SWENG 23/24

Progetto realizzato da:

Andrea Baggio andrea.baggio3@studio.unibo.it 0000838799

Valentina Sant valentina.sant@studio.unibo.it 0000906167

Federico Quarta federico.quarta2@studio.unibo.it 0000915744

INTRODUZIONE

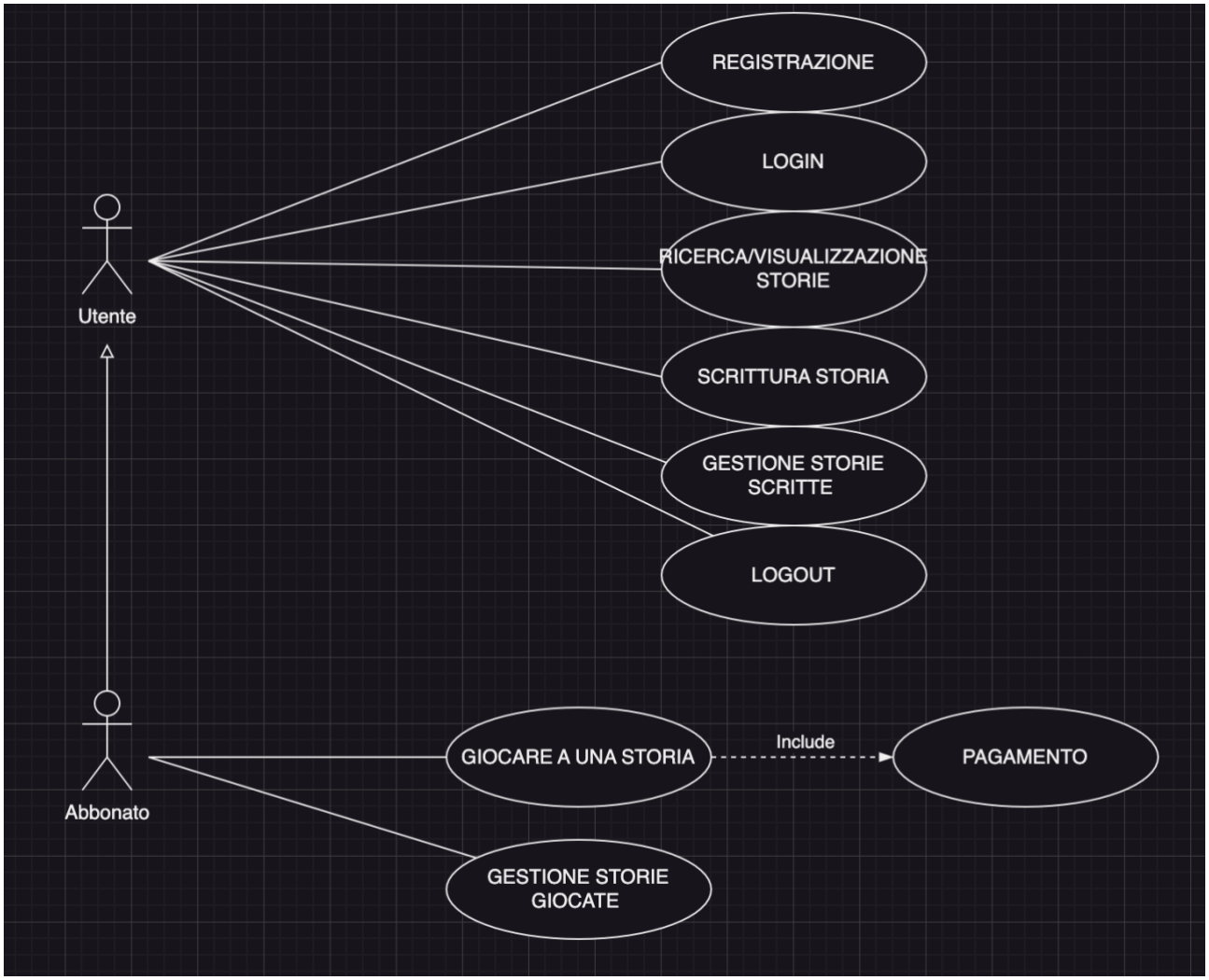
Per svolgere il progetto abbiamo scelto le seguenti tecnologie:

