



PROGETTO INGEGNERIA DEL SOFTWARE

PIATTAFORMA
VIDEOCORSI

STEFANO CALZA'
LORENZO LONGARETTI
MARTINA RASMO

OBIETTIVO

Il progetto prevede la creazione di una piattaforma di videocorsi. Questo progetto nasce dall'esigenza di un membro del team, Stefano, di creare una piattaforma per caricare delle videolezioni preregistrate per i corsi di ICDL (Patente Europea per l'Uso del Computer), in cui si può tener traccia dell'andamento dei corsisti tramite dei quiz, in modo tale da verificare che possano avere una preparazione adeguata per il superamento di un esame finale che fornisce una certificazione riconosciuta a livello europeo.

DIFFICOLTÀ INCONTRATE

- Coordinamento dei reparti del team
- Comunicazione con il cliente
- Creazione di un prodotto facilmente manutenibile
- Creazione di un ambiente sicuro e robusto
- Creazione di un prodotto finale user friendly

PARADIGMA DI PROGRAMMAZIONE

BACKEND



Apache
Tomcat



Java™



MySQL™

PARADIGMA DI PROGRAMMAZIONE

FRONTEND

B Bootstrap



SOFTWARE CONFIGURATION MANAGEMENT

Per il progetto è stato utilizzato GitHub.

Sono state create delle cartelle per ordinare i file in macroaree e diversi branch per continuare a lavorare su codice e documentazione, apportando modifiche e miglioramenti, senza interferire sul codice del progetto principale, il quale si trova nel branch «main».

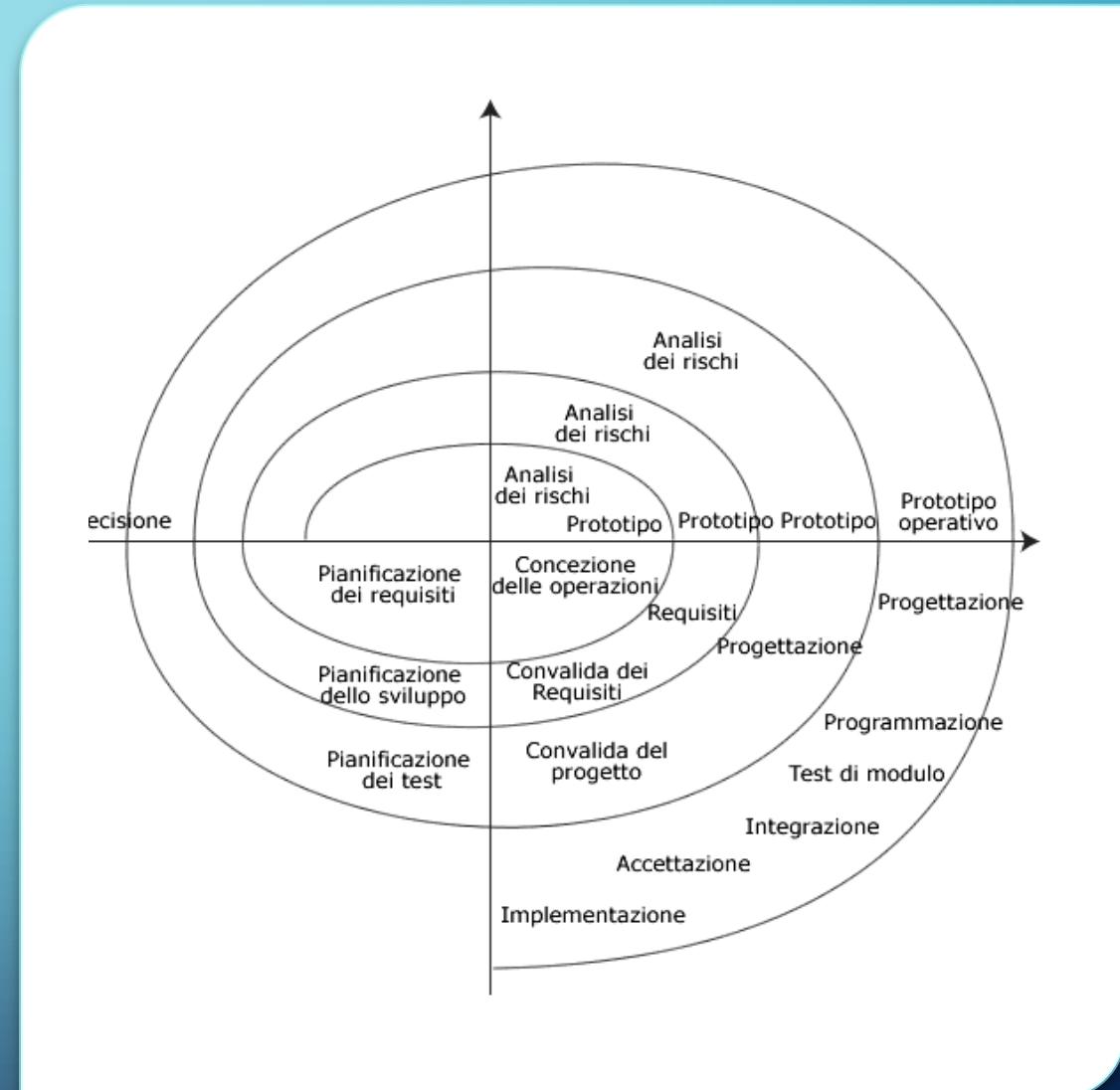
Per ogni merge di un branch nel main, è stata aperta una pull request che doveva essere approvata da almeno un membro del team.

