VezGammon - Report finale Team 1

Progetto di Ingegneria del Software

Lorenzo Peronese - Scrum Master

13 gennaio 2025

Indice

1 Descrizione del prodotto

1.1 Scope

Per il progetto del corso di Ingegneria del Software, il team ha scelto di lavorare a VezGammon™, un'applicazione web di backgammon. Il progetto ha diversi riferimenti alla città di Bologna, in cui è nato e si è sviluppato: in primis ovviamente il nome (*"vez"* viene usato a Bologna come intercalare e significa amico, fratello); inoltre una volta nel sito il cursore diventa un simpatico tortellino e la board utilizza i colori rosso e blu, rappresentativi della città.

L'applicazione offre un'esperienza di gioco completa con diverse modalità: gli utenti possono sfidarsi in partite locali sullo stesso dispositivo, mettere alla prova le proprie abilità contro intelligenze artificiali di vari livelli di difficoltà e confrontarsi online con altri giocatori attraverso un sistema di matchmaking o mediante link di invito diretti. È inoltre possibile organizzare tornei a quattro partecipanti, combinando liberamente giocatori reali e agenti intelligenti.

Il sistema di classificazione si basa sul metodo Elo, ampiamente utilizzato negli scacchi. Ogni nuovo account parte da una base di 800 punti, che vengono poi aggiornati dopo ogni partita online o torneo. Il calcolo dell'aggiornamento tiene conto di diversi fattori: l'esito della partita, l'utilizzo del dado double e il punteggio Elo di entrambi i giocatori. Questo sistema garantisce una competizione equilibrata e permette di mantenere una classifica globale, dove ogni giocatore può aspirare a raggiungere le posizioni più alte.

VezGammon™ include anche ricche funzionalità social e di progressione. Gli utenti possono consultare (e condividere sui social) in ogni momento statistiche dettagliate delle proprie partite, visualizzare grafici che mostrano l'andamento del proprio Elo nel tempo e analizzare i match recenti. Il sistema premia inoltre i giocatori con speciali badge al raggiungimento di specifici obiettivi, come vincere un determinato numero di partite o raggiungere certi livelli di punteggio Elo, aggiungendo un ulteriore elemento di coinvolgimento e progressione al gioco.

1.2 Backlog

NAME	SPRINT	STATUS
#3 come utente, voglio giocare single player contro il computer		Done
#100 come giocatore, voglio poter giocare una partita contro dei bot di diversa difficoltà	sprint2	Done
#26 come creatore di tornei, voglio essere in grado di creare, amministrare tornei	sprint4	Done
#16 come utente, voglio giocare contro altri utenti		Done
#98 come giocatore, voglio poter giocare una partita in locale	sprint2	Done
#99 come giocatore, voglio poter giocare una partita online	sprint3	Done
#26 come creatore di tornei, voglio essere in grado di creare, amministrare tornei	sprint4	Done
#185 come utente, voglio poter personalizzare la mia esperienza di gioco		Done
#24 come giocatore, voglio avere un'interfaccia front-end per navigare all'interno del sito	sprint2	Done
#28 come giocatore, voglio poter scegliere vari temi grafici e ambientazioni	sprint3	Done
#164 come utente, vorrei poter usare il sito anche da cellulare	sprint4	Done
#38 come studente, voglio poter accedere a risorse di addestramento	sprint3	Done
#186 comunicazione e interazion con altri utenti		Done
#37 come giocatore, socievole voglio poter comunicare con i miei avversari in chat	sprint3	Done
#99 come giocatore, voglio poter giocare una partita online	sprint3	Done
#39 come influencer, voglio essere in grado di condividere i miei progressi sui social media	sprint3	Done
#40 come utente, voglio ricevere le notifiche dal sito	sprint4	Done
#187 analisi delle performance		Done
#33 come giocatore, voglio ricevere dei badge digitali come ricompense	sprint3	Done
#34 come studente, voglio essere in grado di analizzare le partite giocate	sprint3	Done
#25 come utente, voglio tener traccia dei miei progressi nel gioco	sprint3	Done
#18 come utente, voglio poter avere un profilo per tenere traccia delle partite giocate e delle statistiche	sprint1	Done

