Ticketing System

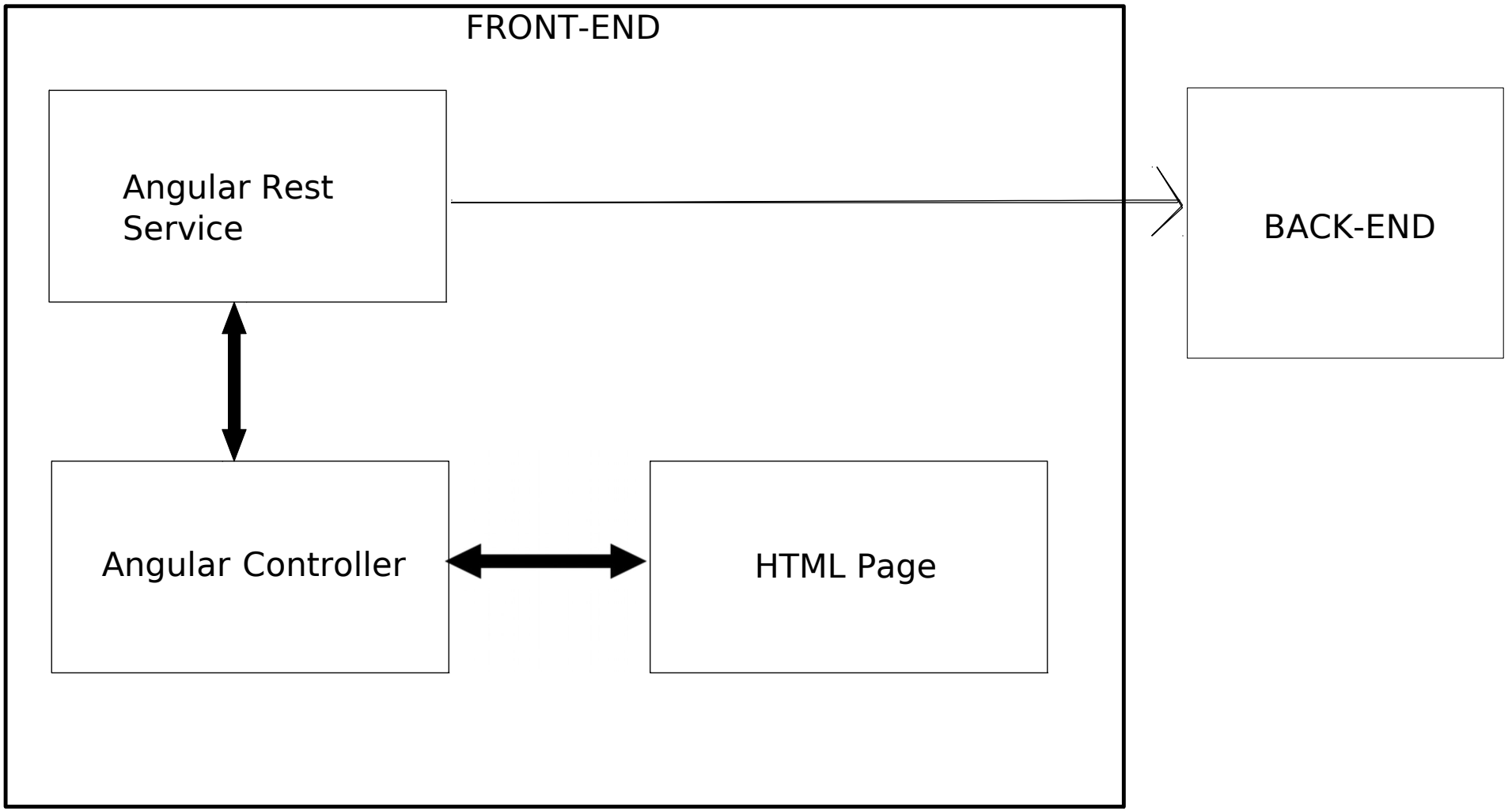
- **Team:**
  - Camarda Emanuele
  - Catullo Gentilcore Maria Josè
  - Celi Melissa
  - Montesano Federica
  - Tuni Gabriele

