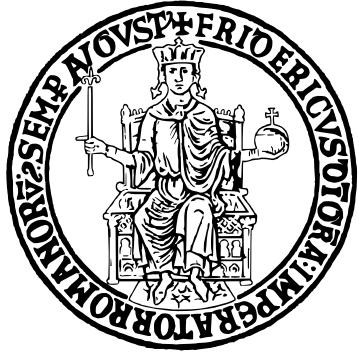


INGEGNERIA DEL SOFTWARE

# BugBoard26

---

Documentazione Progetto



**Davide Gargiulo**      **Francesco Donnarumma**  
*Matricola: N86004689*      *Matricola: N86004658*

Anno Accademico 2024/2025  
*November 11, 2025*

## *Contents*

---

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>Modellazione di tutti i casi d'uso richiesti tramite Use Case Diagrams</b>       | <b>2</b>  |
| <b>2</b> | <b>Individuazione e caratterizzazione del target degli utenti, tramite Personas</b> | <b>3</b>  |
| <b>3</b> | <b>Descrizione dei requisiti non-funzionali e di dominio</b>                        | <b>10</b> |
| 3.1      | Requisiti non-funzionali . . . . .  | 10        |
| 3.2      | Requisiti di dominio . . . . .  | 11        |

## Modellazione di tutti i casi d'uso richiesti tramite Use Case Diagrams

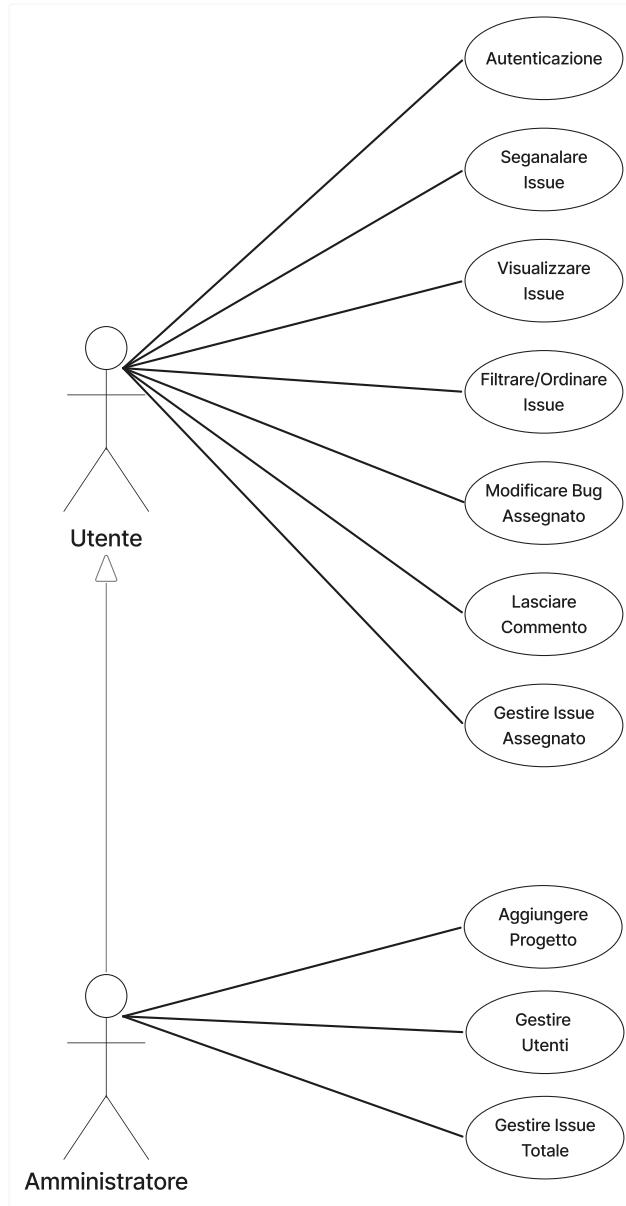
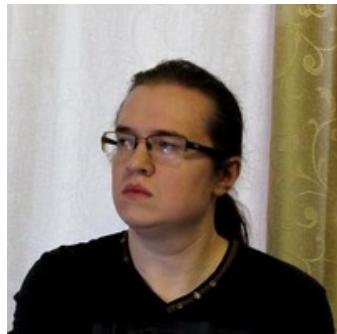


Figure 1.1: Use Case Diagram che rappresenta i casi d'uso principali dell'applicazione BugBoard26.

