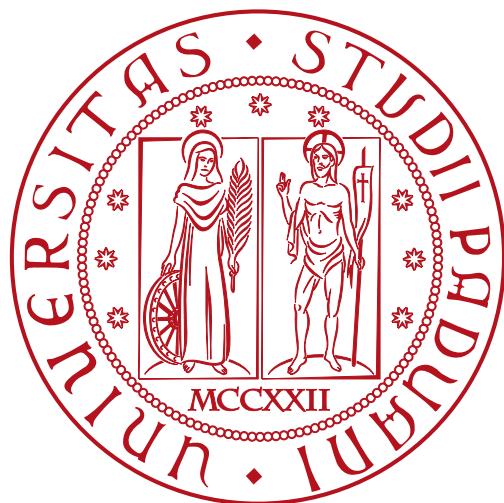


Università degli Studi di Padova

DIPARTIMENTO DI MATEMATICA "TULLIO LEVI-CIVITA"

CORSO DI LAUREA IN INFORMATICA



WebApp per attività laboratoriali di OpenDay

Tesi di laurea

25 Luglio 2025

Relatrice

Prof.ssa Ombretta Gaggi

Laureando

Orlando Virgilio Maria Ferazzani

Matricola 2058653

[Citazione qui](#)

Sommario

Il seguente elaborato descrive l'attività di tirocinio, della durata complessiva di xxx ore, svolta presso l'Università di Padova. Questa attività è stata portata avanti sotto la guida della Prof.ssa Ombretta Gaggi. Il Prof. Claudio Palazzi ha ricoperto il ruolo di tutor accademico.

L'*Università degli Studi di Padova* durante i suoi OpenDay, utilizza [WebApp](#) interattive per avvicinare i ragazzi delle scuole superiori al corso di laurea in Informatica. Queste consentono di far conoscere le basi della programmazione attraverso giochi che stimolano la logica e la creatività. Tuttavia, queste applicazioni sono spesso molto tediose da utilizzare dato il breve tempo a disposizione per le suddette attività e non sempre riescono a coinvolgere gli studenti, soprattutto chi di programmazione non ha mai intrapreso degli studi di alcun genere.

Il tirocino effettuato mira proprio a risolvere questa problematica, sviluppando una [WebApp](#) interattiva che permetta di avvicinare i ragazzi al mondo della programmazione in modo divertente e stimolante.

Ringraziamenti

Padova, Luglio 2025

Orlando Virgilio Maria Ferazzani

Indice

Introduzione	1
1.1 Motivazioni e Contesto	1
1.2 Strumenti e processi	2
1.2.1 Suddivisione del lavoro	2
1.3 Struttura del Documento	4
Scopo del tirocinio	6
2.1 Scopo del progetto	6
2.2 Obiettivi prefissati	7
2.3 Prodotti attesi	9
Analisi dei Requisiti	11
3.1 Tecnologie utilizzate	11
3.1.1 Typescript	12
3.1.2 Axios	12
3.1.3 Firebase	12
3.1.4 GitHub	12
3.1.5 ShadCN	12
3.1.6 NextJS	12
3.1.7 TailwindCSS	12
3.2 Obiettivo del progetto	13
3.3 Utenti Target	13
3.4 Casi d'uso	13

3.5 Attori	14
3.5.1 Primari	14
3.5.2 Secondari	15
3.6 Definizione dei casi d'uso Utente target	15
3.6.1 UC01: Registrazione Utente	15
3.6.2 UC02: Visualizzazione UI	24
3.6.3 UC03: Visualizzazione pagina iniziale laboratorio	27
3.6.4 UC04: Visualizzazione step progressivi	30
3.6.5 UC05: Visualizzazione pagina di chiusura laboratorio .	43
3.7 Definizione dei casi d'uso Utente Admin	45
3.7.1 UC06: Login Admin	45
3.7.2 UC07: Visualizzazione homepage Admin	48
Implementazione	63
Retrospettiva finale	65
Glossario	67
Bibliografia	69

Elenco delle Figure

Figura 1.1 vista della Kanban alla seconda settimana	3
Figura 1.2 vista diagramma di Gantt alla seconda settimana	4

Figura 3.1	Interfaccia di registrazione utente	15
Figura 3.2	UC01: diagramma UML	16
Figura 3.3	Errore nome utente non rispettoso	18
Figura 3.4	UC08: Salvataggio user	18
Figura 3.5	Errore nome utente già utilizzato	20
Figura 3.6	UC08: Salvataggio user	20
Figura 3.7	Errore campo mancante	22
Figura 3.8	UC08: Salvataggio user	22
Figura 3.9	Homepage di Thinky	24
Figura 3.10	UC02: Visualizzazione UI	24
Figura 3.11	Errore generico	26
Figura 3.12	Pagina iniziale del laboratorio	27
Figura 3.13	UC08: Salvataggio user	27
Figura 3.14	UC08: Salvataggio user	29
Figura 3.15	UC08: Salvataggio user	29
Figura 3.16	UC08: Salvataggio user	30
Figura 3.17	UC08: Salvataggio user	32
Figura 3.18	UC08: Salvataggio user	33
Figura 3.19	Step 2 del laboratorio	34
Figura 3.20	UC08: Salvataggio user	34
Figura 3.21	Step 3 del laboratorio	36
Figura 3.22	UC08: Salvataggio user	36
Figura 3.23	Step 4 del laboratorio	37
Figura 3.24	UC08: Salvataggio user	38
Figura 3.25	Step 5 del laboratorio	39
Figura 3.26	UC08: Salvataggio user	39
Figura 3.27	Step 6 del laboratorio	41
Figura 3.28	UC08: Salvataggio user	41
Figura 3.29	Pagina di chiusura del laboratorio	43
Figura 3.30	UC05: Visualizzazione pagina di chiusura laboratorio .	43

Figura 3.31 UC08: Salvataggio user	44
Figura 3.32 Pagina di accesso Admin	45
Figura 3.33 Schermata di accesso Admin	46
Figura 3.34 Errore di accesso	47
Figura 3.35 Visualizzazione homepage Admin	48
Figura 3.36 Visualizzazione tabella lista utenti registrati	49
Figura 3.37 Errore lista utenti vuota	50
Figura 3.38 visualizzazione tab gestione laboratorio	52
Figura 3.39 Pulsanti per la gestione degli step del laboratorio	53
Figura 3.40 UC08: Salvataggio user	58
Figura 3.41 Errore durante il caricamento dei dati	59
Figura 3.42 UC08: Salvataggio user	61

Elenco delle Tabelle

Capitolo 1

Introduzione

1.1 Motivazioni e Contesto

Il corso di laurea triennale in *Informatica* che offre l'Ateneo dell'*Università Di Padova* è a numero chiuso, con un massimo di 220 posti, cifra che nel corso degli anni è aumentata visto il grande interesse da parte di studenti delle superiori di intraprendere questo percorso. Tuttavia, le scuole di provenienza delle matricole mostrano una grande disparità, con il numero degli studenti provenienti da istituti ad indirizzo informatico (come istituti tenici o liceo delle scienze applicate) che supera di gran lunga il numero di studenti provenienti da licei tradizionali. In particolare, parliamo di circa il 75% di studenti che proviene da istituti tecnici, con il restante 25% proveniente da altre scuole.

Questa disparità è data sicuramente dalla «credenza» che l'informatica sia solo programmazione e che quindi uno studente proveniente da un Liceo Scientifico non abbia le competenze necessarie per questo affrontare questo percorso.

Tuttavia, di contro a questa «credenza», è ovvio che non mancano le competenze, ma solamente le conoscenze, che è proprio il vuoto che questo corso va a colmare. Se magari uno studente proveniente da un istituto tecnico ha già delle competenze di programmazione, uno studente proveniente da un liceo scientifico ha sicuramente delle com-

petenze matematiche e logiche che sono fondamentali per affrontare questo percorso.

Altro obiettivo molto importante, è quello di aumentare il numero di iscritti di genere femminile, dimostrando, grazie anche a vari testimonial ([WomenInCS](#)) alle varie studentesse, che la storia dell'informatica è stata fatta in non piccola parte da donne e scienziate.

L'informatica non è solo programmazione, ma è anche progettazione e design. Infatti, il mio progetto si basa sulla creazione di un'applicazione web interattiva, che richiede competenze di progettazione e design oltre a quelle di programmazione.

Personalmente, essendo uno di quei ragazzi timorosi di iniziare il percorso, arrivando da un Liceo Scientifico, mi sono sentito moralmente in dovere di aiutare tutti coloro che si trovano nella posizione in cui io stesso mi sono trovato, anche per questo quindi ho deciso di fare questo tirocinio con la Prof.ssa Ombretta Gaggi.

1.2 Strumenti e processi

Durante il corso del tirocinio, mi sono avvalso di diversi strumenti che ho imparato ad utilizzare nel corso della mia carriera universitaria, e che mi hanno aiutato a portare avanti il mio progetto, come [Git](#) G e [GitHub](#) G, utilissimi per tenere traccia di ogni modifica effettuata al codice sorgente dell'applicazione, nonché per la condivisione di tale codice con la mia relatrice. In ogni caso, tutte le tecnologie saranno discusse nel dettaglio [secondo capitolo](#).

1.2.1 Suddivisione del lavoro

Dovendo presentare un [Piano di Lavoro](#) G per iniziare il mio tirocino, e volendo rispettare gli insegnamenti appresi dal corso di Ingegneria del Software, ho da subito deciso di impostare il mio Way of Working.

Ho quindi dapprima definito tutti gli obiettivi da raggiungere durante il percorso, trovandone 19. A questo punto, ho suddiviso il lavoro da svolgere nelle 8 settimane, potendo quindi definire degli « [sprint](#) ».
A fine di ogni sprint, controllo di aver completato tutto ciò che mi ero prefissato nel [backlog](#) di lavoro e, se ci fosse qualcosa che non ho completato, lo sposto nel backlog del prossimo sprint.

Per il tracciamento delle task da completare, ho utilizzato una [Kanban](#) board^[1.1], divisa in 3 colonne:

1. ToDo: attività da completare
2. In Progress: attività in corso
3. Done: attività completate

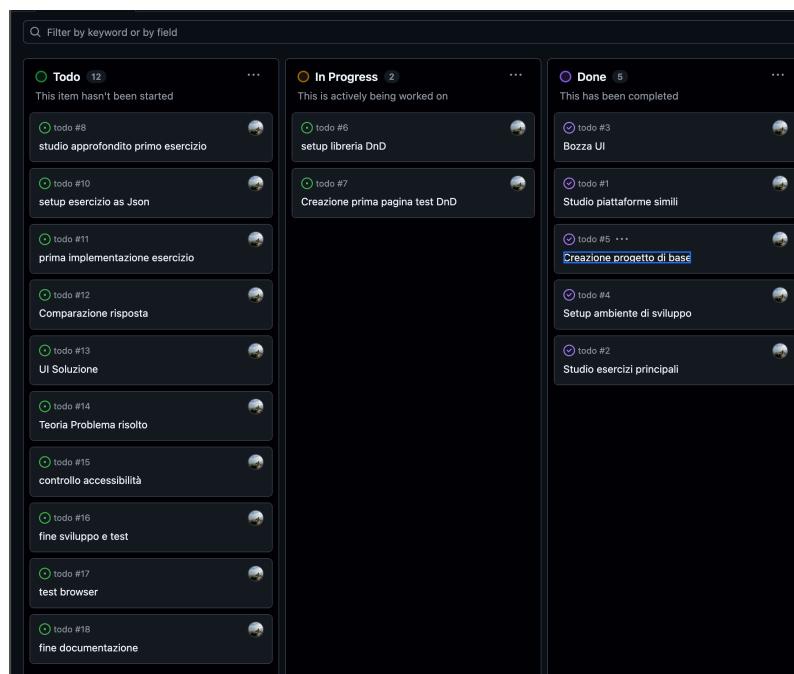


Figura 1.1: vista della Kanban alla seconda settimana

Invece, per visualizzare i tempi di svolgimento previsti ed effettivi di tali task, ho deciso di utilizzare un diagramma di Gantt^[1.2].