Architettura

- Applicazione divisa in front-end e back-end
- Applicazione REST
- Pattern implementato: MVC per il back-end, MVW per il front-end

Tecnologie usate

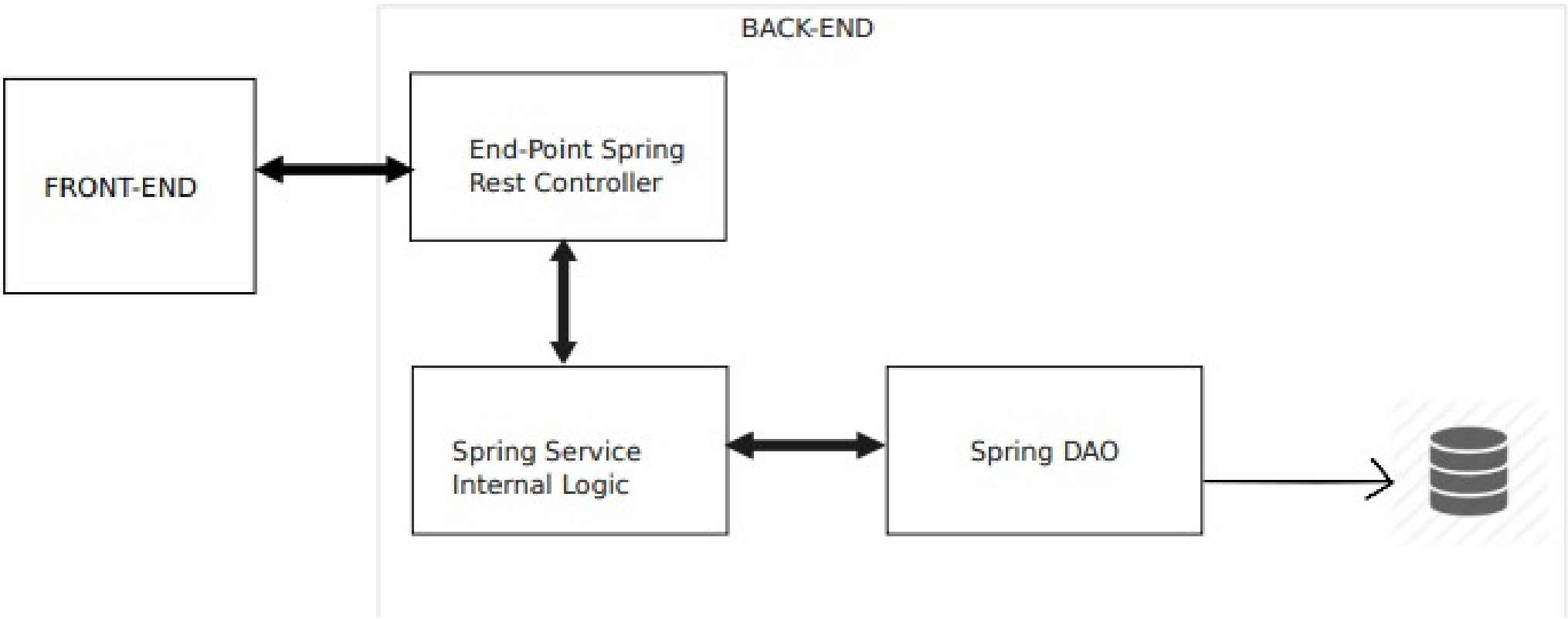
Front-End


FRONT-END
Angular Rest Service
BACK-END
Angular Controller
HTML Page

AngularJS

- Architettura MVW
- Interfaccia web che implementa i criteri della single page application
- Sfruttamento del formato JSON per comunicare con l’applicazione REST del back-end
- Session Storage per fornire autenticazione e permessi

Back-End


BACK-END
FRONT-END
End-Point Spring
Rest Controller
Spring Service
Internal Logic
Spring DAO

Database

- Utilizzo di PostgreSQL
- Database relazionale
- Struttura dei dati
  naturalmente definita
- Maggiore familiarità con i database relazionali

Spring

- Facilità nello sviluppo
- Supporto ad Hibernate
- Architettura MVC con Spring Controller, Spring Service e Dao (repository)
- Supporto al parsing di oggetti JSON