Le comunicazioni ufficiali e le assegnazioni dei compiti sono avvenute tramite issues di GitHub, mentre per le comunicazioni meno formali e l'organizzazione delle riunioni, sono stati utilizzati anche WhatsApp e Discord.

Qualsiasi modifica è stata prima approvata dal CCB per verificare che rispettasse tutti gli standard del progetto.

La documentazione è stata scritta in formato Markdown su GitHub ed è stata visionata e approvata da tutti i membri del team.

SOFTWARE LIFE CYCLE



REQUISITI: MOSCOW

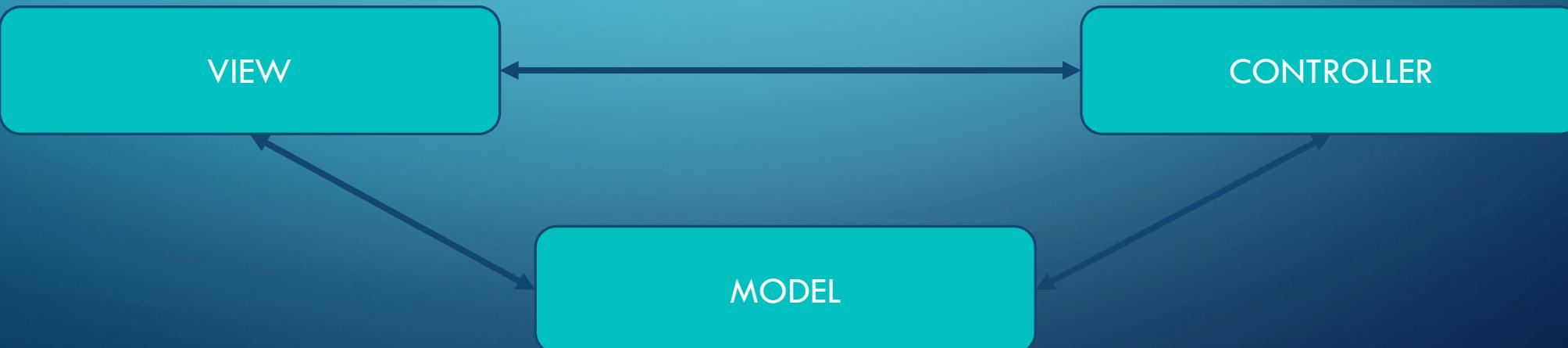


- Possibilità di modifica del profilo
- Implementazione del pagamento
- Implementazione della sicurezza degli account
- Implementazione di una chat utente-docente
- Visualizzazione delle risposte errate
- Implementazione pagina di iscrizione agli esami
- Implementazione dell'obbligo del superamento del quiz per passare al capitolo successivo
- Implementazione della pagina home
- Implementazione della pagina di login
- Implementazione della pagina di iscrizione ai corsi
- Implementazione della pagina di creazione dei corsi
- Implementazione della pagina di visualizzazione dei video
- Implementazione della pagina di esecuzione dei quiz
- Visualizzazione dati account

ARCHITETTURA



Architettura a 3 livelli, ovvero Presentation (la parte HTML), Business Logic (DAO e server) e Data Access (database) seguendo il pattern MVC, composto dal Model (il database), View (il front-end), e Controller (i DAO e il server).