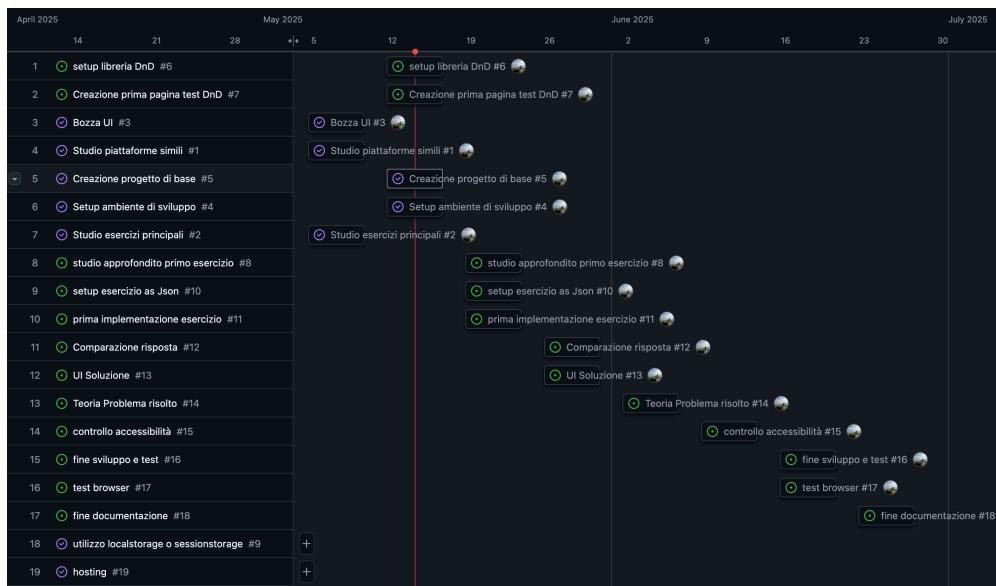


Figura 1.2: vista diagramma di Gantt alla seconda settimana

Tutto questo è conforme al metodo di lavoro [Scrum](#), che prevede una suddivisione del lavoro in sprint e un monitoraggio costante dei progressi. Lo Scrum fa parte della metodologia Agile^[1], creata per migliorare lo sviluppo di prodotti software rallentati dalle tediote fasi di analisi e documentazione.

Dato che il progetto è stato svolto in singolo, mi sono impegnato a mantenere un ritmo di lavoro costante, e di fornire aggiornamenti settimanali alla #myProf, via email o incontri di persona, per discutere i progressi, le difficoltà incontrate e risolvere eventuali problemi prima di andare eccessivamente fuori strada.

1.3 Struttura del Documento

- **Cosa:** Il secondo capitolo fornisce una panoramica di tutto il progetto, dal suo scopo, a i prodotti attesi, descrivendo nel dettaglio il concetto di webapp interattiva e le tecnologie utilizzate e le motivazioni dietro queste.
- **Analisi dei Requisiti:** Il terzo capitolo fornisce una panoramica dei requisiti del progetto, partendo dai requisiti funzionali e non funzio-

nali, passando per i vincoli e le limitazioni, fino ad arrivare ai casi d'uso e alle user stories.

- **Come:** Il [quarto capitolo](#) espone come le scelte descritte nel capitolo precedente sono state implementate. Inoltre, viene fornita una panoramica del codice sorgente, con i file più significativi e le loro funzionalità, oltre che la descrizione delle caratteristiche di accessibilità.
- **Conclusioni:** Il [quarto capitolo](#) fornisce una panoramica dei risultati ottenuti, sia a livello di codice sorgente che di accessibilità. Inoltre, viene fornita una panoramica dei test effettuati e dei risultati ottenuti.
- Nel [Glossario](#) sono riportati i termini tecnici e le abbreviazioni utilizzate nel corso del documento.

Oltre alla struttura qui sopra descritta, si adottano anche i seguenti accorgimenti tipografici:

- le abbreviazioni, termini tecnici (o comunque di uso non comune), o in lingua straniera in prima occorrenza nel documento sono definiti nel glossario consultabile alla fine del documento. Ogni termine nel glossario è evidenziato come segue: [Parola_G](#).
- Altri termini che richiedono un'attenzione particolare, ma che non hanno bisogno di essere definiti, saranno evidenziati in corsivo: *Parola*.

Capitolo 2

Scopo del tirocinio

Il prossimo capitolo fornisce una panoramica dettagliata del progetto di stage, partendo dal suo scopo fino ai prodotti attesi.

2.1 Scopo del progetto

Come evidenziato nel capitolo precedente, questo progetto di stage è volto a colmare una grande “disparità” tra gli studenti che si iscrivono al corso di laurea in *Informatica dell’Università di Padova*.

Possiamo comunque distinguere due “sotto-scopi” che convogliano poi in nel progetto finale:

Il primo scopo, definibile *scopo tecnico*, e rivolto al laureando, è quello di portare a limite le proprie abilità, competenze e conoscenze cercando di creare un prodotto che non solo sia funzionale, ma anche facile da utilizzare, accessibile a tutti e che aiuti effettivamente gli studenti. Questo scopo si traduce, all’atto pratico, nello sviluppo della WebApp interattiva.

Il secondo scopo, definibile *scopo sociale*, è rivolto agli studenti delle scuole superiori, e ha come obiettivo quello di mostrare loro che l’informatica non è solo programmazione, ma anche progettazione e design, e che non importa quale sia il loro *background* scolastico per affrontare questo percorso. In particolare, si vuole dimostrare che ogni scuola superiore, anche se in maniera e con contenuti diversi, permette di affrontare questo percorso, chi in un modo e chi in un altro.

Se dovessi dare quindi una definizione sintetica, quasi matematica come il mio corso ha insegnato, dello scopo del progetto, direi che è:

Si definisce il seguente progetto di sviluppo di una WebApp interattiva per l'orientamento agli Open Day di Informatica il processo che, dato un insieme eterogeneo di studenti delle scuole superiori, realizza uno strumento digitale accessibile e inclusivo, volto a ridurre le disparità di background e di genere e a mostrare che l'informatica è disciplina aperta a tutti, indipendentemente dal percorso scolastico precedente.

2.2 Obiettivi prefissati

Nelle fasi iniziali di quella che possiamo chiamare "candidatura" al progetto, ho definito ciò che ritenevo fossero gli obiettivi principali del progetto, sapendo che questi comunque sarebbero stati mutevoli nel tempo, essendo aggiornati man mano che il progetto andava avanti sulla base delle esigenze e sull'effettivo progresso del progetto. Gli obiettivi definiti sono poi stati discussi insieme alla mia relatrice, e raggruppati in tre diverse categorie:

- **Obiettivi Obbligatori:** sono gli obiettivi che devono essere raggiunti per considerare il progetto completato e soddisfacente. Questi obiettivi sono stati definiti in modo da garantire che il prodotto finale sia funzionale, accessibile e utile per gli studenti delle scuole superiori.
- **Obiettivi Desiderabili:** sono obiettivi che, se raggiunti, migliorano significativamente il progetto e lo rendono più completo e interessante. Questi obiettivi possono essere considerati come "aggiunte" che arricchiscono l'esperienza utente o aggiungono funzionalità utili, ma non sono strettamente necessari per il completamento del progetto.

- **Obiettivi Facoltativi:** sono obiettivi che, se raggiunti, arricchiscono il progetto e lo rendono più completo e interessante. Questi obiettivi possono essere considerati come “extra” che migliorano l’esperienza utente o aggiungono funzionalità utili.

Per la classificazione degli obiettivi è stata adottata la seguente nomenclatura:

- Obbligatorio: O
- Desiderabile: D
- Facoltativo: F

Con il codice identificativo dell’obiettivo che diventa quindi:

[X] [Y]

con X che rappresenta la categoria dell’obiettivo e Y che rappresenta il numero progressivo dell’obiettivo.

Gli obiettivi definiti inizialmente sono stati i seguenti:

Obbligatori

- O01: implementazione dell’applicativo;
- O02: funzionamento base dello stesso;
- O03: primo problema: produttore - consumatore;
- O04: simulazione della sequenza di operazione eseguita come live demo;
- O05: misurazione delle performance del punto O04;

Desiderabili

- D01: implementare il problema dei filosofi a cena;
- D02: possibilità di inserire una sequenza di operazioni;

Facoltativi

- F01: inserimento di uno username per mostrarlo nella scoreboard (no login);

- F02: inserimento di un punteggio in base alla percentuale di performance;
- F03: visualizzazione di una scoreboard di tutti gli utenti;

Come anticipato, questi obiettivi sono stati in parte rivisti in corso d'opera insieme alla Prof.ssa Ombretta Gaggi, e sono stati aggiunti altri obiettivi, che sono stati classificati come segue:

Obbligatori

- O01: implementazione dell'applicativo;
- O02: funzionamento base dello stesso;
- O03: primo problema: lettore - scrittore;
- O04: Inserimento di username e scuola di provenienza per tenere traccia dell'utilizzo dell'applicativo;

Facoltativi

- F01: implementare il problema del produttore - consumatore;
- F02: possibilità di inserire una sequenza di operazioni;
- F03: Visualizzazione delle statistiche di risposte ai quesiti posti agli utenti;

Desiderabili

- D01: Creazione di un'area riservata per l'amministratore senza necessità di login particolari;
- D02: visualizzazione di una dashboard con tutti gli utenti;

2.3 Prodotti attesi

Al completamento del progetto, è prevista la consegna di un prodotto funzionante che rispetti i requisiti definiti in fase di progettazione.

Sarà consegnato il codice sorgente del progetto, che sarà ben documentato e commentato, in modo da permettere una facile comprensione e manutenzione del codice stesso. Inoltre, sarà fornita una documentazione tecnica che descrive le funzionalità implementate, le

tecnologie utilizzate e le modalità di utilizzo del prodotto, oltre che un breve documento che spiega all'utente come utilizzare l'applicativo, e un documento che spiega ad eventuali futuri programmatori come è strutturato il codice, per guiderli in un eventuale manutenzione o potenziamento del progetto.

Oltre che alla documentazione tecnica e al codice sorgente, sarà prodotto questo documenti di tesi, che descrive appunto il progetto sotto ogni punto di vista e ne analizza i risultati ottenuti, le difficoltà incontrate e le soluzioni adottate.

Capitolo 3

Analisi dei Requisiti

Il prossimo capitolo fornisce una panoramica dettagliata del progetto di stage, partendo dal suo scopo, passando per gli obiettivi prefissati e le tecnologie utilizzate, arrivando ai prodotti attesi.

L'analisi dei requisiti è una fase, e di conseguenza, un documento (o in questo caso, capitolo), fondamentale nel ciclo di vita di un progetto.

Il suo scopo è quello di definire in modo chiaro, preciso e dettagliato le funzionalità che il prodotto finale andrà ad offrire, ossia i requisiti obbligatori e opzionali richiesti dal [proponente](#).

Nello specifico, questo capitolo si propone di:

- Fornire un'analisi basata direttamente sulle richieste del proponente, in particolare, si basa sugli obiettivi riscontrati negli incontro con la Prof.ssa Ombretta Gaggi, che sono stati riportati nel capitolo precedente.
- Identificare i requisiti e suddividerli in funzionali e non funzionali.
- Validare e verificare i requisiti rispettando la Way of Working adottata.

3.1 Tecnologie utilizzate

Le tecnologie utilizzate nel progetto e la loro selezione sono state fatte in comune accordo con la Prof.ssa Ombretta Gaggi, tenendo conto delle competenze pregresse, acquisite durante il corso di *Ingegneria del Software* e delle tecnologie più moderne e adatte al contesto del

progetto. Tali scelte sono state fatte con l'obiettivo di assicurare un prodotto finale di alta qualità, facilmente manutenibile e modificabile in futuro, e che potesse essere utilizzato da altri studenti come me, garantendo anche una rapida curva di apprendimento.