Postman

- Fase di testing delle funzionalità non ancora incorporate nel Front-End
- Simulazione richieste HTTP

Agile – problemi riscontrati e risolti

- Sprint time-boxed: 2 settimane
- Difficoltà iniziale nel decidere l'effort da porre ad ogni sprint
- Difficoltà iniziale nel decidere quanta documentazione produrre

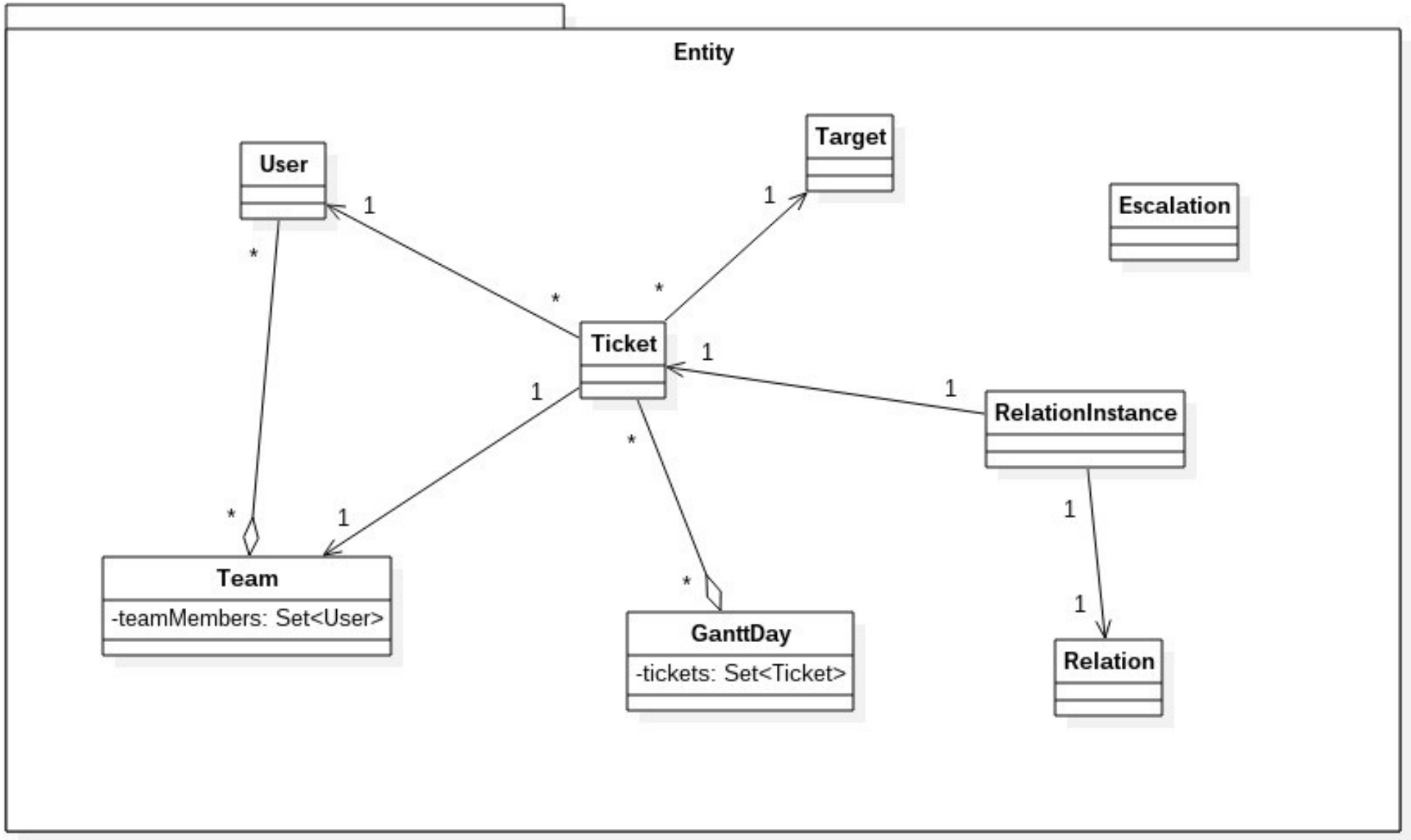
Ticketing System

- Sistema di gestione di **ticket** relativi un determinato **target**, che fornisca supporto agli **utenti**, sia che siano customer, amministratori di sistema o componenti di un **team** di lavoro.
- Ticket:
  - Titolo → oggetto del ticket
  - Descrizione → descrizione del problema
  - Team → team a cui è assegnata la risoluzione del ticket
  - Target → prodotto o servizio a cui è associato il ticket
  - Stato → stato del ticket nel sistema
  - Utente → utente che ha creato il ticket
  - Categoria →ambito del ticket