- Java
- Maven per la gestione delle dipendenze e del processo di build del progetto
- GWT per la realizzazione delle pagine web
- MapDB per la persistenza dei dati
- JUnit per la realizzazione di unit testing

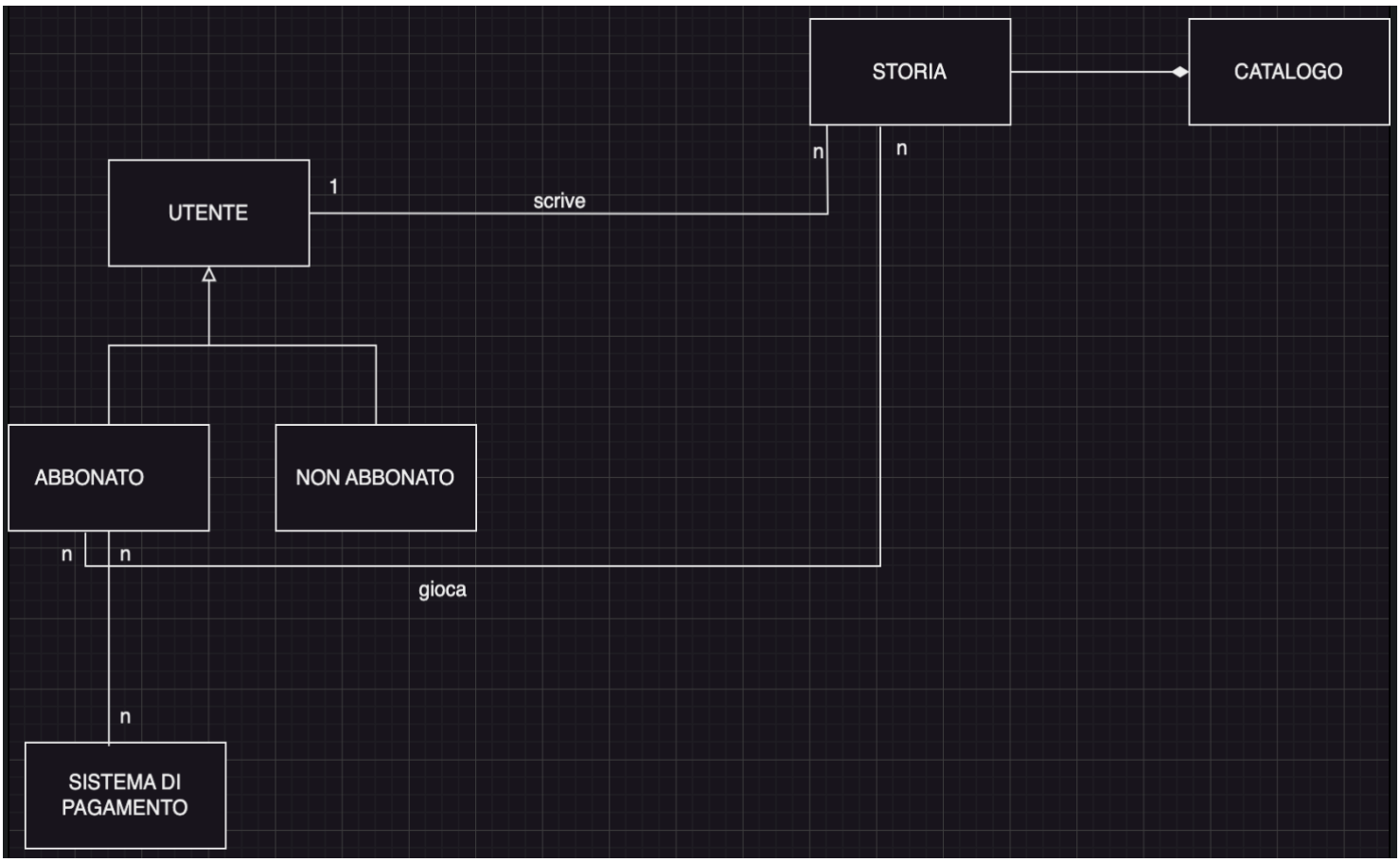
Il progetto è suddiviso in tre directories, che vengono create in automatico da visualStudio alla creazione di un progetto gwt tramite il comando da terminale WebAppCreator:

- "client": nella cartella client troviamo le varie classi java utilizzate esclusivamente lato client; inoltre troviamo i services corrispondenti che gestiscono la comunicazione sincrona (es: LoginService.java) e la comunicazione asincrona (es: LoginServiceAsync.java).
- "server": nella cartella server troviamo le classi java utilizzate esclusivamente lato server ovvero la gestione del Database creata attraverso MapDb e le classi che implementano i metodi definiti nelle interfacce precedenti che ci permettono di eseguire le operazioni richieste (es: LoginServiceImpl.java)
- "shared": nella cartella shared troviamo le classi condivise dal client e dal server.

MODELLO DEI CASI D'USO



MODELLO DI DOMINIO



GLOSSARIO

Utente non abbonato: tipologia di utente che si è registrato all'applicazione senza pagare. L'unica azione che può fare è scrivere le storie.

Utente abbonato: tipologia di utente che si è registrato all'applicazione e paga una tantum utilizzando un sistema di pagamento esterno, ciò permette all'utente, oltre che scrivere le storie, anche di giocarle.

Storia: è una narrazione interattiva scritta da un utente/autore e che può essere giocata da un utente abbonato/giocatore.

Scenario: una sequenza di "situazioni" che fanno parte di una storia, scritte dall'autore e scelte dal giocatore nel momento del gioco.

Catalogo: è un insieme di storie presenti nell'applicazione, aggiunte man mano che vengono scritte dagli utenti.

Scrittura storia: ogni storia è scritta da uno scrittore, che crea gli scenari e le regole che governano le scelte del giocatore.

Giocare una storia: ogni giocatore legge gli scenari di una storia e prende delle decisioni che influenzeranno lo sviluppo della storia, oppure risponde ad un indovinello (testuale o numerico).

Sistema di pagamento: sistema esterno all'applicazione attraverso il quale il giocatore abbonato paga una tantum per poter giocare alle diverse storie.