Figura 1: Backlog di Taiga di Vezgammon™

1.3 UML

1.3.1 Casi d'uso

1.3.2 Deployment

1.3.3 Classi

1.3.4 Tornei

2 Processo Scrum

2.1 Sprint 0

Vedi 3.2

2.2 Sprint 1

2.2.1 Goal

2.2.2 Backlog

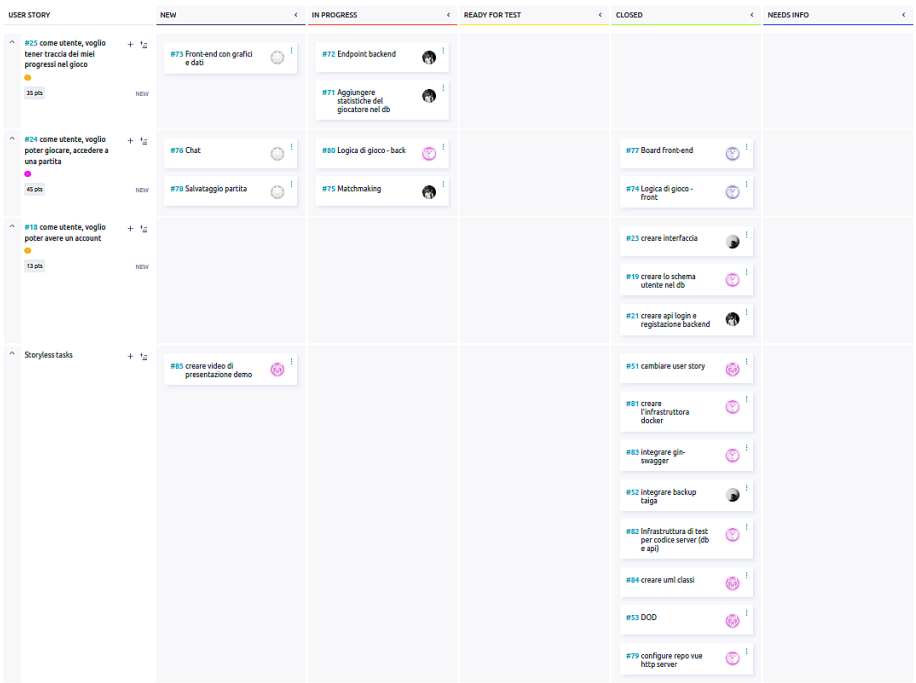

Figura 2: Backlog del primo sprint

2.2.3 Incremento

2.2.4 Definition of Done

2.2.5 Esempio di test fatti

2.2.6 Burndown

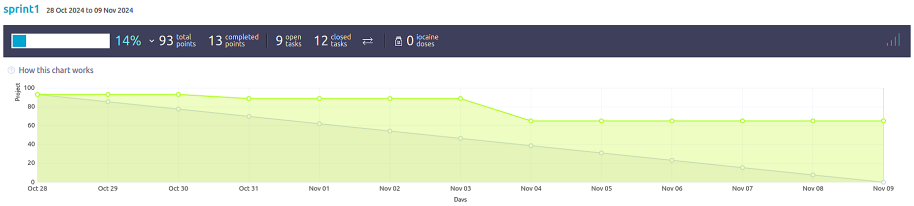

Figura 3: Burndown del primo sprint

2.2.7 Retrospettiva

2.3 Sprint 2

2.3.1 Goal

2.3.2 Backlog

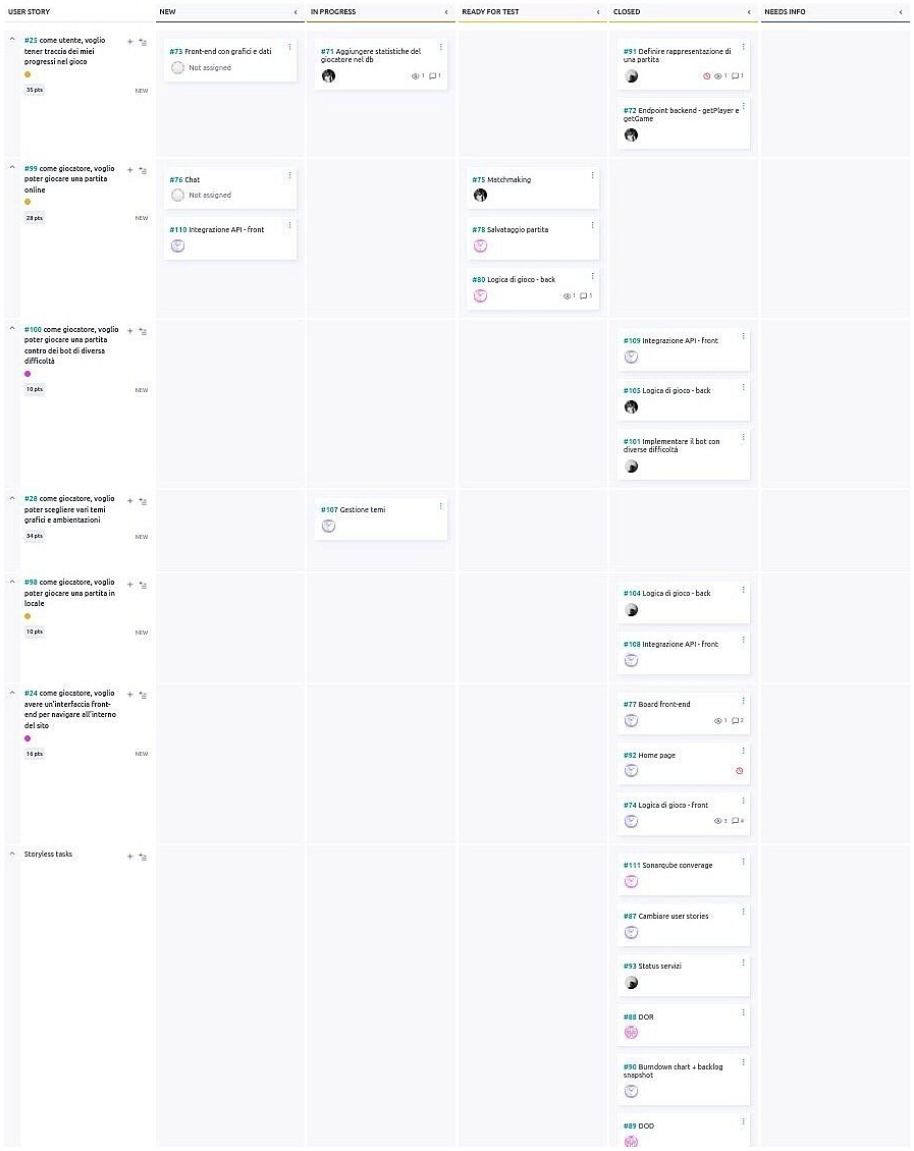

Figura 4: Backlog del secondo sprint

2.3.3 Incremento

2.3.4 Definition of Done

2.3.5 Esempio di test fatti

2.3.6 Burndown

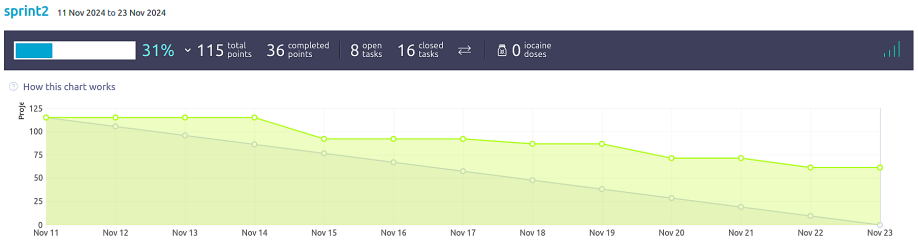

Figura 5: Burndown del secondo sprint

2.3.7 Retrospettiva

2.4 Sprint 3

2.4.1 Goal

2.4.2 Backlog

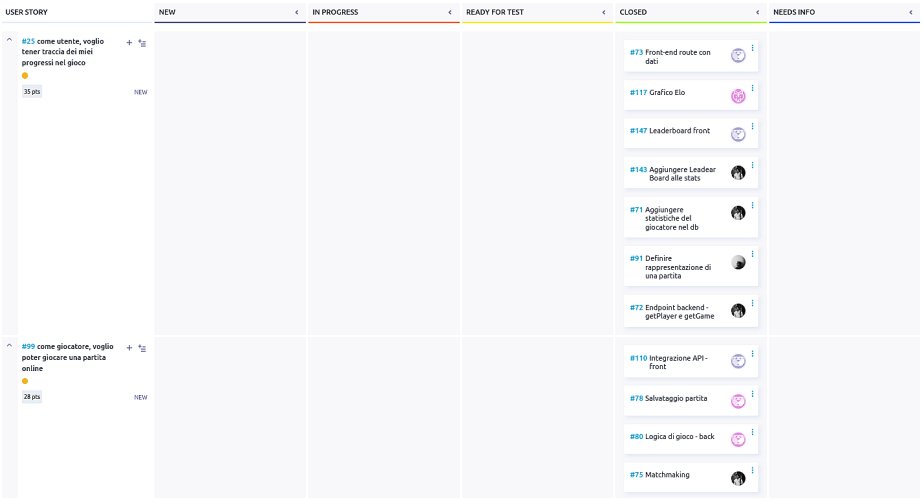

Figura 6: Backlog del terzo sprint - parte 1

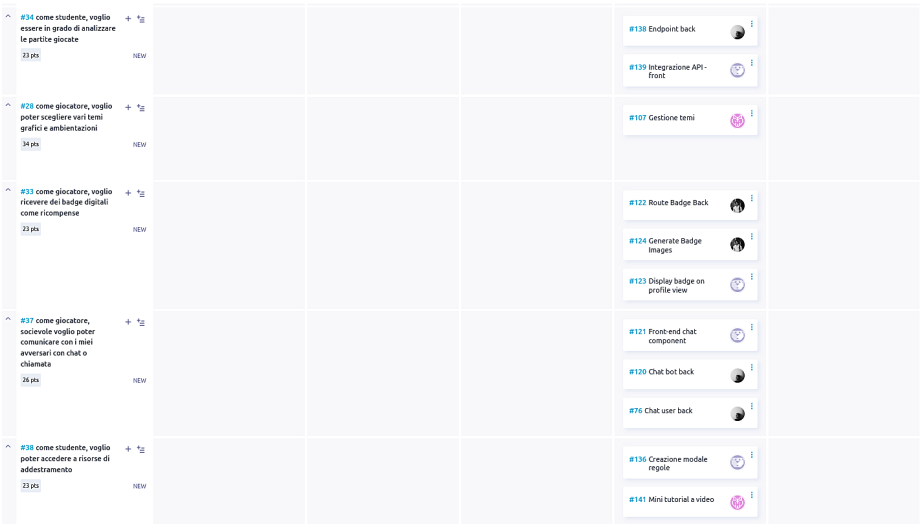