## II

### *Individuazione e caratterizzazione del target degli utenti, tramite Personas*

#### Alexey Kutepon (Tsoding)



|                         |  |
|-------------------------|--|
| <b>Età</b>              | 35 anni, nato in Russia, attualmente nomade digitale   |
| <b>Residenza</b>        | Nomade digitale - Attualmente in Europa dell'Est   |
| <b>Istruzione</b>       | Laurea in Computer Science conseguita in Russia nel 2012, con focus su sistemi operativi e programmazione a basso livello  |
| <b>Lavoro</b>           | Software Engineer e Content Creator dal 2016. Streamer di programmazione su Twitch e YouTube, specializzato in C, Rust, e sviluppo di linguaggi di programmazione. Creatore di progetti open source e tool di sviluppo. Conosciuto per il suo approccio diretto e senza filtri alla programmazione |
| <b>Tratti personali</b> | Pragmatico e diretto, con approccio non-nonsense al codice. Perfezionista tecnico che valorizza semplicità ed efficienza. Ironico e sarcastico, ma genuinamente appassionato di condivisione della conoscenza tecnica  |
| <b>Interessi</b>        | Appassionato di matematica e teoria dei linguaggi di programmazione. Interessato a minimalismo nel design software e filosofia Unix. Nel tempo libero esplora nuove tecnologie, contribuisce a progetti open source e sperimenta con hardware vintage  |

## Biografia

Alexey Kutepov, 35 anni, conosciuto online come Tsoding, è un software engineer e content creator nomade specializzato in programmazione di sistema e sviluppo di linguaggi. Dal 2016 condivide il suo processo creativo attraverso live coding su Twitch, dove programma in C e Rust senza script né editing.

Con un approccio pragmatico e diretto, Tsoding è diventato una figura di riferimento per developer che apprezzano la programmazione a basso livello e la filosofia del codice minimalista ed efficiente.

## Scopi e obiettivi in riferimento al sistema

- **Tracciare bug nei progetti open source personali:** Gestire efficacemente gli issue dei suoi numerosi repository GitHub, mantenendo organizzazione chiara per la community di contributor.
- **Riprodurre e documentare bug complessi:** Creare report tecnici dettagliati con stack trace, memory dumps e condizioni di riproduzione per problemi di basso livello.
- **Prioritizzare fix in base a impatto tecnico:** Categorizzare issue distinguendo tra crash critici, memory leaks, problemi di performance e feature requests.
- **Integrare bug tracking nel workflow di live coding:** Utilizzare l'applicazione durante le sessioni di streaming per mostrare alla community come affrontare sistematicamente i problemi tecnici.

## Hideo Kojima



|                         |  |
|-------------------------|--|
| <b>Età</b>              | 61 anni, nato a Tokyo  |
| <b>Residenza</b>        | Tokyo, Giappone - Quartiere Shibuya  |
| <b>Istruzione</b>       | Laurea in Economia conseguita presso l'Università di Tokyo nel 1986, con studi complementari in cinema e letteratura   |
| <b>Lavoro</b>           | Game Director, Producer e Autore dal 1986. Fondatore di Kojima Productions nel 2015. Creatore di franchise iconici come Metal Gear e Death Stranding. Visionario dell'industria videoludica con approccio cinematografico  |
| <b>Tratti personali</b> | Visionario e meticoloso, con attenzione maniacale ai dettagli narrativi e tecnici. Perfezionista, esigente ma collaborativo. Comunicatore carismatico con forte presenza mediatica   |
| <b>Interessi</b>        | Cinefilo appassionato che colleziona film e oggetti da collezione. Lettore vorace di letteratura e manga. Interessato a tecnologia, intelligenza artificiale e futuro dell'intrattenimento digitale. Attivo sui social media dove condivide le sue passioni quotidiane |

## Biografia

Hideo Kojima, 61 anni, è uno dei game director più influenti e riconosciuti a livello mondiale. Con una formazione in economia e una passione viscerale per il cinema, ha rivoluzionato l'industria videoludica dal 1986, creando opere che fondono gameplay innovativo e narrativa cinematografica.