DESIGN PATTERN

Singleton:

si assicura che una classe abbia una sola istanza e che possa essere navigata tramite determinati metodi. È stato scelto per rendere static la connessione al database, in questo modo c'è una sola istanza di connection.

```
public class ConnectionHandler {  
    private static ConnectionHandler sInstance = null;  
    private static Connection connection = null;  
  
    private ConnectionHandler() {}  
  
    public static ConnectionHandler getInstance() {  
        if(sInstance == null)  
            sInstance = new ConnectionHandler();  
  
        return sInstance;  
    }  
  
    public static Connection getConnection(ServletContext context) throws UnavailableException {
```

DESIGN PATTERN

Interface Read Only:

gestisce i privilegi di modifica di determinate classi. Quest'ultimo è stato scelto di applicarlo ai beans in quanto i DAO devono poter modificare i beans per registrare i valori, mentre i controller devono poter solo accedere ai beans.

MODELLAZIONE

- Diagramma dei casi d'uso
- Diagramma delle classi
- Diagramma di stato
- Diagramma di sequenza
- Diagramma delle attività

DIAGRAMMA DEI CASI D'USO

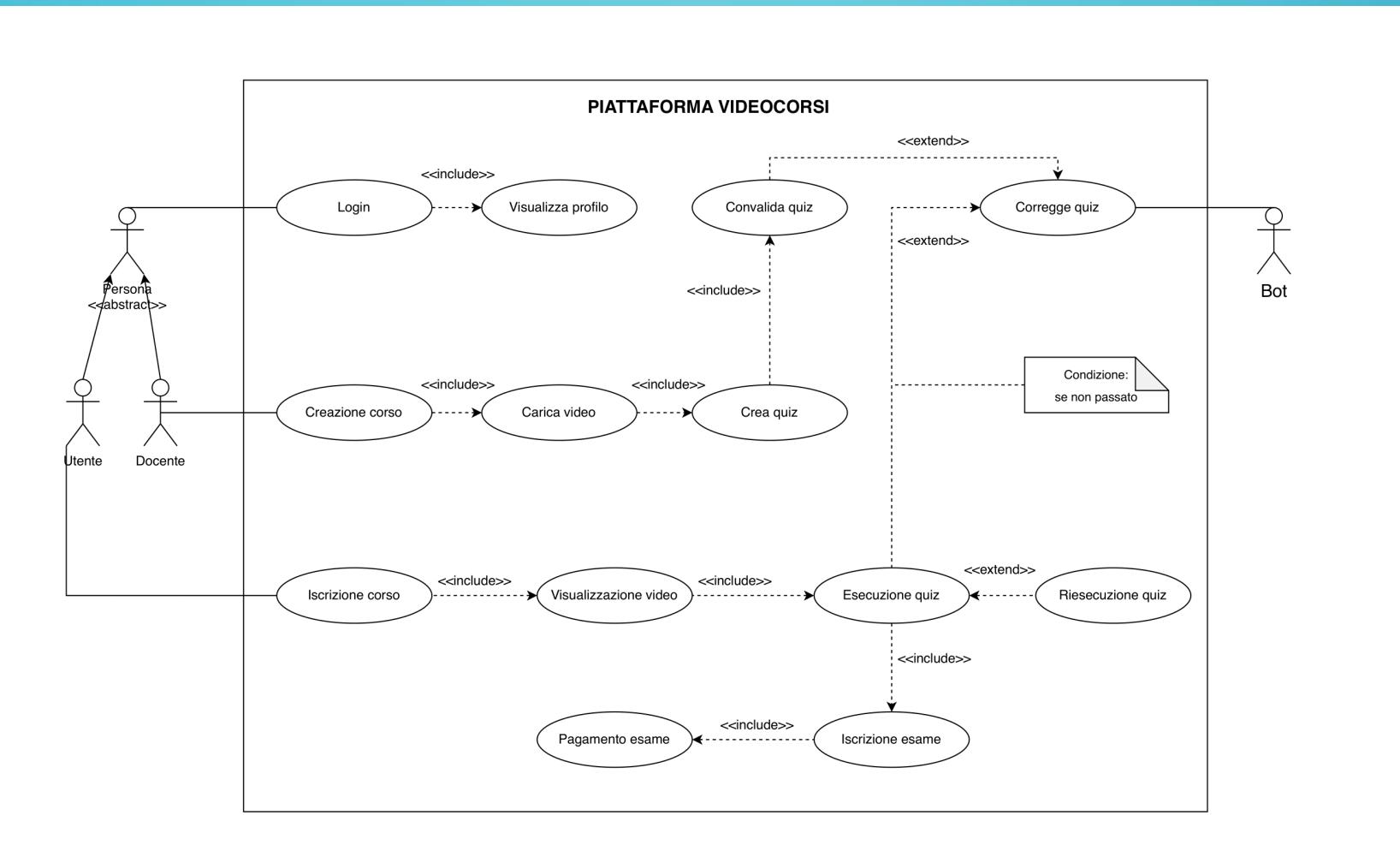


DIAGRAMMA DELLE CLASSI

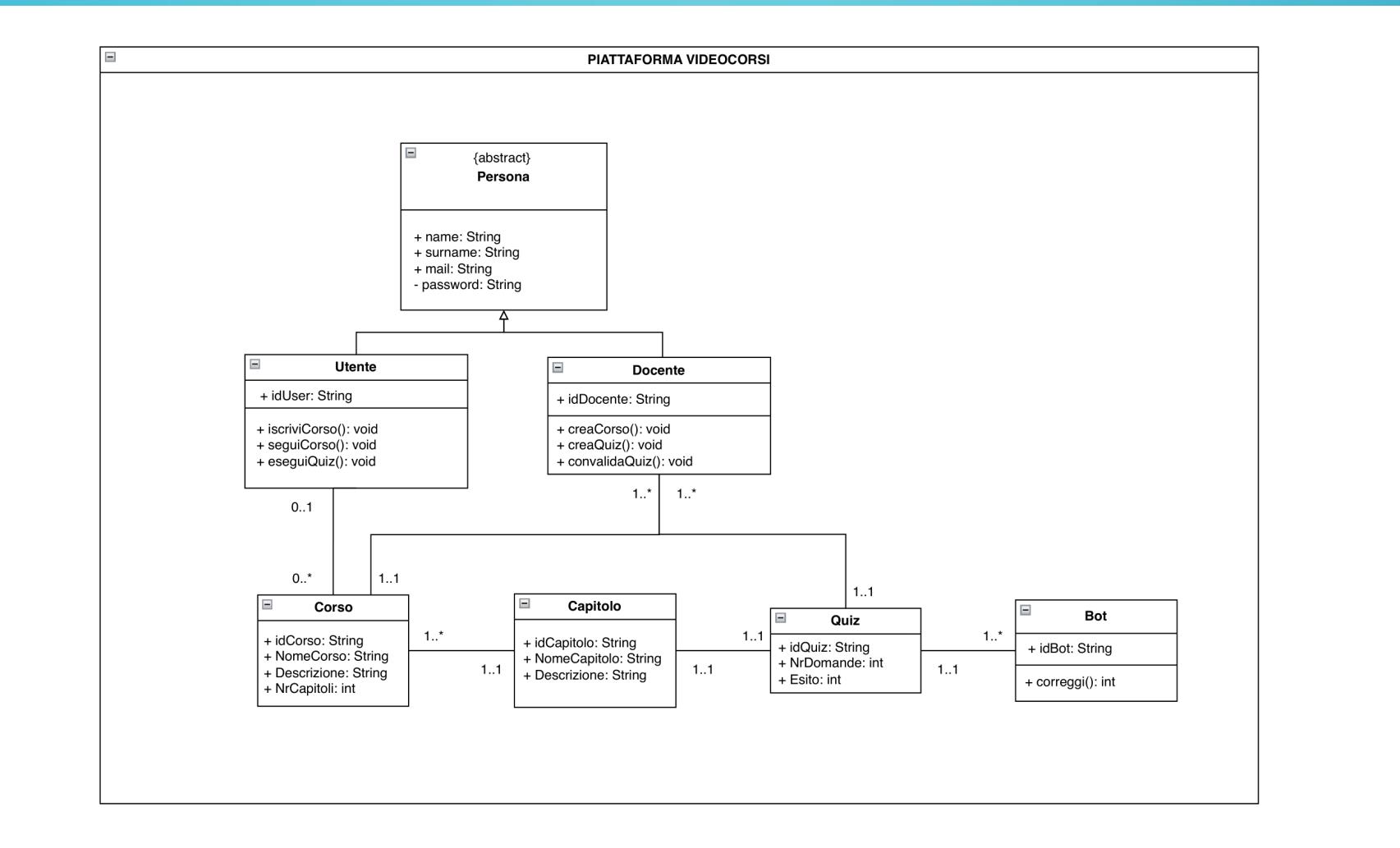


DIAGRAMMA DI STATO

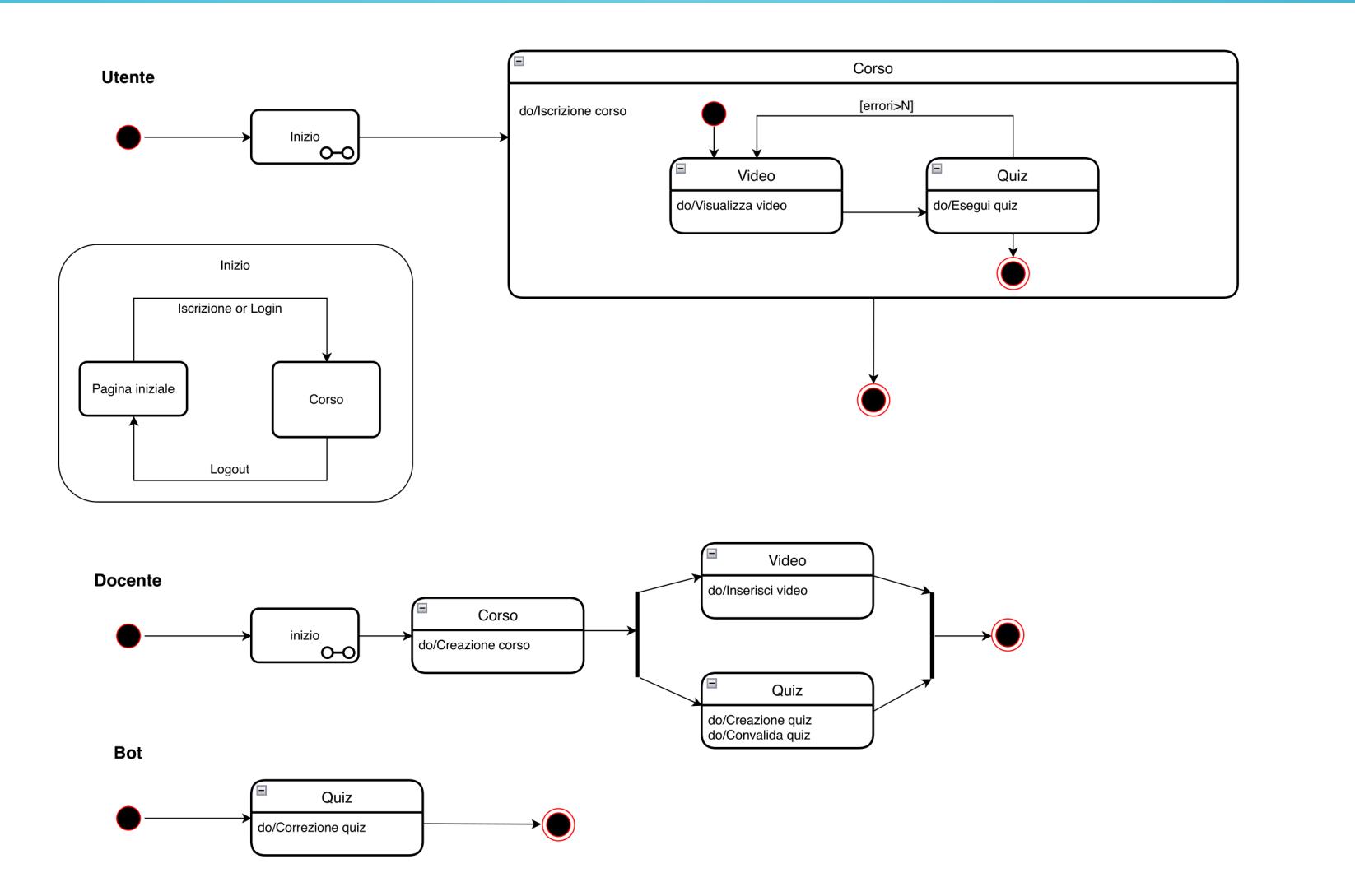


DIAGRAMMA DI SEQUENZA

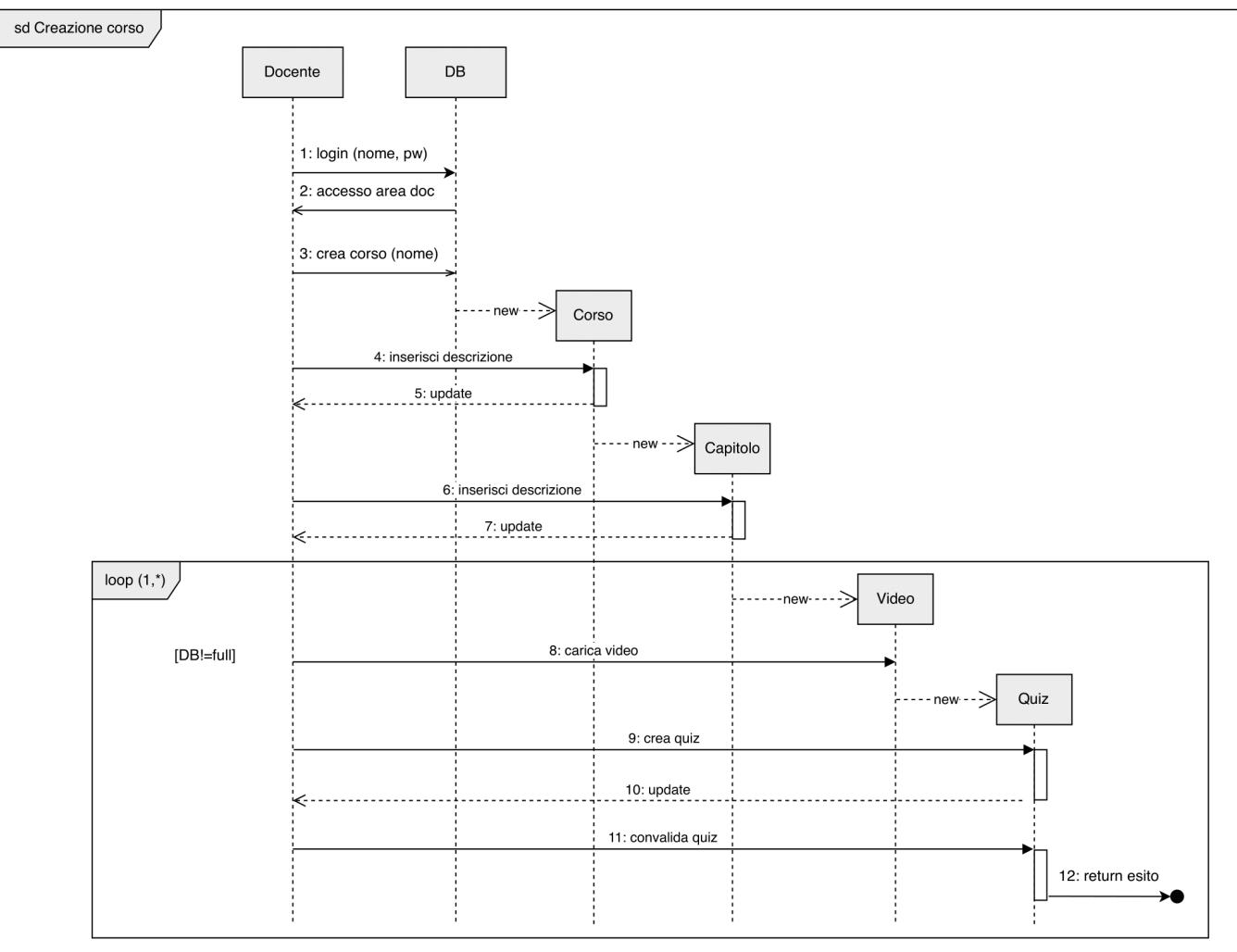
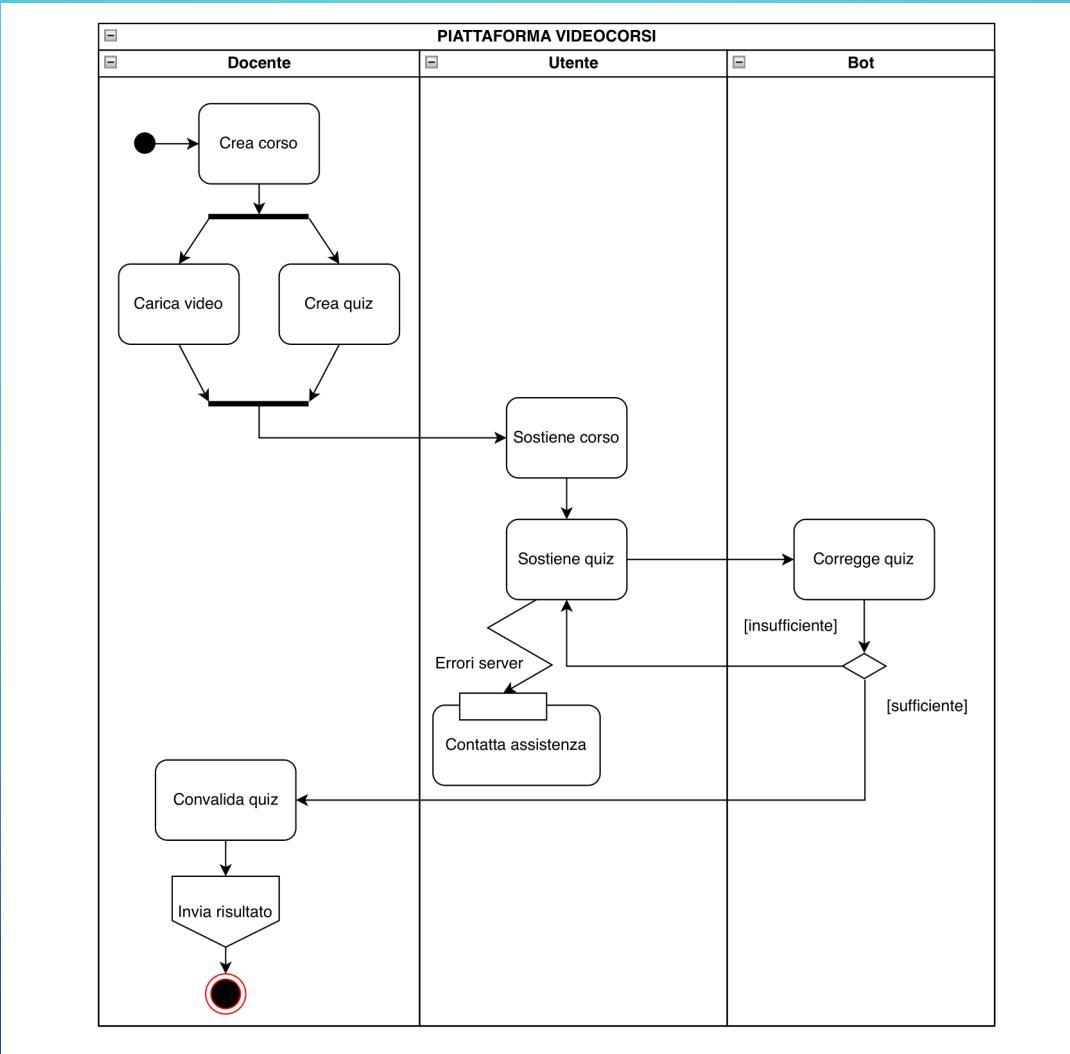


DIAGRAMMA DELLE ATTIVITÀ



IMPLEMENTAZIONE

MUST HAVE:

- ✓ Implementazione della pagina home
- ✓ Implementazione della pagina di login
- ✓ Implementazione della pagina di iscrizione ai corsi
- ✓ Implementazione della pagina di creazione dei corsi
- ✓ Implementazione della pagina di visualizzazione dei video
- ✓ Implementazione della pagina di esecuzione dei quiz
- ✓ Visualizzazione dati account