Figura 7: Backlog del terzo sprint - parte 2

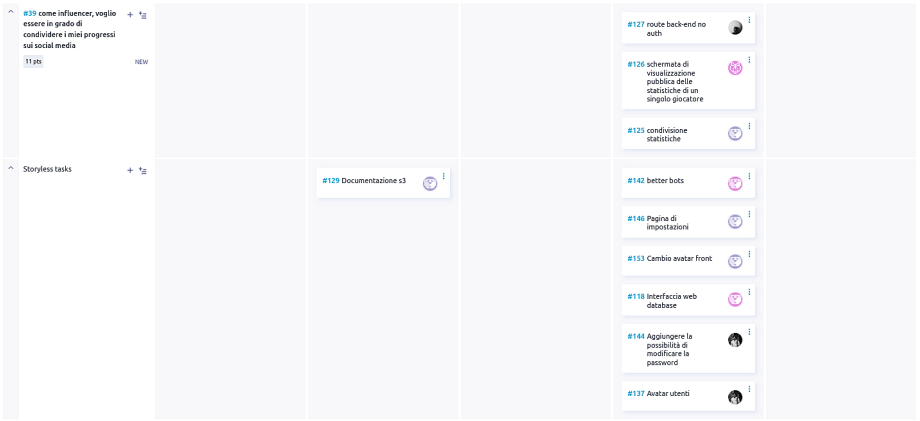

Figura 8: Backlog del terzo sprint - parte 3

2.4.3 Incremento

2.4.4 Definition of Done

2.4.5 Esempio di test fatti

2.4.6 Burndown

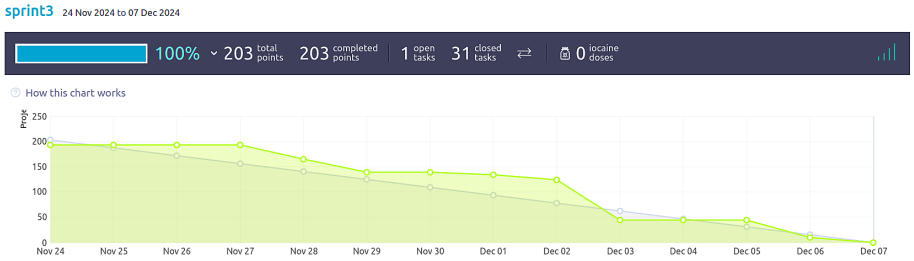

Figura 9: Burndown del terzo sprint

2.4.7 Retrospettiva

2.5 Sprint 4

2.5.1 Goal

2.5.2 Backlog

Figura 10: Backlog del quarto sprint - parte 1

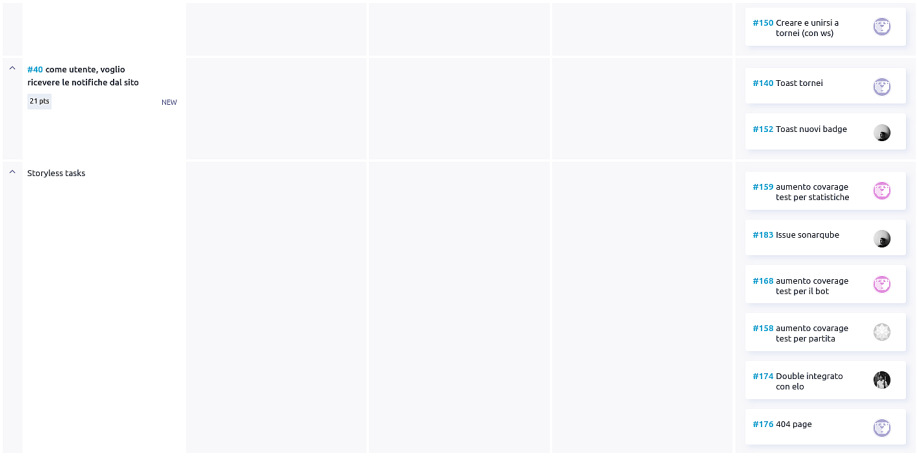

Figura 11: Backlog del quarto sprint - parte 2

2.5.3 Incremento

2.5.4 Definition of Done

2.5.5 Esempio di test fatti

2.5.6 Burndown

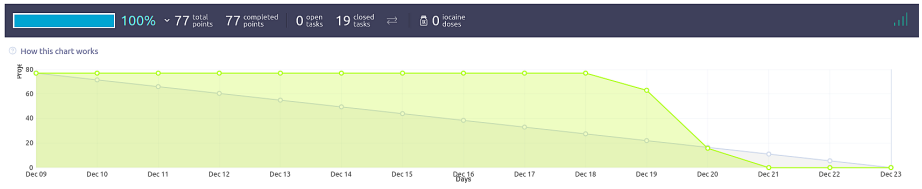

Figura 12: Burndown del quarto sprint

2.5.7 Retrospettiva

2.6 Release sprint

3 Descrizione del processo

Il team ha deciso di comune accordo e su richiesta del *Product Owner* di dedicare al progetto quattro sprint di due settimane ciascuno, in aggiunta allo sprint 0, dedicato alle attività