MagnaFrara

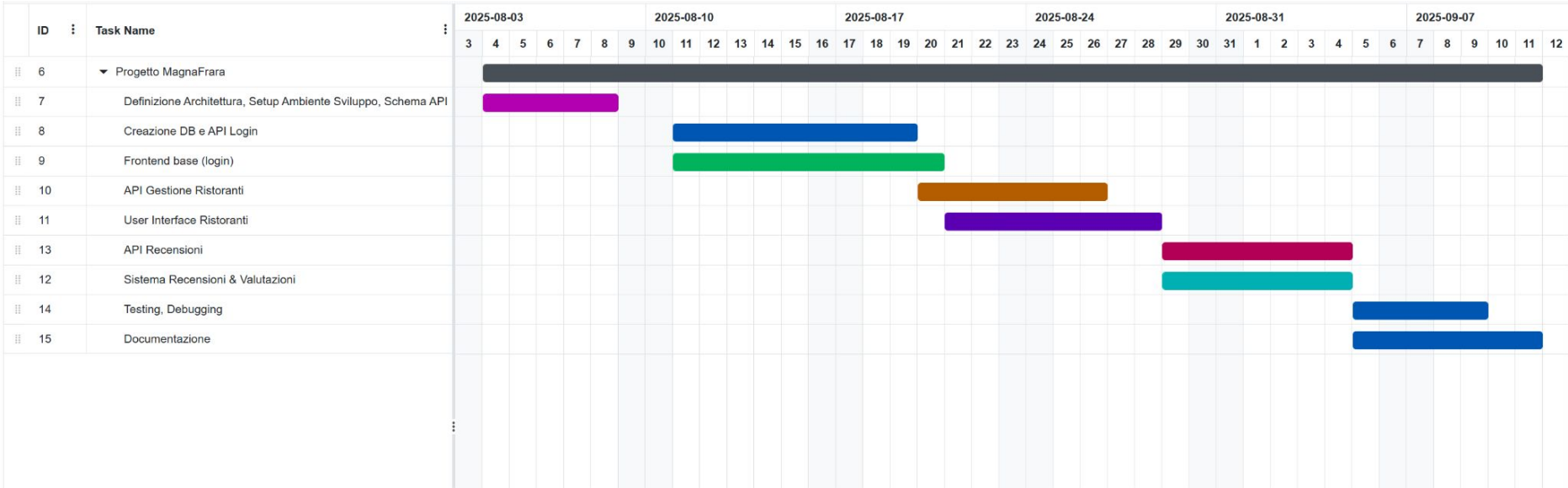
MagnaFrara è un'applicazione web dedicata alla ristorazione tipica nella città di Ferrara. La piattaforma consente agli utenti di cercare ristoranti filtrandoli per nome, genere o valutazione media, di visualizzare le informazioni principali dei locali e di fornire recensioni relative alla propria esperienza.

Allo stesso tempo, i ristoratori possono accedere con account dedicati per gestire i propri ristoranti, aggiornando le informazioni utili come orari di apertura o descrizioni.

L'obiettivo è creare una piattaforma locale che favorisca la riscoperta della tradizione culinaria ferrarese e la valorizzazione dei ristoranti che la propongono.

Segmentation Fault - Andrea Fedozzi, Andrea Gobbi, Emanuele Fava, Camilla Bellantuono