Ticketing System

- Team:
  - Leader → responsabile del team
  - Coordinatore → gestisce il team
  - Membri → componente generico del team
- Target:
  - Nome
  - Versione
  - Descrizione
- Utente:
  - Nome e Cognome
  - Username e Password
  - Email

Class Diagram



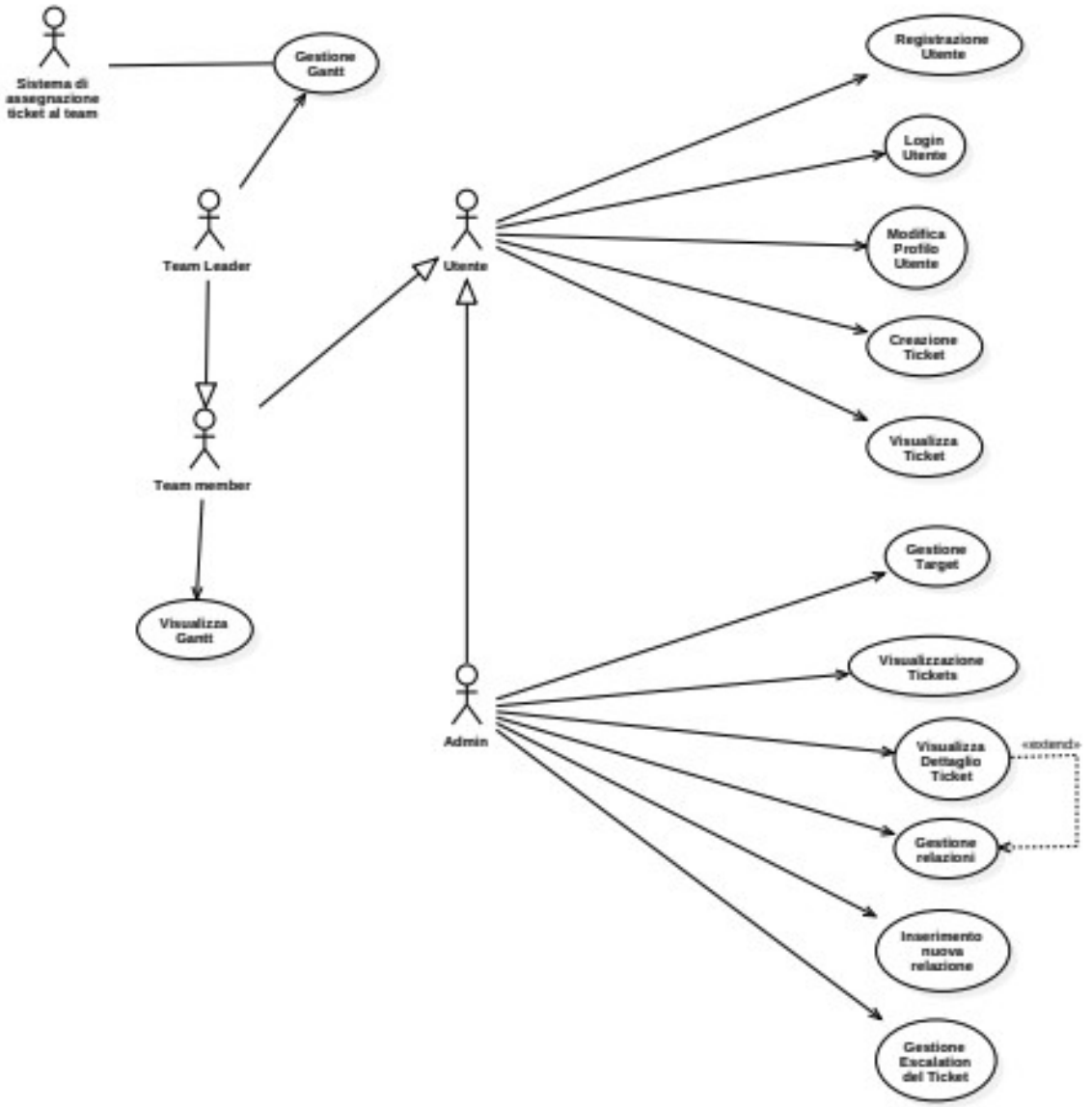
Attori del Sistema

- **Utente**: utente generico del sistema, può eseguire le seguenti operazioni:
  - Registrarsi all'applicazione
  - Modificare il proprio profilo
  - Creare un ticket
  - Visualizzare lo stato dei propri ticket
- **Team Member**: è un membro di un team e un utente in particolare. Può eseguire le seguenti operazioni aggiuntive:
  - Visualizzare il Gantt associato al suo team

Attori del Sistema

- **Team Leader**: gestisce un team ed è un team member in particolare. Può eseguire le seguenti operazioni aggiuntive:
  - Pianificare esecuzione di un ticket aggiungendolo al Gantt
- **Admin**: gestisce il sistema ed è in paricolare un utente. Può eseguire le seguenti operazioni aggiuntive:
  - Gestire i target
  - Visualizzare tutti i ticket del sistema
  - Definire nuove relazioni tra i ticket
  - Creare nuove relazioni tra due ticket
  - Gestire l’escalation dei ticket

Modello dei Casi d'Uso



Use Case: Gestione Relazioni

- 3 relazioni custom nell'applicazione:
  - **Equivalenza**

    A = B se risolvere A equivale a risolvere B
  - **Dipendenza**

    A → B se per risolvere A devo prima risolvere B
  - **Regressione**
  - A ~> B se la risoluzione di B ha generato A
- Definizione di nuove relazioni non custom con proprietà di ciclicità o meno

Flussi base e alternativi
Gestione Relazioni

- Flusso base:
  - Scegliere il primo ticket con cui creare una relazione
  - Scegliere il tipo di relazione da creare