di teambuilding e alla familiarizzazione con l'ambiente di sviluppo, e al release sprint, in cui il focus è andato sulla risoluzione dei bug, qualità del codice e stesura di questo report finale.

3.1 Il team

- **Product Owner**: Diego Barbieri
- **Scrum Master**: Lorenzo Peronese
- **DevOps**: Samuele Musiani
- **Developer frontend**: Emanuele Argonni
- **Developer backend**: Fabio Murer
- **Developer backend**: Omar Ayache

Il team è composto da sei membri ed è la prima volta che lavoriamo tutti insieme su un progetto di questa portata. Sebbene non ci conoscessimo tutti a vicenda inizialmente, alcuni di noi avevano già collaborato durante precedenti attività, sia universitarie che non. Durante gli sprint abbiamo avuto modo di approfondire la conoscenza reciproca in un contesto diverso dal solito, rafforzando il rapporto e favorendo una collaborazione più efficace.

Prima di iniziare lo sprint 0, il team si è riunito di persona per definire i ruoli; questi sono stati rispettati per la maggior parte, ma con un approccio flessibile: lo Scrum Master e il Product Owner hanno aiutato nello sviluppo del frontend, mentre il DevOps ha supportato sia il frontend che il backend quando necessario. Questa versatilità si è rivelata vincente e ha permesso di ottimizzare il lavoro e affrontare le sfide che si sono presentate in modo più efficace.

3.2 Teambuilding

3.3 Gitinspector

3.4 Strumenti di comunicazione

Per comunicare tra noi abbiamo deciso di usare MatterMost, una piattaforma di chat e condivisione interamente open-source. Come per gli altri servizi, anche MatterMost è stato self-hostato sotto il dominio vezgammon.it per avere un maggiore controllo sui dati. Questa scelta ha permesso al team di gestire in autonomia l'infrastruttura di comunicazione, evitando dipendenze da servizi esterni e assicurando la personalizzazione dell'ambiente in base alle necessità del progetto.

MatterMost è stato utilizzato per la comunicazione quotidiana, la condivisione di aggiornamenti e l'organizzazione degli incontri di persona, che si sono svolti con frequenza. Gli incontri hanno avuto luogo presso il laboratorio del gruppo ADMstaff, situato nel seminterrato del Dipartimento di Informatica, in Mura Anteo Zamboni 7.

Il laboratorio è diventato il punto di riferimento per il team, che vi si è riunito interi pomeriggi per daily scrum, sessioni di pair programming e discussioni sull'avanzamento generale del progetto. Questa modalità di lavoro ha favorito una collaborazione continua e diretta, contribuendo significativamente alla coesione del gruppo e al progresso del progetto.

Come strumento di comunicazione secondario è stato usato Telegram per comunicazioni meno "ufficiali" e sopratutto per ricevere notifiche relative allo stato dei vari servizi tramite *Status* (vedi 5.1.7).