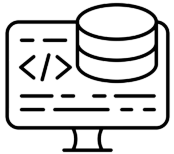
Diagramma di GANTT



Distribuzione del Team

- Coordinamento
- GANTT
- Documentazione

Project Manager
Andrea Fedozzi



- API REST
- DataBase
- Test Unitari

Backend Developer
Emanuele Fava

- User Interface con Angular
- Connessione FE e BE
- Debugging
- Demo

Frontend Developer
Andrea Gobbi

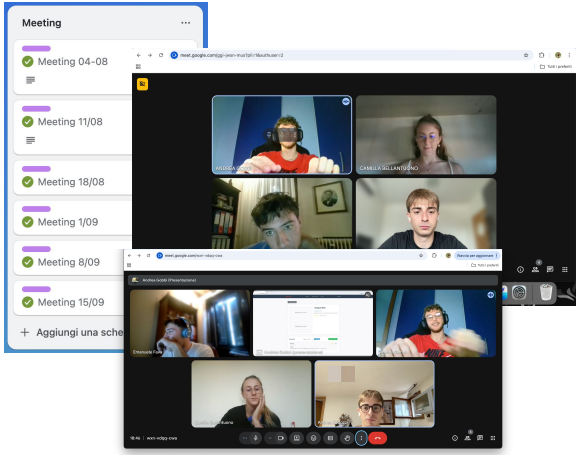
- DataBase
- Testing Generale

BE Developer/Tester
Camilla Bellantuono

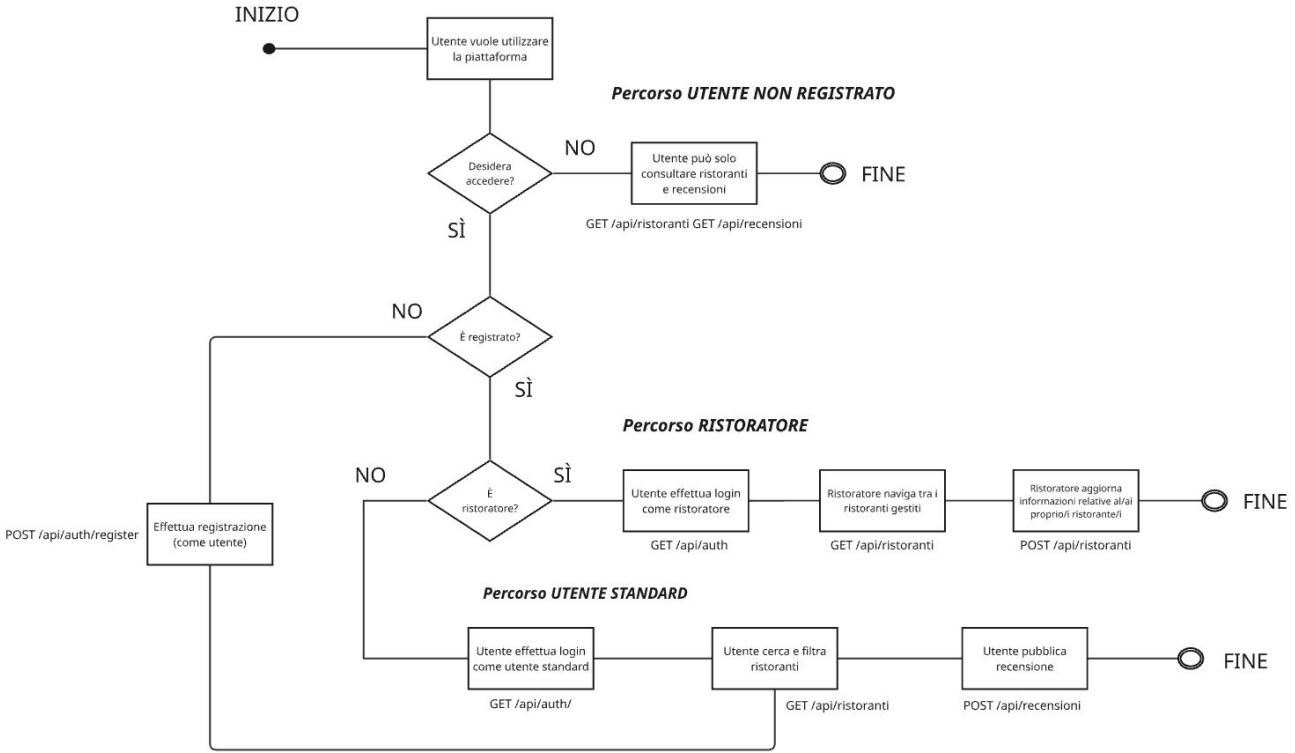
Evidenza delle cerimonie

Cerimonie di progetto

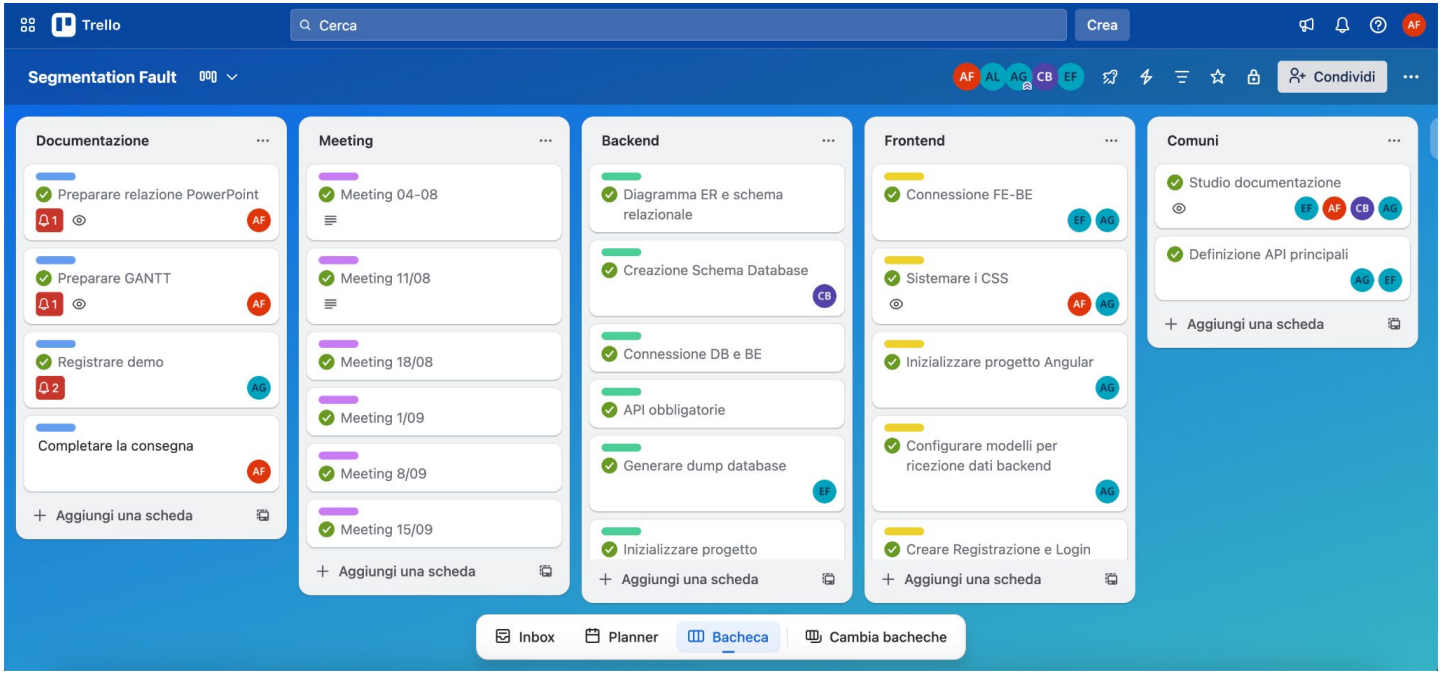
- **Meeting settimanali** → stato dell'avanzamento, assegnazione di task e gestione criticità
- **Retrospective** → raccolta feedback interni, miglioramento dei processi
- **Review** → condivisione del materiale sviluppato, coordinazione tra backend e frontend, demo
- **Planning** → definizione obiettivi settimana successiva, verifica delle tempistiche



Disegno funzionale



Trello



(https://trello.com/b/Avr7qjMt/segmentation-fault)

Test Cases

Test Case #: 1.1 **Test Case Name:** Registrazione e Login Completo **Page:** 1 of ..

System: MagnaFrara **Subsystem:** Authentication

Designed by: SegmentationFault **Design Date:** 08/09/2025

Executed by: SegmentationFault **Execution Date:** 08/09/2025

Short Description: Test completo di registrazione e login utente attraverso l'interfaccia web