3.1.1 Typescript

3.1.2 Axios

3.1.3 Firebase

3.1.4 GitHub

3.1.5 ShadCN

3.1.6 NextJS

Nonstante fin da subito fosse stato chiaro che non sarebbe stato necessario un backend, ho comunque optato per NextJS, perché sapevo che avrei dovuto implementare funzionalità che richiedevano di effettuare chiamate alle [API G](#) di [GitHub G](#). Inoltre, grazie al rendering lato server (SSR) e alla generazione statica delle pagine (SSG) con i [server-side components G](#), Next permette di limitare il caricamento di codice JavaScript necessario per il rendering della pagina verso il client, rendendo così il sito più leggero sul browser, più veloce e piacevole da utilizzare, e soprattutto più [SEO G](#) friendly, quindi, in un'ottica aziendale, più *appeasing* a livello marketing. COMMENTO(non so se aggiungere una parte «teorica» su NextJS)

3.1.7 TailwindCSS

Ho scelto di utilizzare TailwindCSS per la sua flessibilità e per la sua capacità di creare interfacce utente reattive e personalizzabili in modo semplice e veloce. Inoltre, grazie alla sua natura *utility-first*, permette di scrivere meno codice CSS, rendendo il progetto più leggero e veloce da caricare, senza preoccupazioni di conflitti tra stili dati dalla rigidità della gerarchia di specificità di CSS.

3.2 Obiettivo del progetto

Thinky è una WebApp progettata per aiutare gli studenti che sono intenzionati ad iscriversi al corso di laurea in *Informatica dell'Università di Padova* a approcciarsi al mondo della programmazione durante gli Open Day, mediante un laboratorio interattivo svolto insieme alla Prof.ssa Ombretta Gaggi e al Prof. Claudio Palazzi.

L'app utilizza diversi strumenti per fornire all'utente un'esperienza interattiva e coinvolgente, senza esagerare nella difficoltà, sia di utilizzo che dell'attività da svolgere.

3.3 Utenti Target

Gli utenti target di Thinky sono principalmente studenti delle scuole superiori che sono interessati a iscriversi al corso di laurea in *Informatica dell'Università di Padova*. In particolare, si rivolge a:

- Ragazzi e Ragazze di età compresa tra i 17 e i 19 anni, che stanno per diplomarsi e sono interessati a intraprendere un percorso di studi in informatica;
- Studenti provenienti da istituti tecnici e professionali, che potrebbero avere una formazione di base in informatica;
- Studenti provenienti da licei scientifici e classici, che potrebbero avere una formazione più analitico-matematica ma sono interessati a esplorare il mondo della programmazione e dello sviluppo software;

3.4 Casi d'uso

Nella seguente sezione verranno elencati i casi d'uso principali della WebApp, che sono stati identificati in base agli obiettivi del progetto e alle esigenze degli utenti target.

Tutti i casi d'uso adottano la medesima struttura, come segue:

- **Nome Caso D'uso:** Nome del caso d'uso chiaro e descrittivo;
- **Attori:** Gli attori coinvolti nel caso d'uso, che possono essere utenti o sistemi esterni;
- **Precondizione e postcondizione:** Stato del sistema prima e dopo l'esecuzione del caso d'uso;
- **Scenario principale:** descrizione dettagliata dei passi che l'utente deve seguire per completare il caso d'uso;
- **Estensioni:** Eventuali estensioni o variazioni dello scenario principale, che possono includere errori o situazioni particolari che l'utente potrebbe incontrare durante l'esecuzione del caso d'uso;
- **Inclusioni:** Eventuali casi d'uso inclusi che sono necessari per completare il caso d'uso principale;
- **Generalizzazioni:** Eventuali generalizzazioni del caso d'uso, che possono includere casi d'uso più specifici o varianti del caso d'uso principale.
- **User Story:** Una breve descrizione del caso d'uso in forma di user story, che può essere utilizzata per comunicare il caso d'uso in modo più semplice e comprensibile.

È possibile suddividere i casi d'uso del sistema Thinky in due categorie principali:

- I casi d'uso relativi all'utente target, che riguardano le funzionalità principali della WebApp e l'interazione con l'utente;
- I casi d'uso relativi all'utente Admin;

3.5 Attori

3.5.1 Primari

Per attore primario si intende un attore che interagisce direttamente con il sistema per raggiungere un obiettivo specifico. Nel caso di Thinky, l'attore primario è l'utente, ossia lo studente che partecipa al laboratorio interattivo. L'utente ha accesso a tutte le funzionalità della

WebApp e può interagire con il sistema per completare le attività previste dal laboratorio.

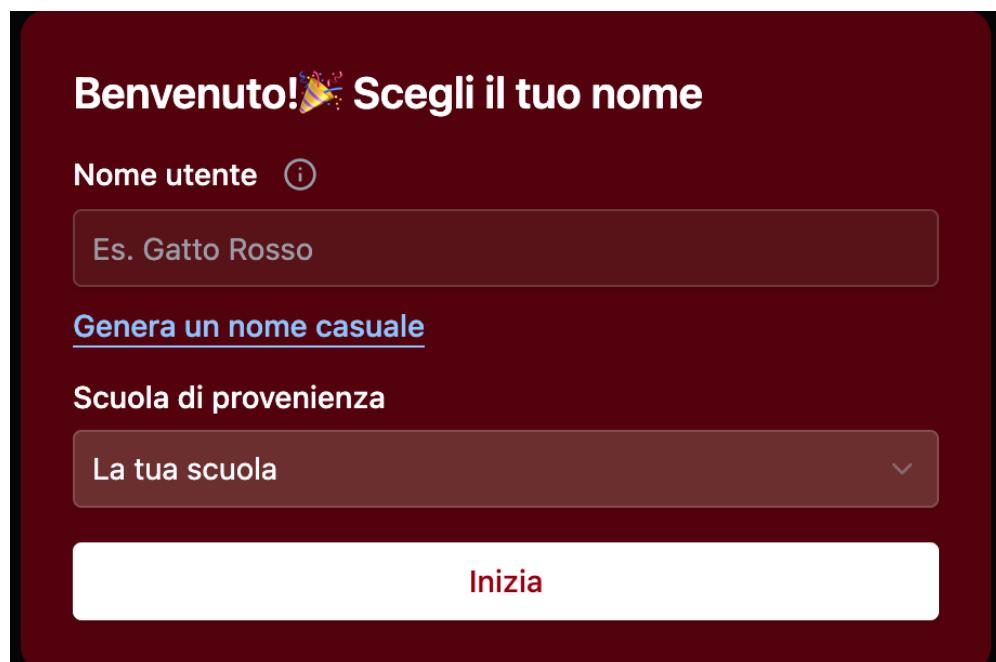
3.5.2 Secondari

Per attore secondario si intende tutti quei servizi o sistemi esterni che Thinky utilizza a supporto delle sue funzionalità. Sono tutti attori su cui non si ha effettuato alcun tipo di sviluppo e che vengono «contattati» dal sistema, invece di contattare il sistema stesso. Nel caso di Thinky, gli attori secondari sono:

- [Firebase](#), servizio di backend as service che fornisce un database in tempo reale;
- [GitHub](#), piattaforma di hosting per progetti software che utilizza Git come sistema di controllo versione;

3.6 Definizione dei casi d'uso Utente target

3.6.1 UC01: Registrazione Utente



The screenshot shows a dark-themed user interface for registration. At the top, it says "Benvenuto! Scegli il tuo nome". Below that is a field labeled "Nome utente" with a placeholder "Es. Gatto Rosso". There is also a link "Genera un nome casuale". Below the name field is a dropdown menu labeled "Scuola di provenienza" with the option "La tua scuola" selected. At the bottom is a large red button labeled "Inizia".

Figura 3.1: Interfaccia di registrazione utente

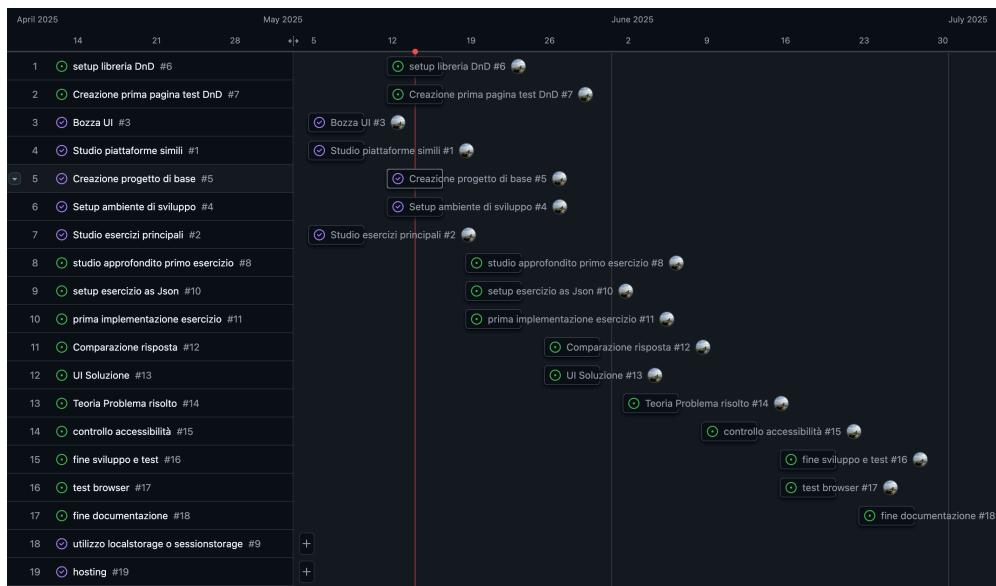


Figura 3.2: UC01: diagramma UML

Attori coinvolti

- **Attori Primari:** Utente
- **Attori Secondari:** GitHub

Precondizioni e Postcondizioni

- **Precondizioni:**
 - Il sistema è connesso e funzionante.
 - L'utente non è registrato nel sistema.
 - Le [API G](#) di GitHub sono disponibili e funzionanti.
- **Postcondizioni:**
 - L'utente è registrato e può accedere alle funzionalità della WebApp.

Scenario principale

- L'utente apre per la prima volta l'interfaccia di Thinky.
- Il sistema rileva che l'utente non è registrato e mostra il form di registrazione.
- L'utente inserisce un nome utente e seleziona la scuola di provenienza.

- Il sistema verifica la validità dei dati inseriti (vedi estensioni UC1.1 - UC1.3).
- Se i dati sono validi, l'utente viene registrato e il nome viene salvato su GitHub.
- L'utente viene reindirizzato alla homepage della WebApp.

Estensioni

- UC1.1: Visualizzazione messaggio di errore se il nome utente non è rispettoso.
- UC1.2: Visualizzazione messaggio di errore se il nome utente è già stato utilizzato.
- UC1.3: Visualizzazione messaggio di errore se non sono stati compilati tutti i campi.

User story

- Come utente, voglio potermi registrare al sistema per accedere alle funzionalità, senza fornire dati sensibili.