3.5 Uso di LLM

3.6 SonarQube

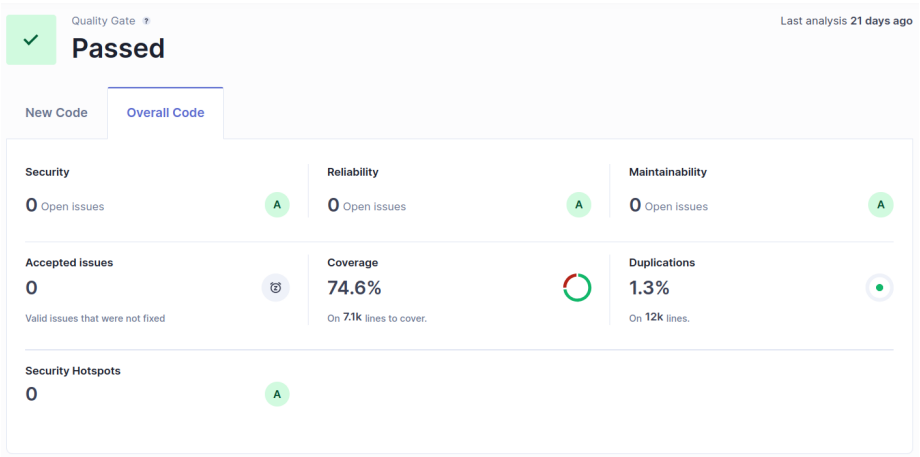

Figura 13: Vista generale di SonarQube, che mostra un riepilogo delle metriche del progetto, tra cui copertura del codice, vulnerabilità, codice duplicato e molto altro.
L'obiettivo di una copertura del 50% è stato abbondantemente superato; tuttavia le aree di codice meno coperte dai test riguardano principalmente le API e le parti che fanno uso dei web socket, in quanto più difficili da testare.

Figura 14: Grafico delle issue rilevate nel tempo: c'è un picco verso il 15 novembre, quando abbiamo terminato di self-hostare il tool e risolvere i problemi che ne sono derivati; per il resto dello sviluppo il numero delle issue è stato mantenuto sotto controllo e i problemi sono stati risolti a intervalli regolari.

3.7 Retrospettiva finale con Essence

4 Demo 3 min

Link qui

5 Artefatti realizzati ?? Software utilizzato

Per il progetto sono stati realizzati e resi disponibili diversi artefatti, sia di prodotto che di processo. Tutto l'ambiente è stato self-hostato per avere maggior controllo sull'infrastruttura e avere meno dipendenze possibili con l'esterno. Questo ha inizialmente richiesto un notevole sforzo ma a lungo termine sono emersi i benefici di questa scelta.

5.1 Artefatti di processo

5.1.1 GitLab

GitLab è stato utilizzato per la gestione del codice sorgente e per il versioning. Sono stati utilizzati i tag per denotare il Minimun Value Product di ogni sprint. Il team ha qui configurato il repository, gestito le merge request e segnalato e risolto le issue.
Il repository è accessibile all'indirizzo `gitlab.vezgammon.it`

5.1.2 Taiga

Taiga è stato scelto per la gestione del project management. È stato utilizzato per tracciare le attività, gestire il backlog (vedi 1.2), definire le storie utente e monitorare lo stato di avanzamento del progetto durante gli sprint.
Il progetto Taiga è accessibile all'indirizzo `taiga.vezgammon.it`

5.1.3 MatterMost

Come precedentemente descritto nella Sezione 3.4,MatterMost è stato utilizzato per la comunicazione interna del team. È stata la piattaforma principale per gli scambi di messaggi, la condivisione di aggiornamenti e per accordarsi sugli incontri quotidiani.
MatterMost self-hosted è accessibile all'indirizzo `mattermost.vezgammon.it`