DIARIO DEL PROGETTO

Data	Fase	Attività
17.02.24	Inizio	Scaricati di tutti i software utili al progetto
17.02.24	Inizio	Integrazione Board in GitHub
18.02.24	Inception	Discussione per la creazione del modello dei casi d'uso
18.02.24	Inception	Discussione per la creazione del modello di dominio
18.02.24	Inception	Creata versione finale modello dei casi d'uso
18.02.24	Inception	Creata versione finale del modello di dominio
18.02.24	Inception	Descrizione dettagliata di casi d'uso
18.02.24	Inception	Creato il glossario dei dati
18.02.24	Inception	Scelta dello stile da adottare per la pagina web
19.02.24	Construction	Creato Product Backlog e SpringPlanning su BoardGitHub
Dal 20.02.24 al 21.02.24	Sprint 1	Scrum Master: Valentina Sant Development: Federico Quarta Product Owner: Andrea Baggio
Dal 20.02.24 al 21.02.24	Sprint 1	Creazione del progetto, creato lo scheletro con le prime classi: user, story, scenario...
Dal 21.02.24 al 28.02.24	Sprint 2	Scrum Master: Andrea Baggio Development: Valentina Sant Product Owner: Federico Quarta
Dal 21.02.24 al 28.02.24	Sprint 2	Inizio logica di registrazione e aggiunta pagina login, homepage e pagamento
Dal 28.02.24 al 04.03.24	Sprint 3	Scrum Master: Federico Quarta Development: Andrea Baggio Product Owner: Valentina Sant
Dal 28.02.24 al 04.03.24	Sprint 3	Creazione pagina di ricerca e parziale implementazione della logica riguardante la pagina di ricerca
Dal 04.03.24 al 09.03.24	Sprint 4	Scrum Master: Valentina Sant Development: Federico Quarta Product Owner: Andrea Baggio
Dal 04.03.24 al 09.03.24	Sprint 4	Css di login, registrazione e homepage e provati degli sfondi, aggiunta delle pagine di scrivi e gioca storia e sistemato il login
Dal 09.03.24 al 20.03.24	Sprint 5	Scrum Master: Andrea Baggio Development: Valentina Sant Product Owner: Federico Quarta
Dal 09.03.24	Sprint 5	Modifiche e test eseguiti dal developer team in base alle

al 20.03.24		richieste del product owner e verifiche dello scrum master
Dal 20.03.24 al 27.03.24	Sprint 6	Scrum Master: Federico Quarta Development: Andrea Baggio Product Owner: Valentina Sant
Dal 20.03.24 al 27.03.24	Sprint 6	Aggiornamento della ricerca delle storie e aggiunta la selezione dei fitri per la ricerca e modifiche alle funzionalità di scrittura storia
Dal 27.03.24 al 29.03.24	Sprint 7	Scrum Master: Valentina Sant Development: Federico Quarta Product Owner: Andrea Baggio
Dal 27.03.24 al 29.03.24	Sprint 7	Aggiunte le funzionalità per eliminazione, salvataggio e caricamento storia e modifiche css
Dal 29.03.24 al 08.04.24	Sprint 8	Scrum Master: Andrea Baggio Development: Valentina Sant Product Owner: Federico Quarta
Dal 29.03.24 al 08.04.24	Sprint 8	Aggiunto pulsante logout e funzionalità per modificare le storie già scritte e test junit per cartella shared

ORGANIZZAZIONE DEL LAVORO

Per lo sviluppo del progetto abbiamo seguito la suddivisione in fasi prevista dalla metodologia Agile con la gestione dei processi in Scrum

Inizialmente abbiamo redatto un Product Backlog ed uno Sprint planning in cui abbiamo deciso a grandi linee come suddividere il lavoro nell'arco di tempo che avevamo a disposizione per svolgere il progetto, dividendo le varie attività. Inoltre per ogni Sprint valutavamo le priorità alle varie fasi di lavoro.

Per ogni sprint, ogni membro del team si è scambiato di ruolo ricoprendo uno dei ruoli previsti dal metodo Scrum:

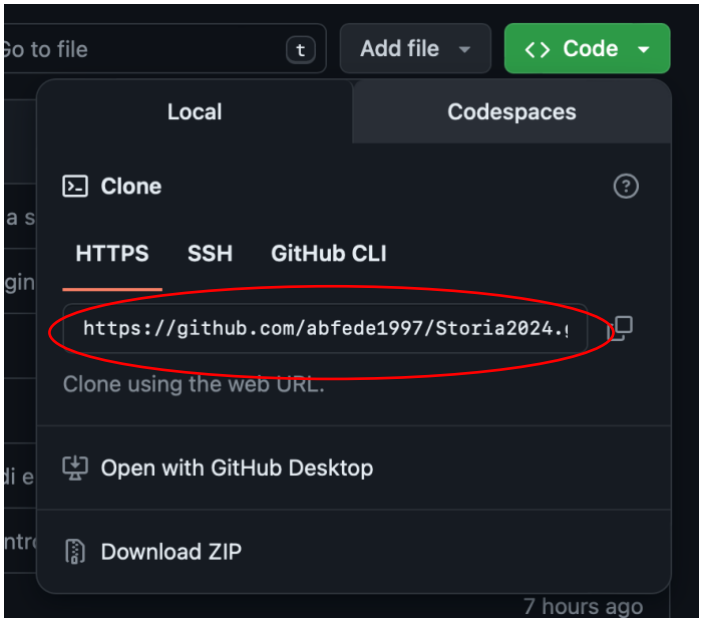
- **Product Owner**: colui che decide le priorità, le scadenze e le caratteristiche dello sprint a cui è a capo.
- **Scrum Master**: è il responsabile del corretto utilizzo della metodologia Scrum.
- **Developer Team**: colui che si occupava di implementare il progetto in totale autogestione