Pre-conditions

Il backend è attivo su localhost:8080
Il frontend Angular è attivo su localhost:4200
Il database è vuoto o non contiene l'email test@magnafara.com
Il browser è aperto sulla pagina principale

Step	Action	Expected System Response	Pass/Fail	Comment
1	Navigare su http://localhost:4200	Il sistema reindirizza automaticamente a /account mostrando la form di login	pass	
2	Cliccare sul pulsante "Non hai un account? Registrati"	La form si trasforma in registrazione con campi: Nome, Cognome, Email, Password	pass	
3	Compilare la form: Nome="Mario", Cognome="Rossi", Email="mario.rossi@test.com", Password="password123"	I campi vengono riempiti correttamente, il pulsante "Registrati" diventa attivo	pass	
4	Cliccare su "Registrati"	Messaggio "Registrazione completata con successo!" appare, poi login automatico e redirect a /ristoranti	pass	
5	Verificare che l'header mostri "Ciao, Mario!"	L'header mostra il nome utente loggato e i link di navigazione corretti	pass	
6	Cliccare su "Account" nell'header	La pagina account mostra i dettagli utente con Nome, Email e Tipo account: "Utente"	pass	

Post-conditions

L'utente Mario Rossi è salvato nel database
L'utente è loggato nel sistema
I dati di sessione sono salvati in localStorage

Test Case #: 2.1 **Test Case Name:** Navigazione e Ricerca Ristoranti **Page:** 1 of ..