5.1.4 Jenkins

Jenkins è stato utilizzato come *core* per l'automazione del flusso di lavoro tramite CI/CD e dei processi di build e deployment. Il team ha configurato job automatici per costruire, testare e distribuire l'applicazione.
Jenkins self-hosted è accessibile all'indirizzo `jenkins.vezgammon.it`

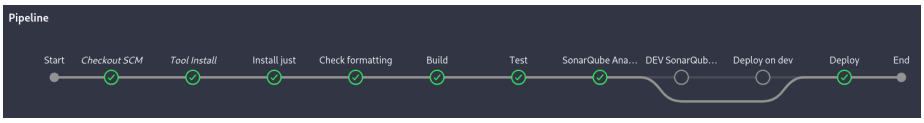

Figura 15: Pipeline di Jenkins: viene lanciata a ogni evento push su GitLab e controlla formattazione, build e test; sui commit delle branch *main* e *develop* viene anche eseguito il controllo statico del codice con SonarQube e il deploy sul relativo server, di produzione o di sviluppo.

5.1.5 SonarQube

Come precedentemente descritto nella Sezione 3.6, SonarQube è stato utilizzato per l'analisi della qualità del codice. Il team ha configurato SonarQube per eseguire analisi statiche e rilevare eventuali problematiche riguardanti la qualità del codice, la copertura dei test e la manutenzione.
SonarQube self-hosted è accessibile all'indirizzo `sonarqube.vezgammon.it`

5.1.6 Swagger

Per la documentazione delle API implementate e il testing più rapido, il team ha scelto di utilizzare il tool *Swagger* (`https://swagger.io/`).
Vengono riportate alcune immagini che ne illustrano il funzionamento; per la documentazione completa si rimanda al file **SWAGGER-FILE**

Figura 16: API utilizzate per gestire una generica partita

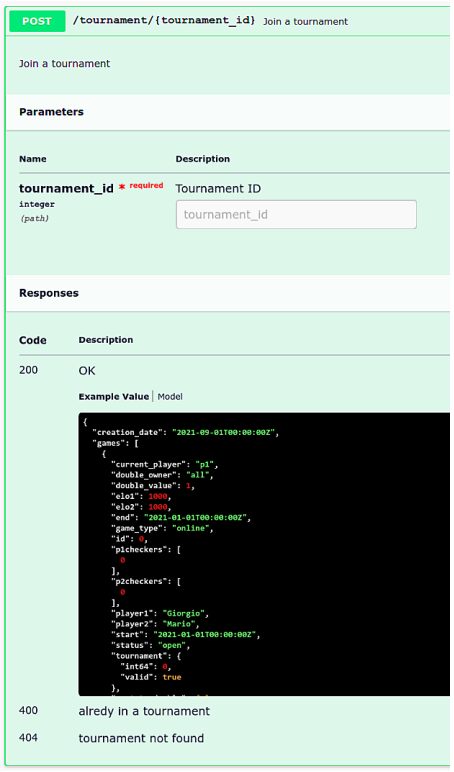

Figura 17: Nel dettaglio, l'interfaccia di Swagger per ogni API

5.1.7 Status

Il sistema di monitoraggio dei servizi è stato sviluppato per tenere traccia della disponibilità dei vari strumenti utilizzati nel progetto.
É stato inoltre sviluppato un bot Telegram per avvisare tempestivamente il team di eventuali problemi con il server e/o i singoli servizi.
Questo artefatto è accessibile all'indirizzo `status.vezgammon.it`

5.2 Artefatti di prodotto

Il team ha configurato due server principali per ospitare l'applicazione:

- **Server di produzione**: `vezgammon.it`
- **Server di sviluppo**: `dev.vezgammon.it`

Il server di produzione è stato utilizzato per la versione stabile, mentre il server di sviluppo è stato dedicato al testing, al debug e alle fasi di sviluppo continuo.