UC1.1: Visualizzazione messaggio di errore se il nome utente non è rispettoso

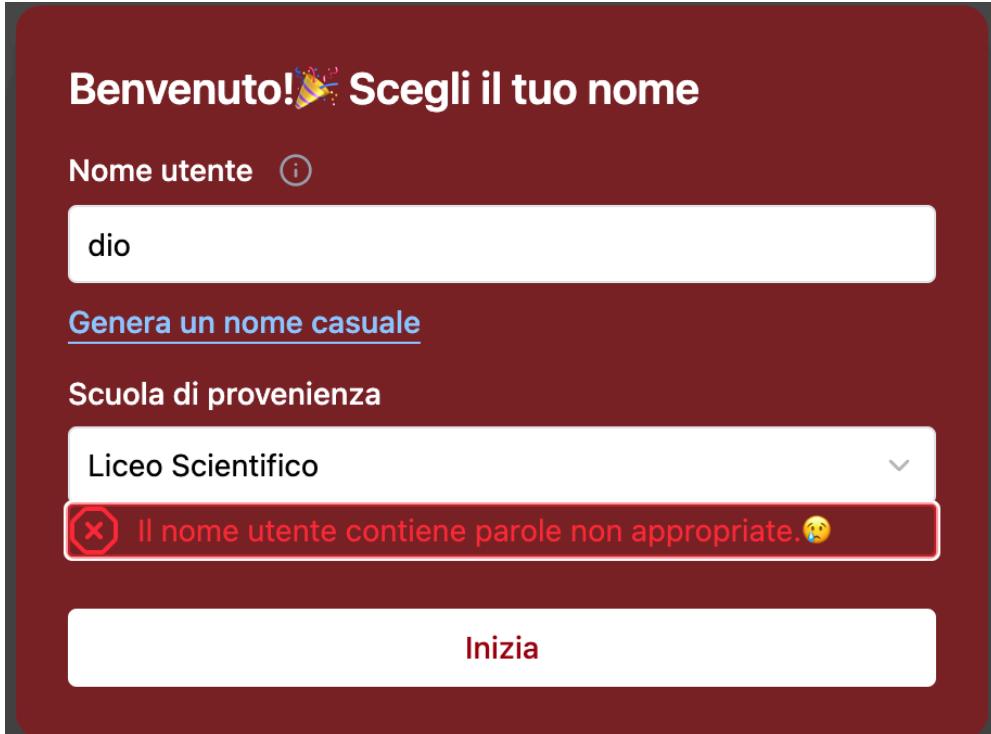


Figura 3.3: Errore nome utente non rispettoso

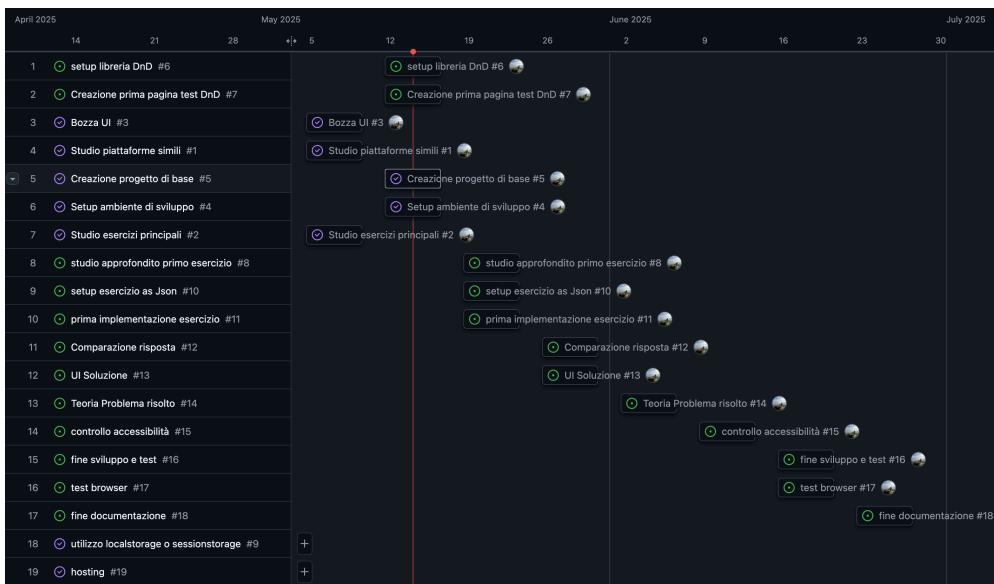


Figura 3.4: UC08: Salvataggio user

Attori coinvolti

- **Attori Primari:** Utente
- **Attori Secondari:** GitHub

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'utente ha inserito un nome utente non rispettoso.
- Le API di GitHub sono disponibili e funzionanti..

- **Postcondizioni:**

- Il sistema visualizza un messaggio di errore.

Scenario principale

- L'utente inserisce un nome utente non rispettoso.
- Il sistema visualizza un messaggio di errore.
- L'utente corregge il nome utente e ripete la registrazione.

User story

- Quando inserisco un nome utente non rispettoso, il sistema mostra un messaggio di errore così posso correggerlo e ripetere la registrazione.

UC1.2: Visualizzazione messaggio di errore se il nome utente è già utilizzato



Figura 3.5: Errore nome utente già utilizzato

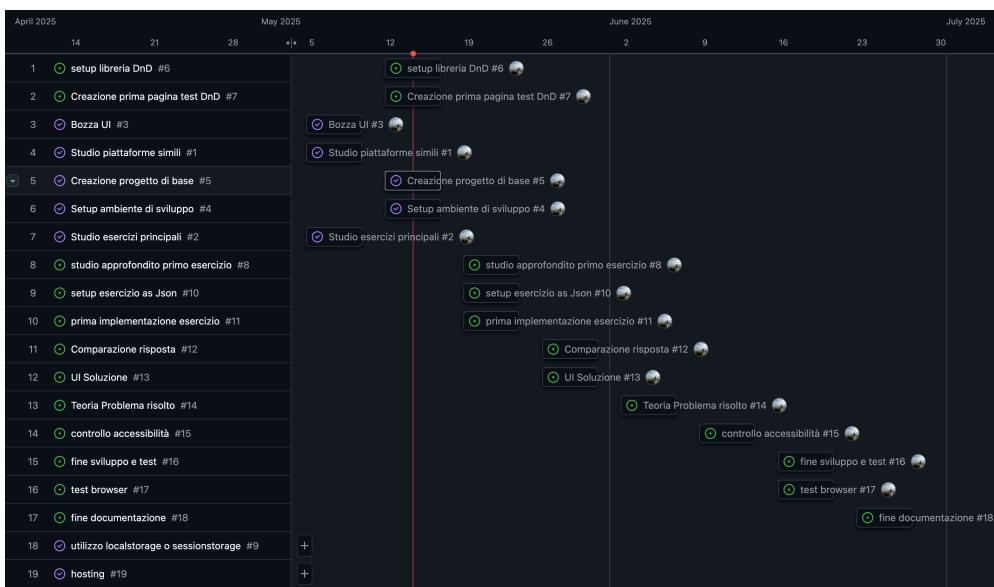


Figura 3.6: UC08: Salvataggio user

Attori coinvolti

- **Attori Primari:** Utente
- **Attori Secondari:** GitHub

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- Il nome utente inserito è già utilizzato da un altro utente.
- Le API di GitHub sono disponibili e funzionanti.

- **Postcondizioni:**

- Il sistema visualizza un messaggio di errore.

Scenario principale

- L'utente inserisce un nome utente già utilizzato.
- Il sistema visualizza un messaggio di errore.
- L'utente corregge il nome utente e ripete la registrazione.

User story

- Quando inserisco un nome utente già utilizzato, il sistema mostra un messaggio di errore così posso correggerlo e ripetere la registrazione.

UC1.3: Visualizzazione messaggio di errore se non sono stati compilati tutti i campi



Figura 3.7: Errore campo mancante

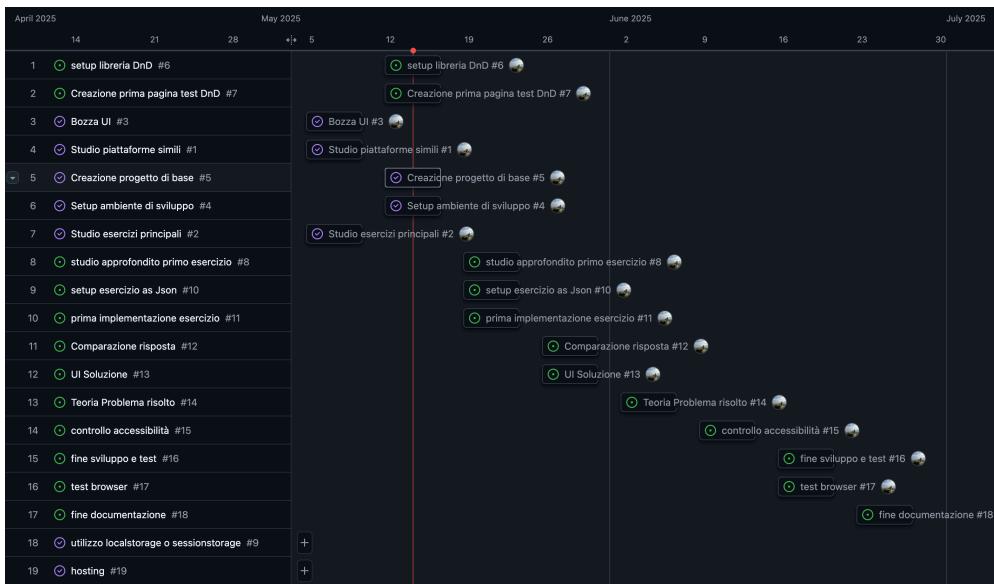


Figura 3.8: UC08: Salvataggio user

Attori coinvolti

- **Attori Primari:** Utente
- **Attori Secondari:** GitHub

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'utente ha inserito un nome utente ma non ha compilato il campo scuola di provenienza.
- Le API di GitHub sono disponibili e funzionanti.

- **Postcondizioni:**

- Il sistema visualizza un messaggio di errore.

Scenario principale

- L'utente inserisce un nome utente ma lascia vuoto il campo scuola di provenienza.
- Il sistema visualizza un messaggio di errore.
- L'utente completa tutti i campi e ripete la registrazione.

User story

- Quando non compilo tutti i campi richiesti, il sistema mi avvisa con un messaggio di errore così posso completare i dati mancanti.

3.6.2 UC02: Visualizzazione UI

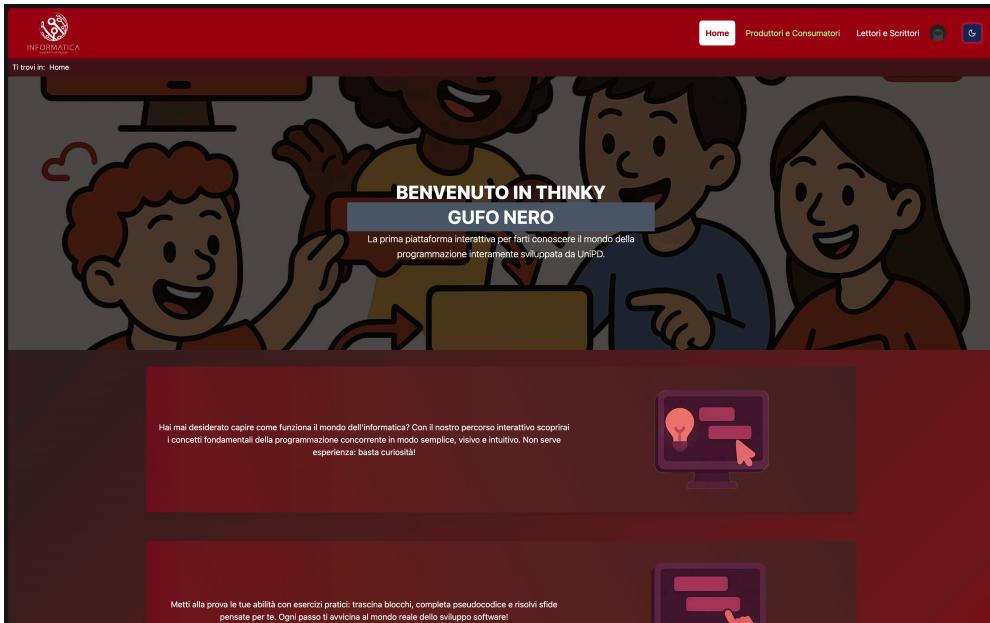


Figura 3.9: Homepage di Thinky

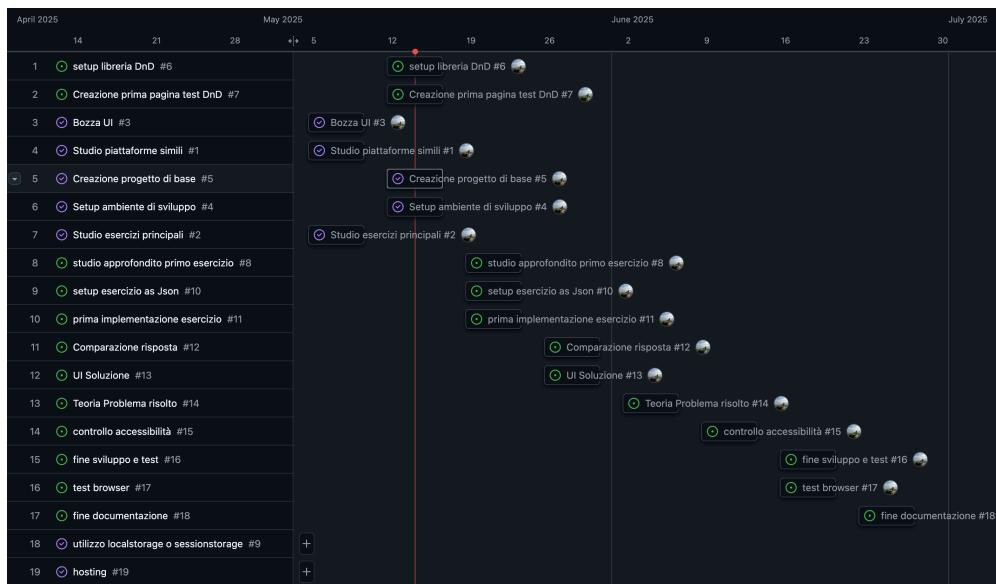


Figura 3.10: UC02: Visualizzazione UI

Attori coinvolti

- **Attori Primari:** Utente

Precondizioni e Postcondizioni

- **Precondizioni:**

- L'utente è registrato nel sistema.
- Il sistema è connesso e funzionante.