Fondatore di Kojima Productions, è celebre per la sua attenzione maniacale ai dettagli e per la sua visione autoriale che ha ridefinito i confini tra videogiochi, cinema e arte interattiva.

## Scopi e obiettivi in riferimento al sistema

- **Mantenere la visione artistica del progetto:** Assicurarsi che ogni bug risolto non comprometta l'esperienza narrativa e l'immersione cinematografica prevista per il gioco.
- **Coordinare team multidisciplinari internazionali:** Utilizzare l'applicazione per sincronizzare il lavoro tra programmatore, designer, artisti e tester distribuiti in diverse sedi globali.
- **Prioritizzare issue che impattano l'esperienza emotiva:** Categorizzare i bug in base al loro effetto sulla narrazione, l'atmosfera e il coinvolgimento emotivo del giocatore.
- **Documentare dettagliatamente ogni problema:** Creare report completi con contesto narrativo, screenshot, video e note precise per guidare il team verso soluzioni che rispettino la visione creativa.
- **Monitorare milestone critiche pre-release:** Tenere traccia dello stato di risoluzione degli issue in vista di demo, eventi stampa e lancio finale, garantendo standard qualitativi eccellenti.

## Michele Deserto



|                         |   |
|-------------------------|---|
| <b>Età</b>              | 35 anni, nato a Bari, residente a Milano da 8 anni  |
| <b>Residenza</b>        | Milano, Italia - Zona Porta Nuova   |
| <b>Istruzione</b>       | Diploma in Architettura Moderna conseguito a Bari nel 2009, con specializzazione in design sostenibile  |
| <b>Lavoro</b>           | Content Creator dal 2015, specializzato in recensioni videoludiche e tecnologia su YouTube e Twitch. Appassionato di giochi indie, RPG e hardware. Assunto da poco come bug tester dalla FromSoftware |
| <b>Tratti personali</b> | Flemmatico e riflessivo, con approccio creativo ai problemi. Calmo, empatico e buon ascoltatore con la sua community  |
| <b>Interessi</b>        | Oltre ai contenuti digitali, ama viaggiare scoprendo nuove culture e tradizioni. Appassionato di architettura e fotografia urbana. Nel tempo libero cucina e suona la chitarra                        |

## Biografia

Michele Deserto, 35 anni, è un content creator milanese specializzato in videogiochi e tecnologia. Con un diploma in architettura moderna e un approccio flemmatico e creativo, dal 2015 produce contenuti su YouTube e Twitch dedicati a giochi indie, RPG e innovazioni tech. Appassionato viaggiatore, ama scoprire nuove culture e tradizioni, che spesso ispirano i suoi contenuti digitali.

## Scopi e obiettivi in riferimento al sistema

- **Diminuire la presenza di bug nei giochi recensiti:** Garantire che i giochi presentino meno bug tecnici, migliorando l'esperienza di gioco per i suoi spettatori.
- **Tracciare efficacemente i bug critici:** Utilizzare l'applicazione per categorizzare e prioritizzare gli issue in base alla loro gravità e impatto sul gameplay.
- **Collaborare con il team di sviluppo:** Condividere report dettagliati e screenshot attraverso l'applicazione per facilitare la comunicazione con i programmatore.
- **Monitorare lo stato di risoluzione:** Tenere traccia dei bug segnalati e verificare quali vengono risolti nelle patch successive.
- **Organizzare i test per piattaforme diverse:** Gestire gli issue separando i bug per console, PC e altre piattaforme su cui vengono testati i giochi.

## Yan Chernikov (TheCherno)



|                         |   |
|-------------------------|---|
| <b>Età</b>              | 29 anni, nato in Australia, residente negli Stati Uniti   |
| <b>Residenza</b>        | Los Angeles, California - USA   |
| <b>Istruzione</b>       | Laurea in Computer Science conseguita in Australia nel 2017, con specializzazione in computer graphics e game engine development  |
| <b>Lavoro</b>           | Software Engineer e Content Creator dal 2012. Creatore della popolare serie YouTube su C++ e game engine development. Lead Engine Programmer con esperienza in AAA studios. Fondatore di Hazel Engine, un game engine educativo open source                     |
| <b>Tratti personali</b> | Metodico e didattico, con approccio sistematico all'insegnamento della programmazione. Paziente e chiaro nelle spiegazioni, ma esigente sulla qualità del codice. Entusiasta e motivante con la sua community   |
| <b>Interessi</b>        | Appassionato di architettura software e design patterns. Interessato a rendering graphics, fisica e ottimizzazione performance. Nel tempo libero sperimenta con nuove tecnologie grafiche, contribuisce alla community open source e gioca a giochi competitivi |

## Biografia

Yan Chernikov, 29 anni, conosciuto come TheCherno, è un software engineer e educator specializzato in C++ e game engine development. Dal 2012 produce contenuti educativi di alta qualità su YouTube, rendendo accessibili concetti complessi di programmazione grafica e architettura di game engine.