System: MagnaFrara **Subsystem:** Ristoranti

Designed by: SegmentationFault **Design Date:** 08/09/2025

Executed by: SegmentationFault **Execution Date:** 08/09/2025

Short Description: Test della funzionalità di ricerca e navigazione ristoranti

Pre-conditions

L'utente è loggato nel sistema
Esistono almeno 15 ristoranti nel database
L'utente si trova sulla pagina /ristoranti

Step	Action	Expected System Response	Pass/Fail	Comment
1	Verificare il caricamento della griglia ristoranti	La pagina mostra 9 ristoranti in griglia 3x3, con skeleton loader durante il caricamento iniziale	pass	
2	Verificare i dettagli di ogni card ristorante	Ogni card mostra: immagine, nome, tipo cucina, stelle valutazione, numero recensioni, descrizione troncata	pass	
3	Digitare "pizza" nella barra di ricerca	Durante la digitazione appare un spinner, dopo 300ms vengono filtrati solo i ristoranti con "pizza" nel nome/tipo cucina	pass	
4	Cambiare ordinamento da "Nome" a "Valutazione"	Un spinner appare brevemente, i ristoranti si riordinano per valutazione decrescente	pass	
5	Cliccare sul pulsante "↑ Crescente"	I ristoranti si riordinano per valutazione crescente con animazione	pass	
6	Cliccare su una card ristorante	Navigazione alla pagina /ristorante/{id} del ristorante selezionato	pass	
7	Testare la paginazione cliccando "Successiva"	La pagina 2 viene caricata con i successivi 9 ristoranti, l'URL rimane /ristoranti	pass	

Post-conditions

La ricerca funziona con debouncing di 300ms
L'ordinamento persiste durante la navigazione tra pagine
Le immagini hanno fallback per errori di caricamento

Funzionalità implementate

- **Utenti non registrati:**
  - Sola visualizzazione di ristoranti e recensioni
  - Filtro per nome, categoria, valutazione media
  - Filtro sulle recensioni per data e valutazione
  - Ordinamento per valutazione media o nome

- **Utenti registrati:**
  - Visualizzazione di ristoranti e recensioni, con possibilità di pubblicare recensioni
  - Filtro per nome, categoria, valutazione media
  - Filtro sulle recensioni per data e valutazione
  - Ordinamento per valutazione media o nome
  - Visualizzazione proprie statistiche e livello raggiunto

- **Ristoratori:**
  - Visualizzazione di ristoranti e recensioni, con possibilità di pubblicare recensioni (non sui propri ristoranti)
  - Filtro per nome, categoria, valutazione media
  - Filtro sulle recensioni per data e valutazione
  - Ordinamento per valutazione media o nome
  - Possibilità di modifica delle informazioni relative ai propri ristoranti

Dettaglio di una funzionalità

Pubblicazione di una recensione

- Dalla scheda ristorante (GET /api/ristoranti/{id_ristorante}) l'utente apre il form, inserisce titolo, testo e voto decimale 1.0–5.0 e invia.
- Il backend riceve la richiesta POST /api/recensioni/{id_rist} con Body { id_utente, titolo, testo, valutazione }, valida i campi (obbligatorietà di titolo e testo, range della valutazione) e verifica che l'utente sia loggato.
- Se i dati sono corretti, salva la recensione, assegna la data di pubblicazione e aggiorna le metriche del ristorante (media e numero recensioni).
- La risposta (201 Created) restituisce l'oggetto recensione completo, immediatamente visibile sia nell'elenco del ristorante (GET /api/recensioni/{id_rist}) sia nello storico dell'utente (GET /api/recensioni/utente/{id_utente}). In caso di input non valido o ristorante inesistente, vengono restituiti errori strutturati (400 VALIDATION_ERROR, 404 NOT_FOUND); per problemi imprevisti 500 INTERNAL_ERROR.
- L'utente vedrà così la propria recensione pubblicata e la media del ristorante aggiornata nel dettaglio.
  La recensione può essere anche visualizzata nella sezione "Le mie recensioni", che mostra anche livello dell'account, numero di recensioni e valutazioni assegnate.

Feedback dell’esperienza

- *“Ci siamo divisi bene i compiti, ma alla fine ci siamo aiutati a vicenda un po’ su tutto.”*
- *“Non conoscevamo Angular nè altre tecnologie che abbiamo utilizzato, ma le abbiamo imparate a conoscere lavorandoci sopra.”*
- *“È stato bello vedere il progetto prendere forma, da zero a un’applicazione completa che funziona davvero.”*
- *“Questo progetto è stato da solo più utile di tante esercizi teorici per capire davvero come si lavori in team e quale sia il processo dietro ad un’applicazione completa.”*
- *“All’inizio ci sembrava un’enormità di lavoro, ma con organizzazione e impegno siamo riusciti a terminare quello che ci eravamo prefissati.”*

- Segmentation Fault