- **Postcondizioni:**

- L'utente visualizza l'interfaccia utente della WebApp.

Scenario principale

- L'utente accede alla WebApp.
- Il sistema visualizza l'interfaccia utente.

Estensioni

- UC2.1: Visualizzazione messaggio di errore generico in caso di mal-funzionamento del sistema.

Generalizzazioni

- UC2.2: Visualizzazione interfaccia in modalità scura.
- UC2.3: Visualizzazione interfaccia in modalità chiara.

User story

- Come utente, voglio visualizzare l'interfaccia della WebApp per interagire con essa e scegliere la modalità del tema.

UC2.1: Visualizzazione messaggio di errore generico



More info:
Error
Errore dimostrativo

Figura 3.11: Errore generico

Attori coinvolti

- **Attori Primari:** Utente

Precondizioni e Postcondizioni

- **Precondizioni:**
 - Il sistema è connesso e funzionante.
 - Si è verificato un malfunzionamento.
- **Postcondizioni:**
 - Il sistema visualizza un messaggio di errore generico.

Scenario principale

- L'utente accede alla WebApp.
- Si verifica un malfunzionamento.
- Il sistema mostra un messaggio di errore generico.

User story

- In caso di malfunzionamento, il sistema mostra un messaggio di errore generico così l'utente è informato e può riprovare.

3.6.3 UC03: Visualizzazione pagina iniziale laboratorio

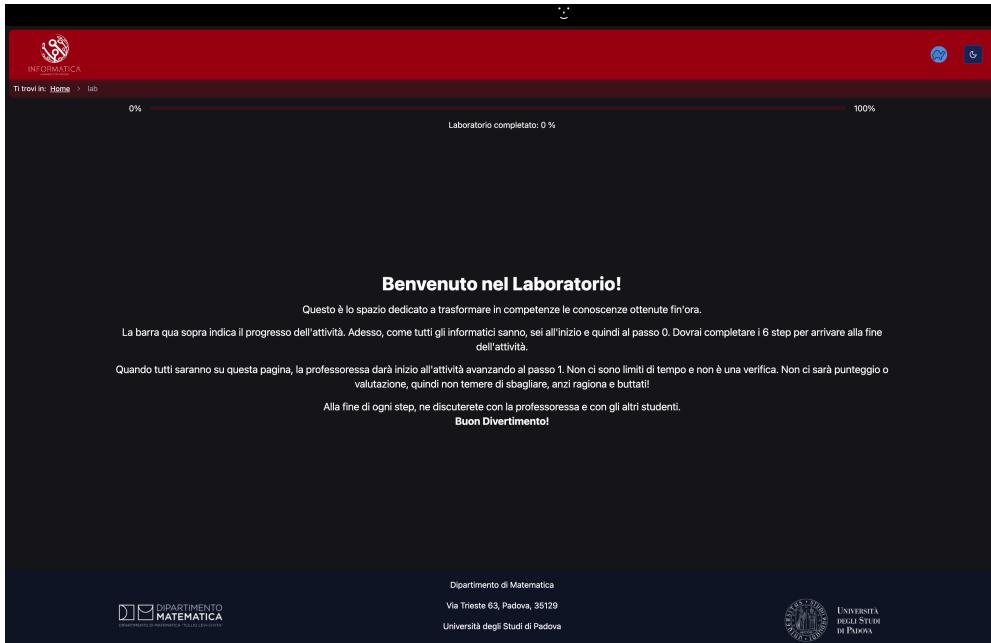


Figura 3.12: Pagina iniziale del laboratorio

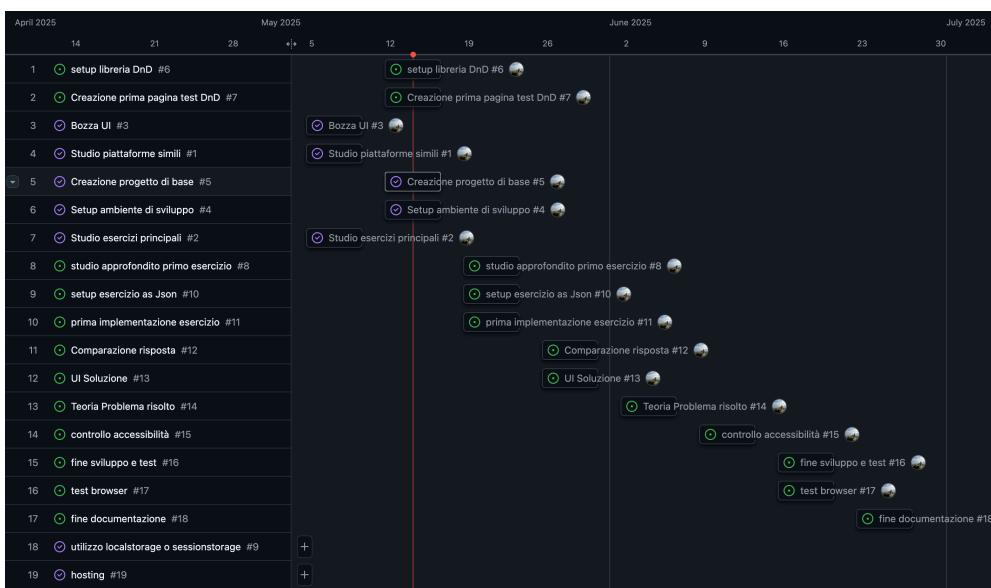


Figura 3.13: UC08: Salvataggio user

Attori coinvolti

- **Attori Primari:** Utente
- **Attori Secondari:** Firebase

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'utente è registrato.
- L'utente ha cliccato sul pulsante «Vai al laboratorio».

- **Postcondizioni:**

- L'utente visualizza la pagina iniziale del laboratorio.

Scenario principale

- L'utente clicca sul pulsante «Vai al laboratorio».
- Il sistema carica la pagina iniziale del laboratorio.

Estensioni

- UC3.1: Visualizzazione messaggio di errore se il caricamento di *currentStep* fallisce.

User story

- Come utente, voglio visualizzare la pagina iniziale del laboratorio per iniziare l'attività.

UC3.1: Visualizzazione messaggio di errore se il caricamento di currentStep non è andato a buon fine

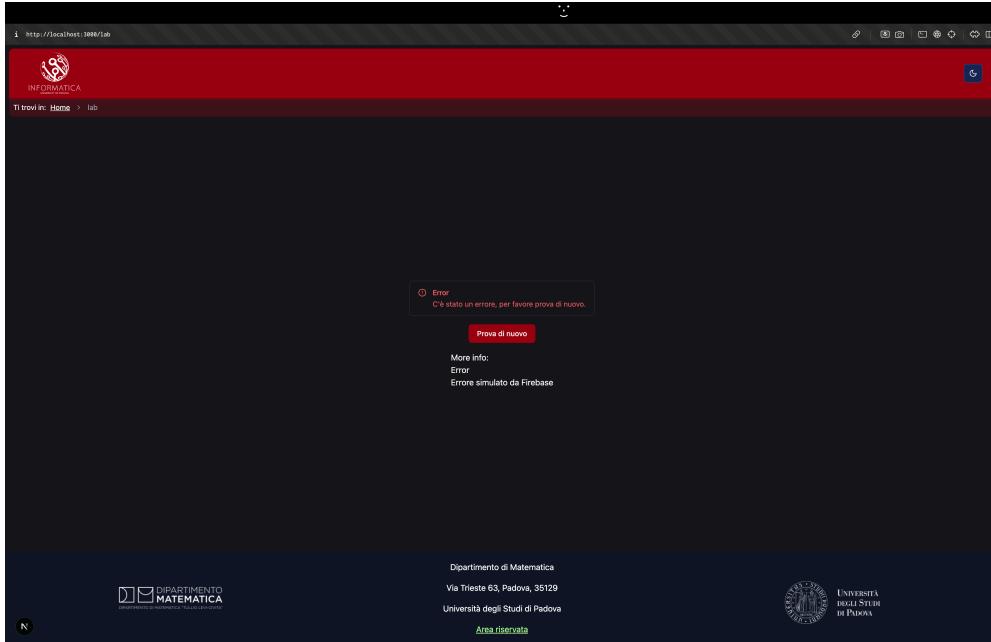


Figura 3.14: UC08: Salvataggio user

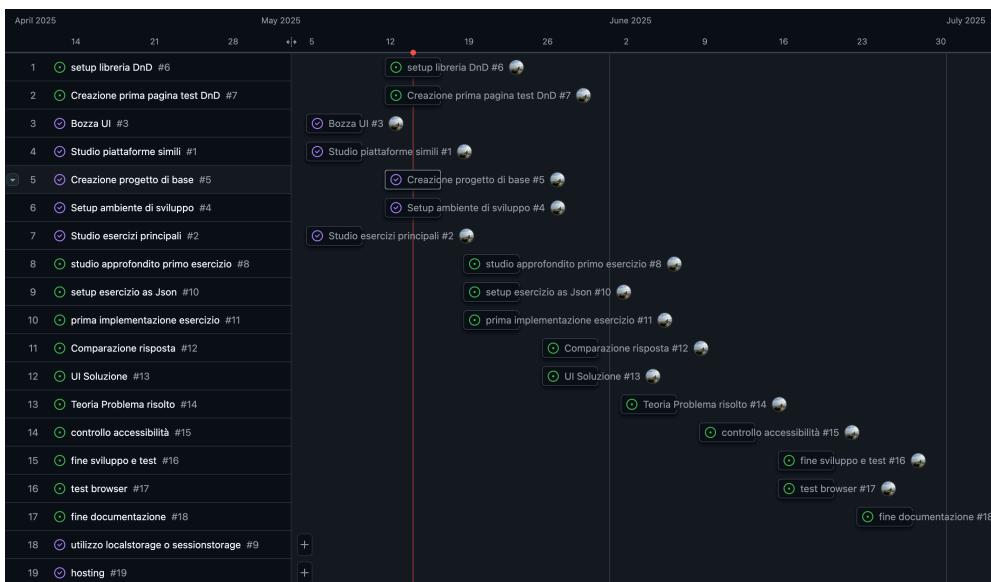


Figura 3.15: UC08: Salvataggio user

Attori coinvolti

- Attori Primari

- **Attori Secondari:** Firebase

Precondizioni e Postcondizioni

- **Precondizioni:**

- L'utente è registrato.
- Il sistema è connesso ma la richiesta di *currentStep* fallisce.

- **Postcondizioni:**

- Il sistema mostra un messaggio di errore.

Scenario principale

- L'interfaccia utente invia una richiesta API a Firebase per la variabile *currentStep*, ma la richiesta fallisce.

User story

- Se il caricamento di *currentStep* fallisce, il sistema mostra un messaggio di errore per informare l'utente e consentirgli di riprovare.