IMPLEMENTAZIONE

SHOULD HAVE

- ✓ Implementazione dell'obbligo del superamento del quiz per passare al capitolo successivo

COULD HAVE

- ✓ Implementazione della sicurezza degli account
- X Implementazione di una chat utente-docente
- X Visualizzazione delle risposte errate
- X Implementazione pagina di iscrizione agli esami

WON'T HAVE

- X Possibilità di modifica del profilo
- X Implementazione del pagamento

TESTING

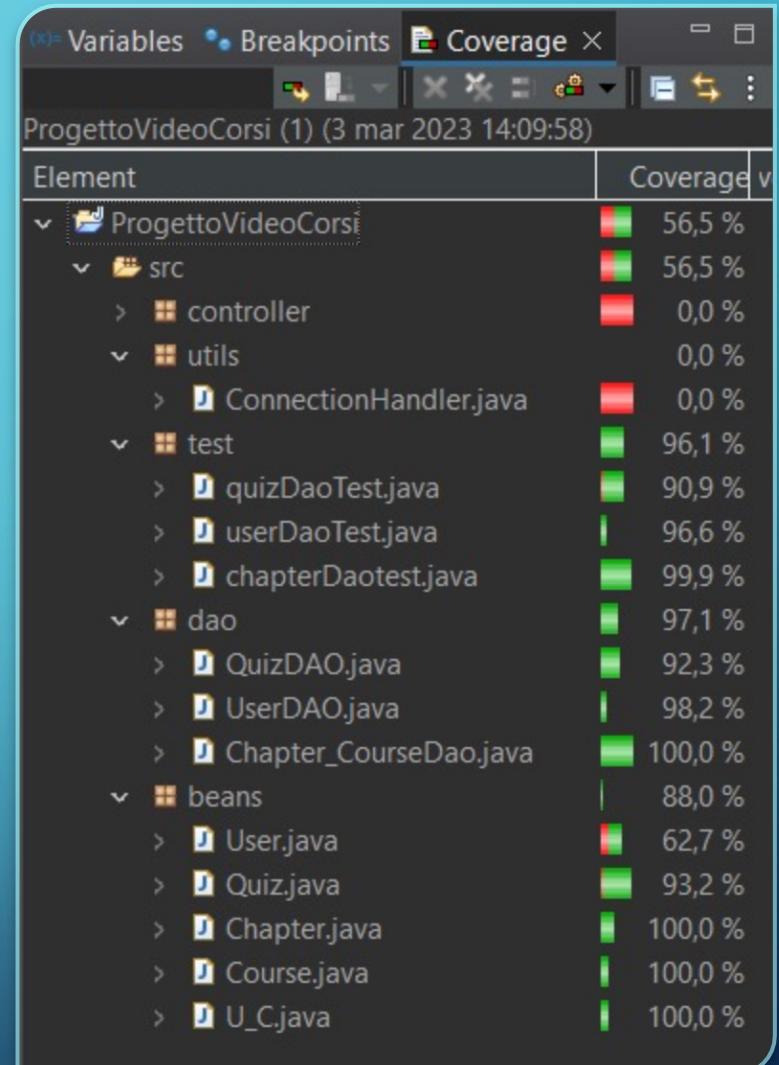
Durante lo sviluppo del back-end, come prima forma di test, si è svolta la lettura e l'ispezione del codice da parte di un membro del team che non ha scritto il codice da controllare. Già in questa fase è stato così possibile individuare errori e punti critici del codice.

Sono stati poi eseguiti una serie di test di unità tramite Junit.



TESTING

È stato infine eseguito un test di copertura dalla piattaforma eclipse (i controller sono stati testati a mano e saranno ulteriormente testati nelle prossime versioni tramite Mockito).



DEMO

Si può eseguire una demo della piattaforma sia per i corsisti che per i docenti.

PS. In questa fase è stato aggiunto solo un quiz (che si ripete poi per tutti i capitoli di tutti i corsi) per uno scopo dimostrativo. Prima della release ufficiale verranno inserite nel DB tutte le domande di tutti i capitoli.

GRAZIE PER LA CORTESE ATTENZIONE