La durata di ogni Sprint variava in base alla quantità di lavoro che dovevamo eseguire, in particolare chi era Developer di quello sprint si dedicava solo a quello durante il giorno. Nel caso in cui al termine di uno Sprint rimaneva qualche task non completata essa si spostava automaticamente nello Sprint successivo.

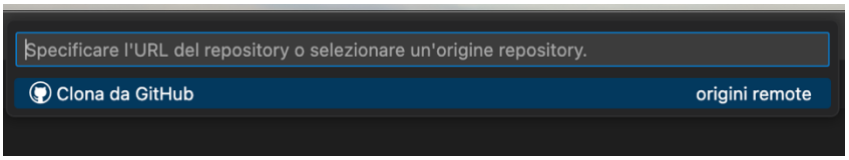
Ogni giorno tutto il Team faceva un Daily Scrum ovvero una riunione su Microsoft Teams in cui ci aggiornavamo sul lavoro che era stato svolto fino a quel momento e ci confrontavamo su eventuali dubbi. Il giorno in cui il membro che ricopre il ruolo di Developer aveva ultimato le task previste in quello Sprint si svolgeva lo Sprint review, ovvero si revisionava e si discuteva l'andamento generale del lavoro, le tempistiche e il coordinamento del team di lavoro, si testava ciò che era stato implementato, si procedeva con la chiusura del suddetto Sprint e si aggiornava la board di Github (non sempre però chi ricopriva il ruolo di developer eseguiva il commit su GitHub). Invece per quanto riguarda lo Sprint retrospective ragionavamo su cosa potesse essere migliorato negli sprint futuri. In ultima, definivamo quali dovevano essere i ruoli per lo sprint successivo e dal giorno seguente ognuno si occupava della sua mansione.

MANUALE DELLO SVILUPPATORE

Per la visualizzazione del progetto è necessaria la sua importazione in VisualStudioCode, che è uno degli IDE consigliati per lo sviluppo implementativo. Per fare ciò andiamo su GitHub selezioniamo la nostra repository, clicchiamo su code e facciamo un copia incolla dell'URI.



Dopodiché:
- Apriamo VisualStudioCode e selezioniamo "Clone repository GIT"
- Clicchiamo "Clona da GitHub" e inseriamo l'URI copiato



A questo punto si aprirà il progetto su VSCode, aprendo il terminale e posizionandosi sulla cartella del progetto digitiamo il seguente comando:
"mvn clean compile war:exploded gwt:devmode"
In questo modo si aprirà la nostra applicazione web.
Per quanto riguarda la parte del pagamento, per far si che funzioni bisogna eseguire i seguenti passi:

- Estrarre il file RAR "naive-payment-provider" dalla cartella di destinazione del progetto (dopo averla clonata)
- Aprire la cartella estratta "naive-payment-provider" in una nuova finestra di VSCode
- Digitare il comando da terminale "mvn spring-boot:run"
- Avviato il server con il comando precedente, una volta in ascolto è possibile dalla pagina di pagamento del progetto digitare le credenziali per pagare