3.6.4 UC04: Visualizzazione step progressivi

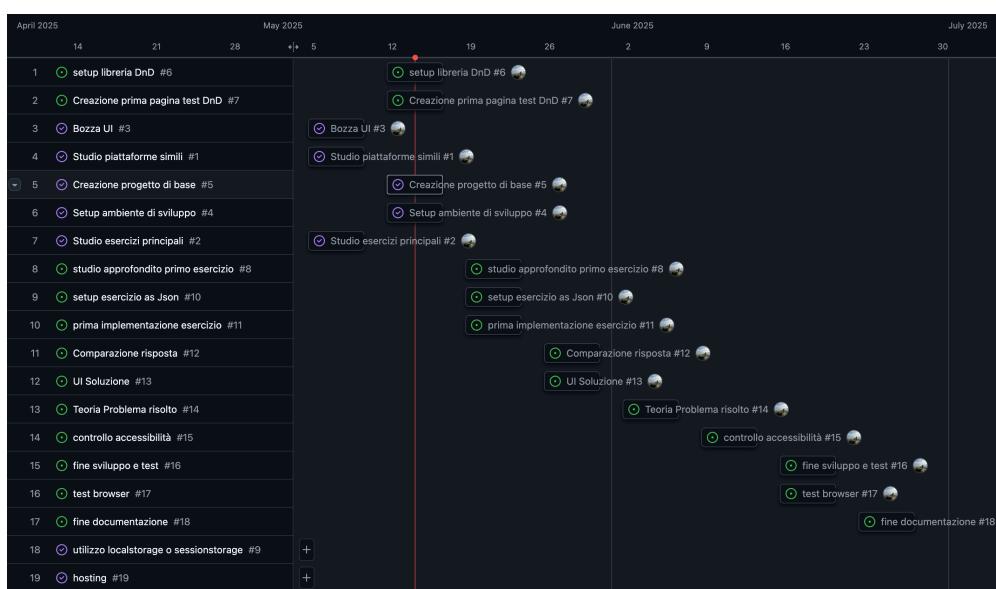


Figura 3.16: UC08: Salvataggio user

Attori coinvolti

- **Attori Primari:** Utente

- **Attori Secondari:** Firebase

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'utente è registrato.
- L'utente ha visualizzato la pagina iniziale del laboratorio.
- Il sistema ha caricato *currentStep* correttamente.

- **Postcondizioni:**

- L'utente visualizza lo step progressivo del laboratorio.

Scenario principale

- L'utente attende che l'admin carichi lo step successivo.
- Il sistema visualizza lo step progressivo del laboratorio.
- L'utente può interagire con esso.

Estensioni

- UC3.1: Visualizzazione messaggio di errore se il caricamento di *currentStep* fallisce.

Generalizzazioni

- UC4.1: Visualizzazione Step 1
- UC4.2: Visualizzazione Step 2
- UC4.3: Visualizzazione Step 3
- UC4.4: Visualizzazione Step 4
- UC4.5: Visualizzazione Step 5
- UC4.6: Visualizzazione Step 6

User story

- Come utente, voglio visualizzare lo step progressivo del laboratorio per interagire con esso e completare l'attività.

UC4.1: Visualizzazione Step 1

The screenshot shows a user interface for a programming exercise. At the top, there's a navigation bar with the logo of the University of Parma (INFOPARMA) and a progress bar indicating 'Laboratorio completato: 14 %'. The main area is titled 'Step 1: Completa il comportamento del Produttore' and instructs the user to 'Completa l'esercizio trascinando le risposte nella colonna a sinistra.' On the left, there are several code snippets in a dark grey box with placeholder text 'Completa qua' and 'Drop items here'. On the right, there's a list of responses in a 'Risposte' box, each with a small icon and a colored background: a red box for 'SenaforoRosso(Vuoto)' and three green boxes for 'SenaforoRosso(Scaffale)', 'SenaforoVerde(Pieno)', and 'SenaforoVerde(Scaffale)'.

Figura 3.17: UC08: Salvataggio user

Attori coinvolti

- **Attori Primari:** Utente
- **Attori Secondari:** Firebase

Precondizioni e Postcondizioni

- **Precondizioni:**
 - Il sistema è connesso e funzionante.
 - L'utente è registrato.
 - L'admin ha caricato lo step 1.
 - Il sistema ha caricato *currentStep* correttamente.
- **Postcondizioni:**
 - L'utente visualizza lo step 1.

Scenario principale

- L'utente attende che l'admin carichi lo step 1.
- Il sistema visualizza lo step 1.

- L'utente può interagire con esso.

Inclusioni

- UC4.2.1: Drag and Drop

User story

- Come utente, voglio visualizzare lo step 1 del laboratorio per interagire con esso e completare l'attività.

UC4.1.1: Drag and Drop

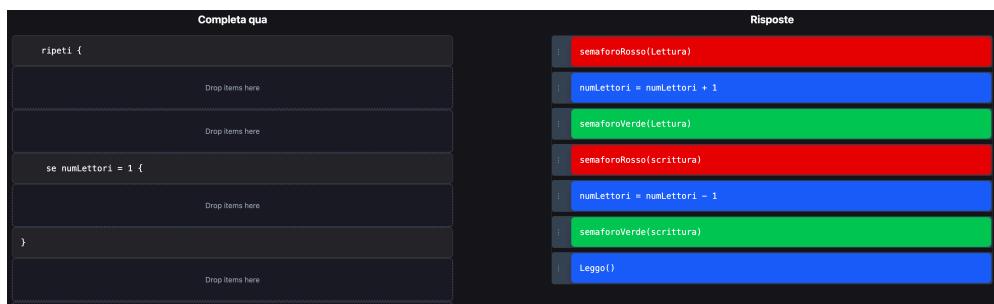


Figura 3.18: UC08: Salvataggio user

Attori coinvolti

- **Attori Primari:** Utente

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'utente è registrato.
- L'utente ha visualizzato lo step 1.

- **Postcondizioni:**

- L'utente può trascinare e rilasciare gli elementi dello step 1.

Scenario principale

- L'utente visualizza lo step 1.
- Il sistema permette il drag and drop degli elementi.
- L'utente interagisce con gli elementi trascinati e rilasciati.

User story

- Come utente, voglio poter trascinare e rilasciare gli elementi dello step 1 per completare l'attività.

UC4.2: Visualizzazione Step 2

The screenshot shows a user interface for a laboratory exercise. At the top, there's a navigation bar with a logo and the word 'INFORMATICA'. Below it, a progress bar indicates '0%' on the left and '100%' on the right, with a middle section labeled 'Laboratorio completato: 29 %'. The main area is titled 'Step 2: Completa il comportamento del consumatore' and contains the instruction 'Completa l'esercizio trascinando le risposte nella colonna a sinistra.' On the left, there are several code snippets in a dark box with placeholder text 'Drop items here'. On the right, there's a 'Risposte' (Responses) section with four items listed in a grid:

SenaforoRosso(Pieno)
SenaforoRosso(Scaffale)
SenaforoVerde(Vuoto)
SenaforoVerde(Scaffale)

Figura 3.19: Step 2 del laboratorio

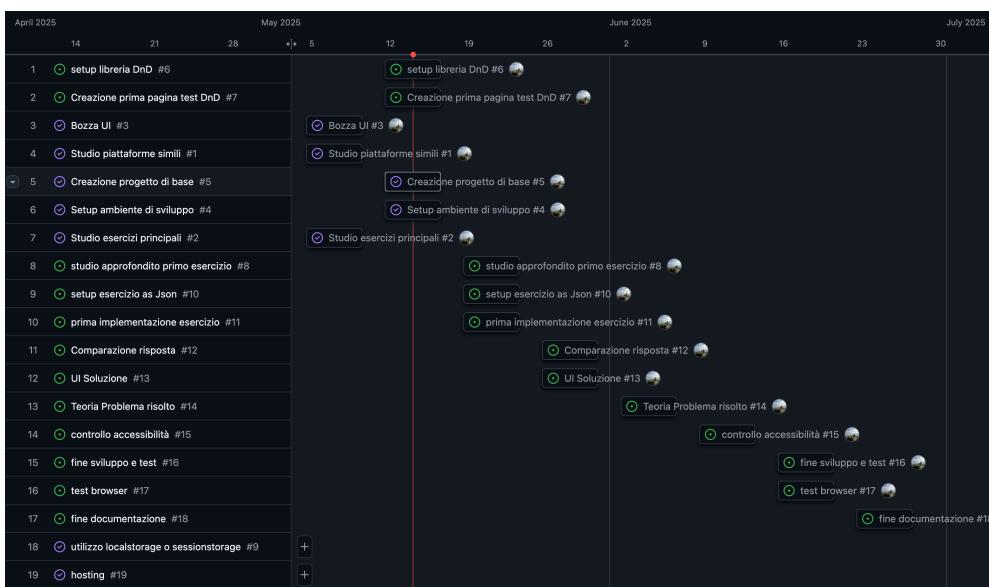


Figura 3.20: UC08: Salvataggio user

Attori coinvolti

- **Attori Primari:** Utente
- **Attori Secondari:** Firebase

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'utente è registrato.
- L'admin ha caricato lo step 2.
- Il sistema ha caricato *currentStep* correttamente.

- **Postcondizioni:**

- L'utente visualizza lo step 2.
- L'utente può interagire con esso.

Scenario principale

- L'utente attende che l'admin carichi lo step 2.
- Il sistema visualizza lo step 2.
- L'utente può interagire con esso.

Inclusioni

- UC4.1.1: Drag and Drop

User story

- Come utente, voglio visualizzare lo step 2 del laboratorio per interagire con esso e completare l'attività.

UC4.3: Visualizzazione Step 3

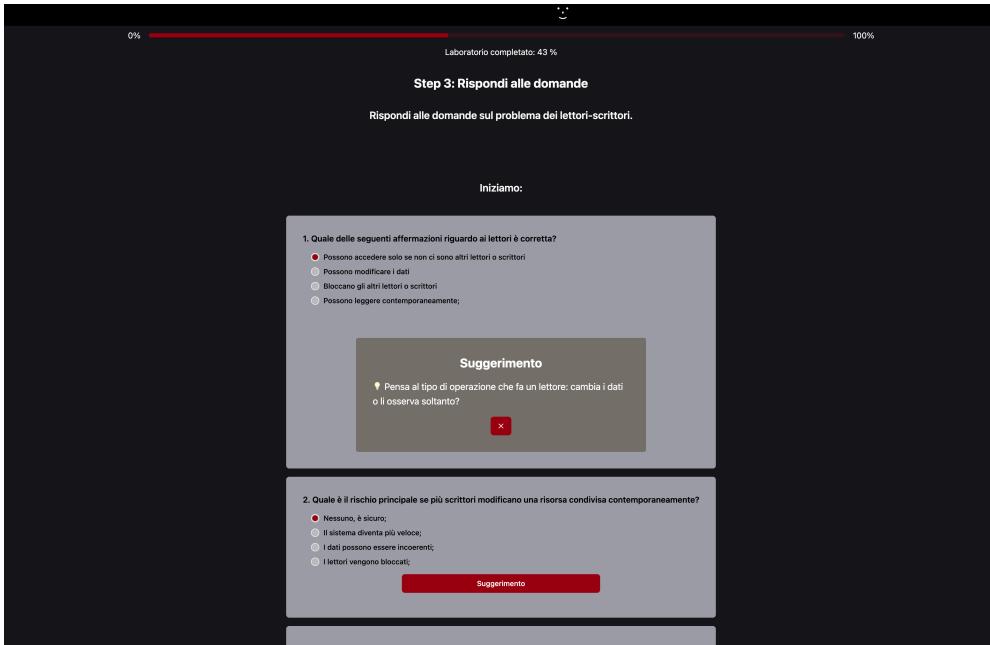


Figura 3.21: Step 3 del laboratorio

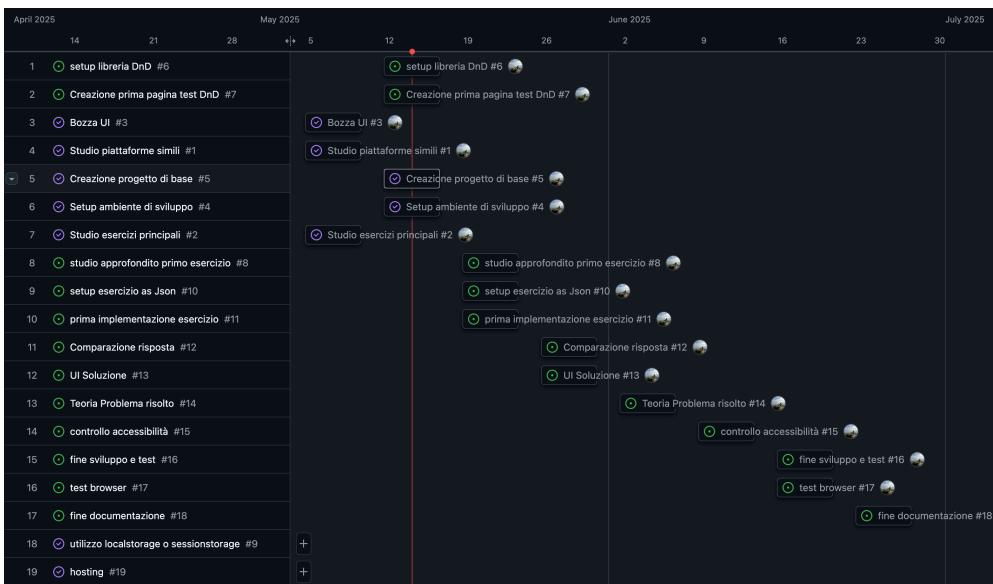


Figura 3.22: UC08: Salvataggio user

Attori coinvolti

- **Attori Primari:** Utente
- **Attori Secondari:** Firebase

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'admin ha caricato lo step 3.
- Il sistema ha caricato *currentStep* correttamente.

