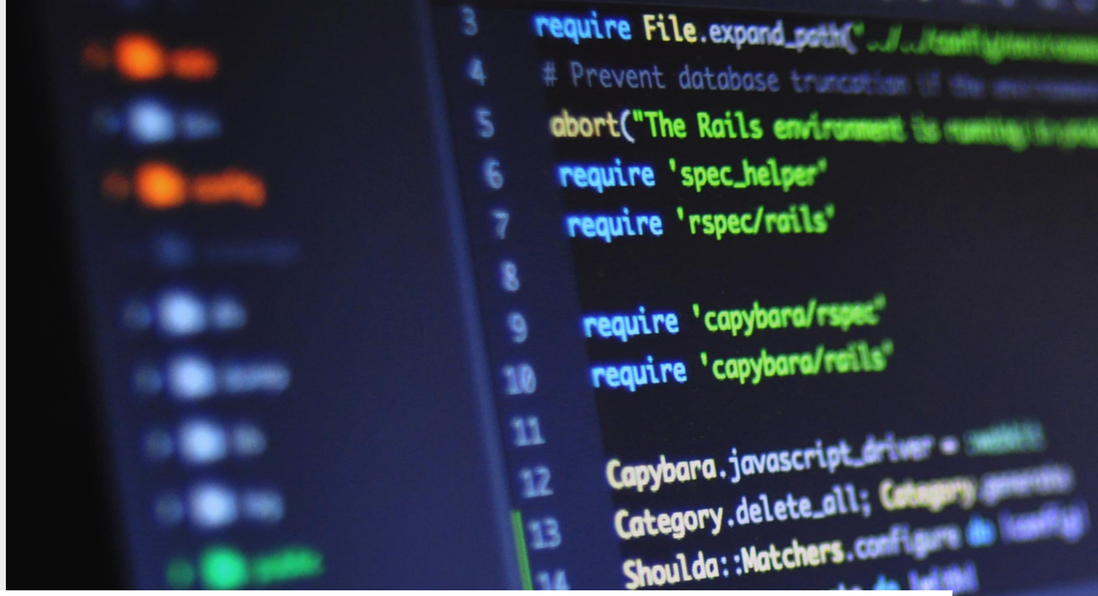
```
public class DomandeFactory {

    public static Domanda createDomanda(TipoDomanda type, Integer idQuiz, String domanda, String rispostaCorretta,
            List<String> possibiliRisposte)
            throws IllegalArgumentException, InvalidRecordInsertionException, IOException {
        Domanda obj = null;
        switch (type) {
        case VeroFalso:
            obj = new DomandaVeroFalso(idQuiz, domanda, rispostaCorretta);
            break;
        case RispostaMultipla:
            obj = new DomandaRispostaMultipla(idQuiz, domanda, rispostaCorretta, possibiliRisposte);
            break;
        }
        return obj;
    }

}
```



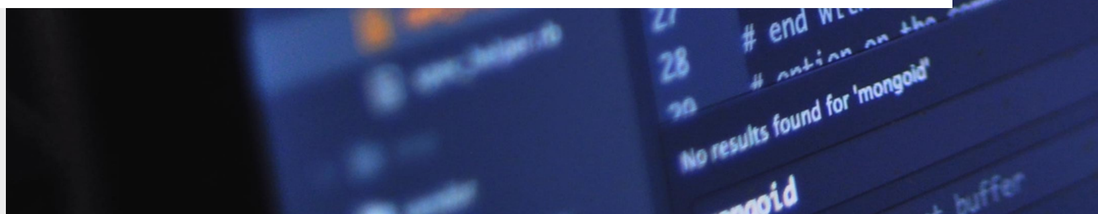
Quizzy

Software Design

Metriche di qualità del software



Package	Afferent Couplings	Efferent Couplings	Abstractness	Instability
it.quizzy.databaselayer	0	3	0	1
it.quizzy.databaselayer.exceptions	5	0	0	0
it.quizzy.databaselayer.models	8	6	0.2	0.43
it.quizzy.databaselayer.models.domande	4	8	0	0.67
it.quizzy.databaselayer.util	4	2	0	0.33
it.quizzy.logiclayer.factory	1	3	0	0.75
it.quizzy.logiclayer.manager	4	8	0	0.67
it.quizzy.logiclayer.server	3	3	0.29	0.5
it.quizzy.uilayer.launch	0	5	0	1
it.quizzy.uilayer.websocket	0	4	0	1



Implementazione

- ✅ Tipi di domande (VF, RM)
- ✅ Creazione Quiz
- ✅ Quiz Giocabili
- ✅ Personalizzazione profilo utente

- ❌ Modifica quiz
- ❌ Clonazione quiz
- ❌ Quiz con immagini e risposte variabili
- ❌ App mobile

Quizzy

Testing

Automatico con JUnit su:

- DataBase Layer
- Logic Layer

Element	Coverag	Covered Instructions	Missed Instructions	Total Instructions
DataBaseLayer	57,2 %	2.185	1.632	3.817
src/test/java	95,4 %	559	27	586
src/main/java	80,0 %	774	193	967
it.quizzy.databaselayer.models	87,8 %	438	61	499
Domanda.java	100,0 %	10	0	10
Utente.java	91,3 %	84	8	92
Partita.java	88,1 %	89	12	101
Docente.java	86,7 %	130	20	150
Quiz.java	85,6 %	125	21	146
it.quizzy.databaselayer.models.do	82,3 %	292	63	355
TipoDomanda.java	100,0 %	24	0	24
DomandaRispostaMultipla.java	87,9 %	116	16	132
DomandaVeroFalso.java	76,4 %	152	47	199

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
LogicLayer	86,5 %	1.534	239	1.773
src/main/java	82,4 %	1.015	217	1.232
it.quizzy.logiclayer.factory	89,3 %	25	3	28
DomandeFactory.java	89,3 %	25	3	28
it.quizzy.logiclayer.manager	89,0 %	560	69	629
DocenteManager.java	85,1 %	74	13	87
PartitaManager.java	91,2 %	331	32	363
QuizManager.java	87,6 %	99	14	113
UtenteManager.java	84,8 %	56	10	66
it.quizzy.logiclayer.server	74,8 %	430	145	575