  - Scegliere il secondo ticket con cui creare la relazione
  - Inviare la richiesta di creazione della nuova relazione
  - Relazione creata con successo

Flussi base e alternativi
Gestione Relazioni

- Flussi alternativi:
  - A1: relazione riflessiva non permessa → creazione relazione fallisce
  - A2: creazione ciclo in relazione non ciclica → creazione relazione fallisce e viene visualizzato il ciclo che si sarebbe creato

Use Case: Gestione Escalation

- I ticket in stato di attesa (pending) vengono classificati sulla base di un rank che ne determina la priorità.
- Rank = **a** * customerPriority +

  b * teamPriority + **c** * pendingTime

  ove **a**, **b** e **c** sono dei parametri configurabili dall'admin.
- Il sistema ricalcola il rank dei ticket periodicamente

Use Case: Gestione Gantt

- Un teamMember può visualizzare il gantt relativo al team di cui fa parte
- Un teamLeader può visualizzare e modificare il Gantt relativo a un team.
- In particolare può pianificare nel tempo l’esecuzione di un ticket che ancora non è stato pianificato sulla base del suo rank.

Flussi base e alternativi
Gestione Gantt

- Flusso base:
  - Scegliere un team
  - Scegliere il ticket da pianificare
  - Scegliere un data di inizio per l’esecuzione programmata del ticket
  - Scegliere una durata stimata per l’esecuzione del ticket
  - Inserimento della pianificazione nel Gantt

Flussi base e alternativi Gestione Gantt

- Flussi alternativi:
  - A1: Non vi è disponibilità nei giorni in cui si sta cercando di pianificare l'esecuzione del ticket → flusso termina con la visualizzazione dei giorni senza disponibilità
  - A2: Si vuole pianificare l'esecuzione di un ticket in giorni antecedenti l'esecuzione dei ticket da cui dipende → flusso termina con la visualizzazione dei ticket da eseguire prima