- **Postcondizioni:**

- L'utente visualizza lo step 3.
- L'utente può interagire con esso.

Scenario principale

- L'utente attende che l'admin carichi lo step 3.
- Il sistema visualizza lo step 3.
- L'utente può interagire con esso.

UC4.4: Visualizzazione Step 4

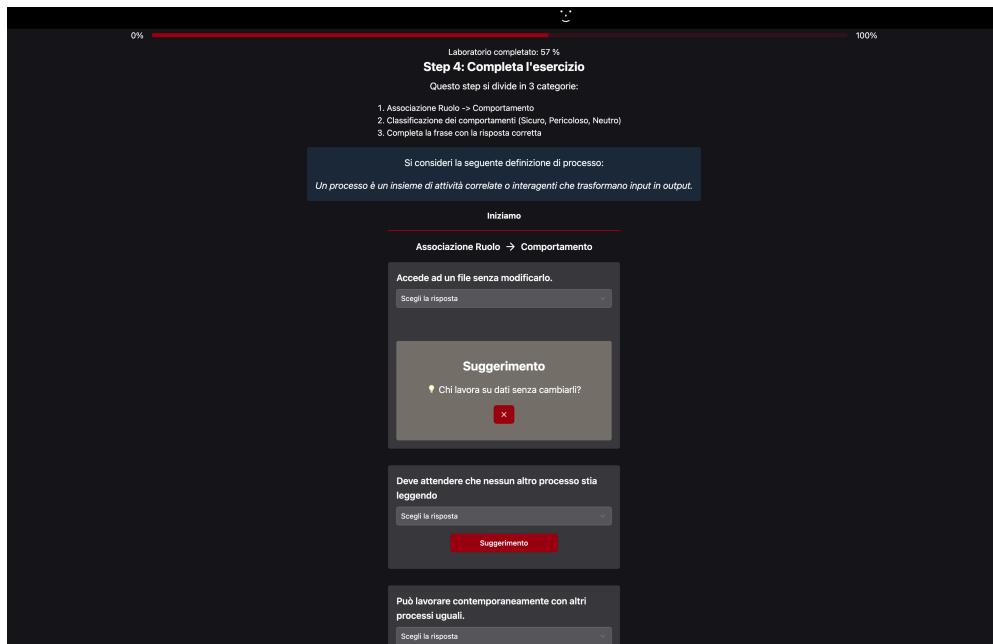


Figura 3.23: Step 4 del laboratorio

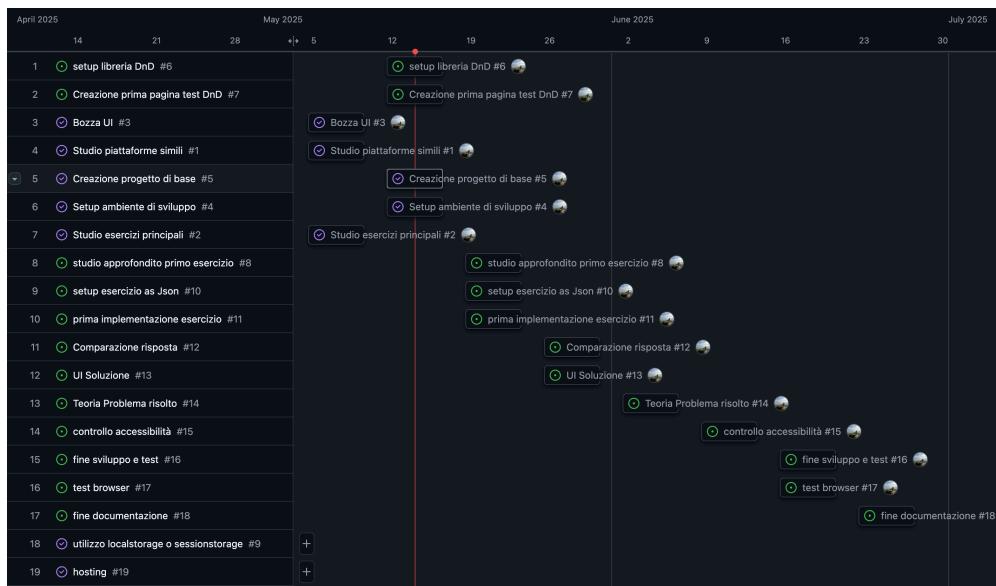


Figura 3.24: UC08: Salvataggio user

Attori coinvolti

- **Attori Primari:** Utente
- **Attori Secondari:** Firebase

Precondizioni e Postcondizioni

- **Precondizioni:**
 - Il sistema è connesso e funzionante.
 - L'admin ha caricato lo step 4.
 - Il sistema ha caricato *currentStep* correttamente.
- **Postcondizioni:**
 - L'utente visualizza lo step 4.
 - L'utente può interagire con esso.

Scenario principale

- L'utente attende che l'admin carichi lo step 4.
- Il sistema visualizza lo step 4.
- L'utente può interagire con esso.

UC4.5: Visualizzazione Step 5

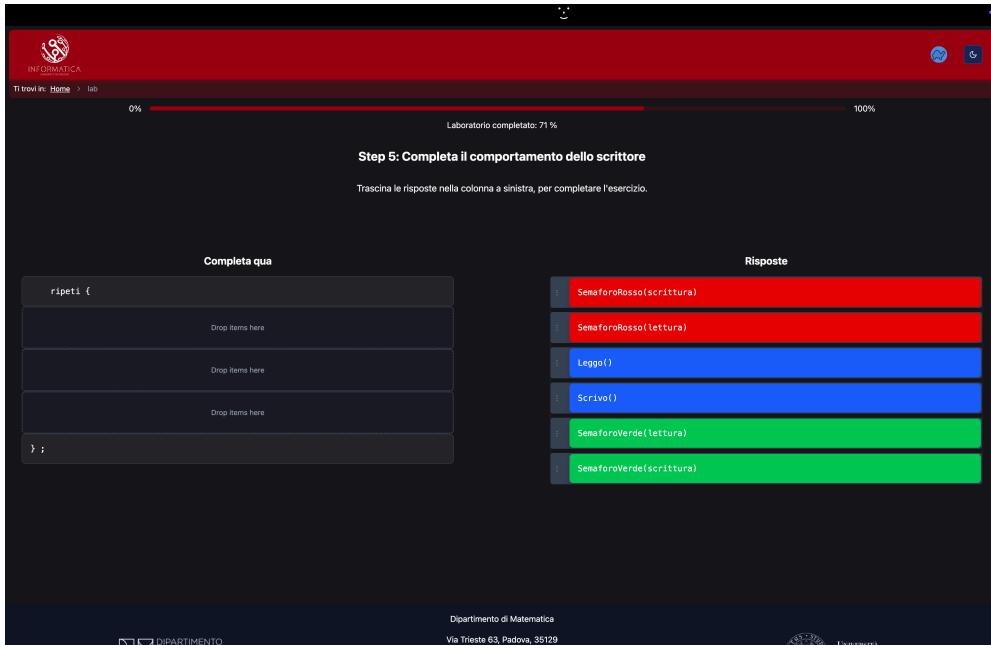


Figura 3.25: Step 5 del laboratorio

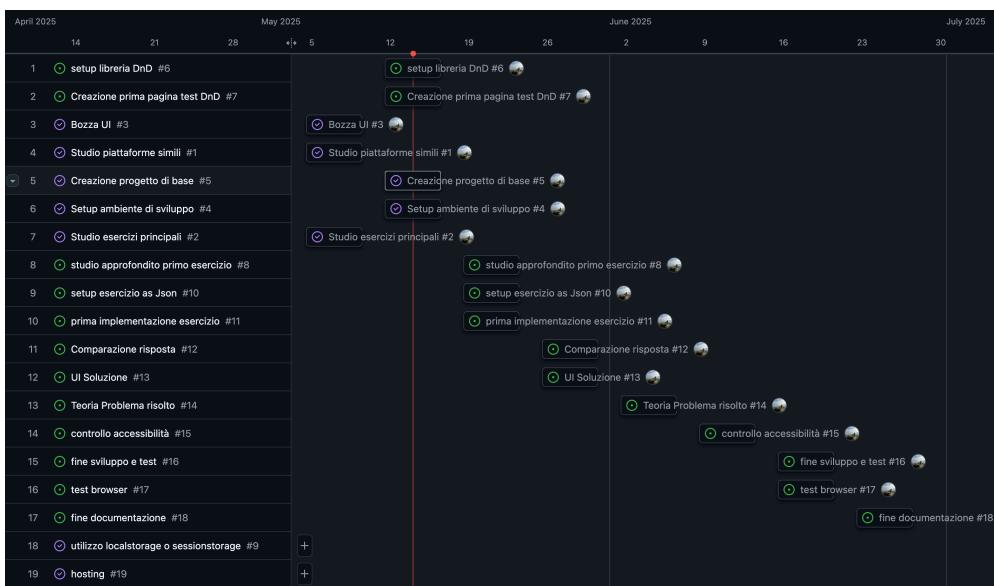


Figura 3.26: UC08: Salvataggio user

Attori coinvolti

- **Attori Primari:** Utente
- **Attori Secondari:** Firebase

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'admin ha caricato lo step 5.
- Il sistema ha caricato *currentStep* correttamente.

- **Postcondizioni:**

- L'utente visualizza lo step 5.
- L'utente può interagire con esso.

Scenario principale

- L'utente attende che l'admin carichi lo step 5.
- Il sistema visualizza lo step 5.
- L'utente può interagire con esso.

Inclusioni

- UC4.1.1: Drag and Drop

User story

- Come utente, voglio visualizzare lo step 5 del laboratorio per interagire con esso e completare l'attività.