Con esperienza in studi AAA e una passione genuina per l'insegnamento, TheCherno ha costruito una delle community più rispettate per developer che vogliono comprendere profondamente come funzionano i motori grafici moderni.

---

## Scopi e obiettivi in riferimento al sistema

- **Gestire issue del progetto Hazel Engine:** Organizzare e prioritizzare bug report e feature requests della community per il suo game engine educativo open source.
  - **Documentare problemi di rendering e graphics:** Creare report dettagliati per bug complessi relativi a shader, pipeline grafiche e ottimizzazioni rendering con screenshot e frame analysis.
  - **Coordinare sviluppo tra tutorial series:** Utilizzare l'applicazione per pianificare quali issue affrontare durante le prossime puntate della serie YouTube, mantenendo coerenza didattica.
  - **Categorizzare per subsystem dell'engine:** Organizzare issue per moduli specifici (renderer, physics, ECS, scripting) facilitando la navigazione per contributor e studenti.
  - **Tracciare problemi cross-platform:** Monitorare bug specifici per Windows, macOS e Linux, assicurando che Hazel Engine funzioni correttamente su tutte le piattaforme.
-

# III

## *Descrizione dei requisiti non-funzionali e di dominio*

3.1

### Requisiti non-funzionali

| Codice      | Requisito   |
|-------------|---|
| <b>SR01</b> | Il Sistema deve essere realizzato in modo distribuito prevedendo almeno due macro-componenti indipendenti (back-end/front-end).                                 |
| <b>SR02</b> | Il back-end deve esporre interfacce di programmazione accessibili via rete.   |
| <b>SR03</b> | Il back-end dovrebbe essere distribuito utilizzando tecnologie di containerizzazione.   |
| <b>SR04</b> | Il front-end deve essere un'interfaccia utente che si appoggia ai servizi offerti dal back-end esclusivamente attraverso la rete.                               |
| <b>SR05</b> | La parte front-end deve essere realizzata come applicazione web app spa.  |
| <b>SR06</b> | La logica applicativa e la persistenza dei dati non devono essere gestite esclusivamente tramite servizi esterni.   |
| <b>SR07</b> | Il Sistema deve essere scritto in un linguaggio di programmazione che supporta il paradigma orientato agli oggetti.   |
| <b>SR08</b> | Il Sistema deve essere conforme al GDPR per la protezione dei dati personali degli utenti.  |
| <b>SR09</b> | Nel caso in cui il Sistema venga distribuito attraverso store di terze parti, deve essere conforme alle policy sui contenuti per sviluppatori di questi ultimi. |

## Requisiti di dominio

| Codice      | Requisito  |
|-------------|--|
| <b>DR01</b> | Una Issue deve necessariamente contenere un titolo e una descrizione. Opzionalmente anche una priorità, un'immagine e una tipologia. |
| <b>DR02</b> | Un utente normale può agire solo su Issue a lui assegnate, mentre un admin può agire su tutte le Issue disponibili.                  |
| <b>DR03</b> | Ogni Issue ha un ciclo di vita definito da stati e transizioni irreversibili. to-do →in progress →done.                              |
| <b>DR04</b> | Le password degli utenti devono essere memorizzate in formato hash utilizzando algoritmi crittografici sicuri.                       |
| <b>DR05</b> | La priorità di un'Issue deve assumere valori da un insieme predefinito.  |
| <b>DR06</b> | Solo gli amministratori possono creare nuovi account utente nel sistema.   |
| <b>DR06</b> | Un'Issue può essere assegnata a un solo utente alla volta.   |
| <b>DR07</b> | L'account amministratore di default deve essere configurato con credenziali sicure.  |