UC4.6: Visualizzazione Step 6

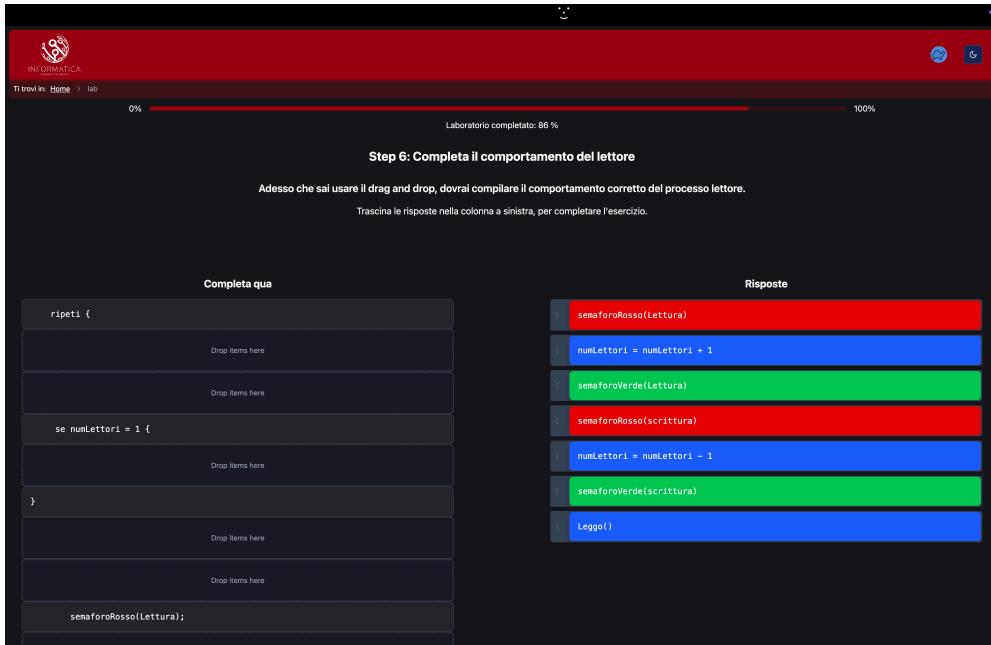


Figura 3.27: Step 6 del laboratorio

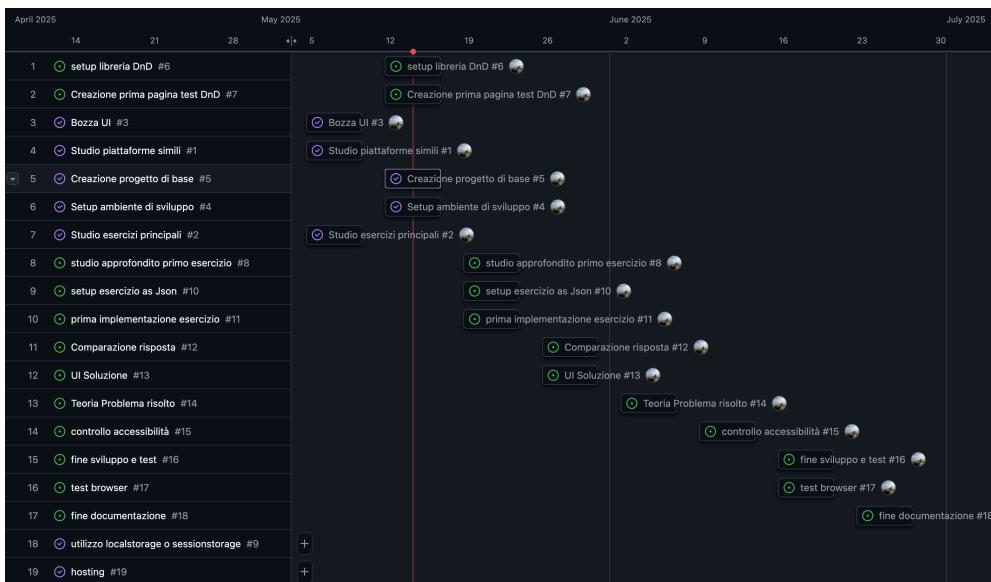


Figura 3.28: UC08: Salvataggio user

Attori coinvolti

- Attori Primari:** Utente
- Attori Secondari:** Firebase

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'admin ha caricato lo step 6.
- Il sistema ha caricato *currentStep* correttamente.

- **Postcondizioni:**

- L'utente visualizza lo step 6.
- L'utente può interagire con esso.

Scenario principale

- L'utente attende che l'admin carichi lo step 6.
- Il sistema visualizza lo step 6.
- L'utente può interagire con esso.

Inclusioni

- UC4.1.1: Drag and Drop

User story

- Come utente, voglio visualizzare lo step 6 del laboratorio per interagire con esso e completare l'attività.

3.6.5 UC05: Visualizzazione pagina di chiusura laboratorio

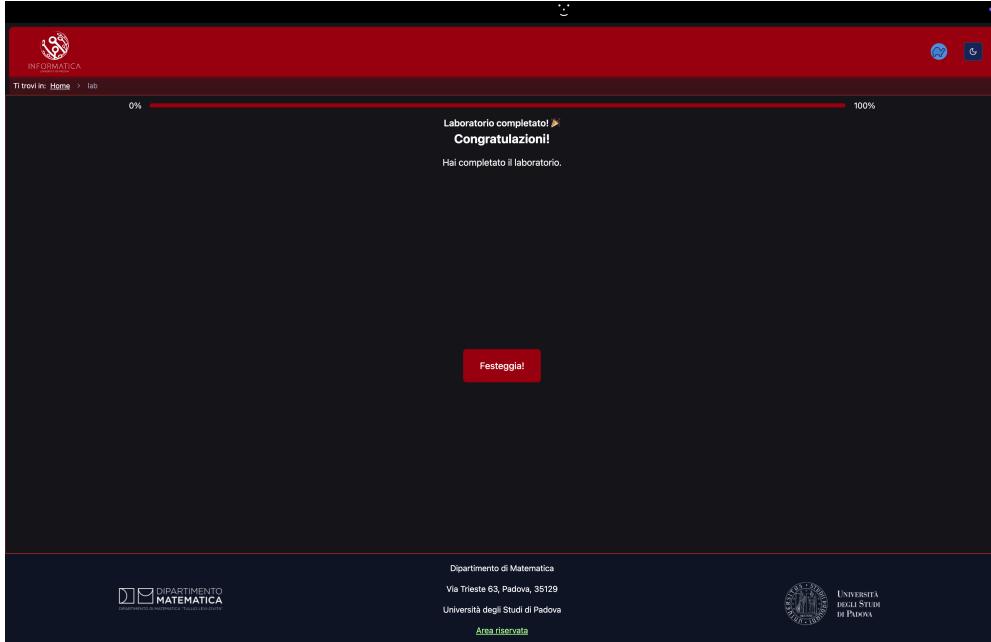


Figura 3.29: Pagina di chiusura del laboratorio

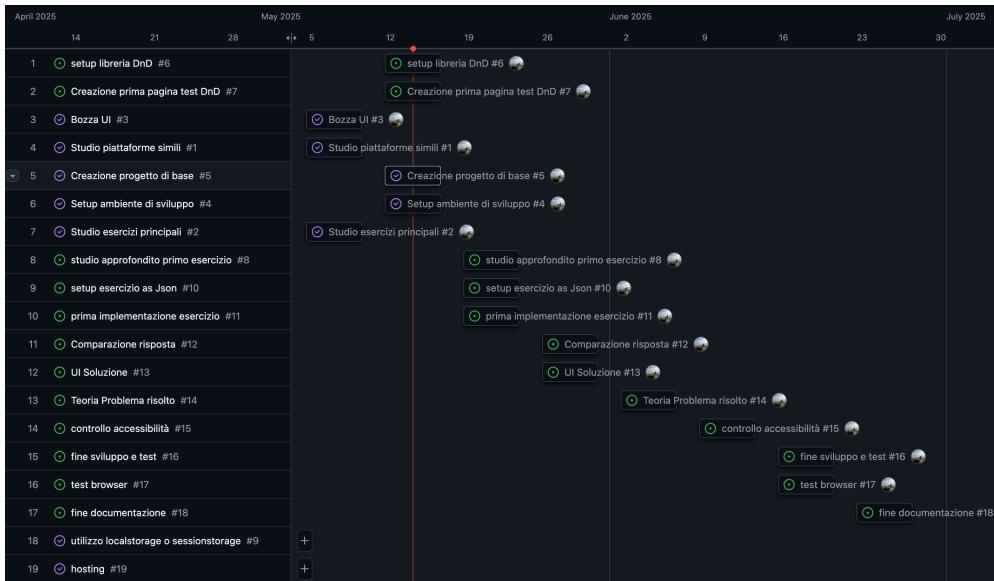


Figura 3.30: UC05: Visualizzazione pagina di chiusura laboratorio

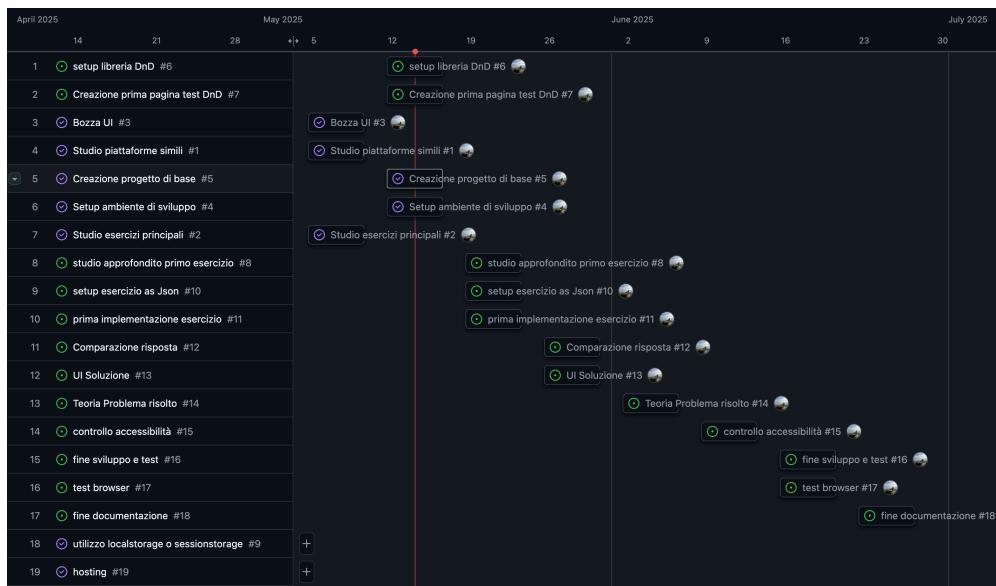


Figura 3.31: UC08: Salvataggio user

Attori coinvolti

- **Attori Primari:** Utente
- **Attori Secondari:** Firebase

Precondizioni e Postcondizioni

- **Precondizioni:**
 - Il sistema è connesso e funzionante.
 - L'admin ha caricato lo step 7.
 - Il sistema ha caricato *currentStep* correttamente.
- **Postcondizioni:**
 - L'utente visualizza lo step 7.
 - L'utente può interagire con esso.

Scenario principale

- L'utente attende che l'admin carichi lo step 7.
- Il sistema visualizza lo step 7.
- L'utente può interagire con esso.

User Story

- Come utente, voglio visualizzare lo step 7 del laboratorio per finire l'attività laboratoriale.

3.7 Definizione dei casi d'uso Utente Admin

L'utente Admin è considerabile come una generalizzazione dell'utente target, in quanto ha accesso a tutte le funzionalità della WebApp, ma con privilegi aggiuntivi che gli consentono di gestire il sistema. L'utente Admin, tuttavia, non necessita di passare per la registrazione avendo già le credenziali (in questo caso salvate sul file .env). Si considerino quindi solo i seguenti casi d'uso, specifici per l'utente Admin, da sommare ai casi d'uso appena elencati.

3.7.1 UC06: Login Admin

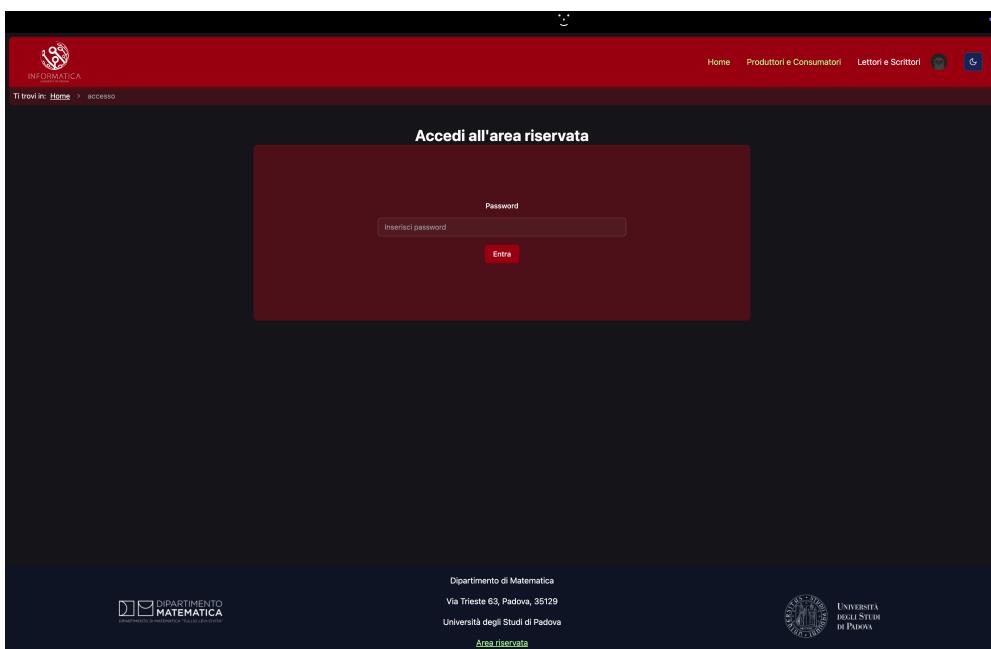


Figura 3.32: Pagina di accesso Admin

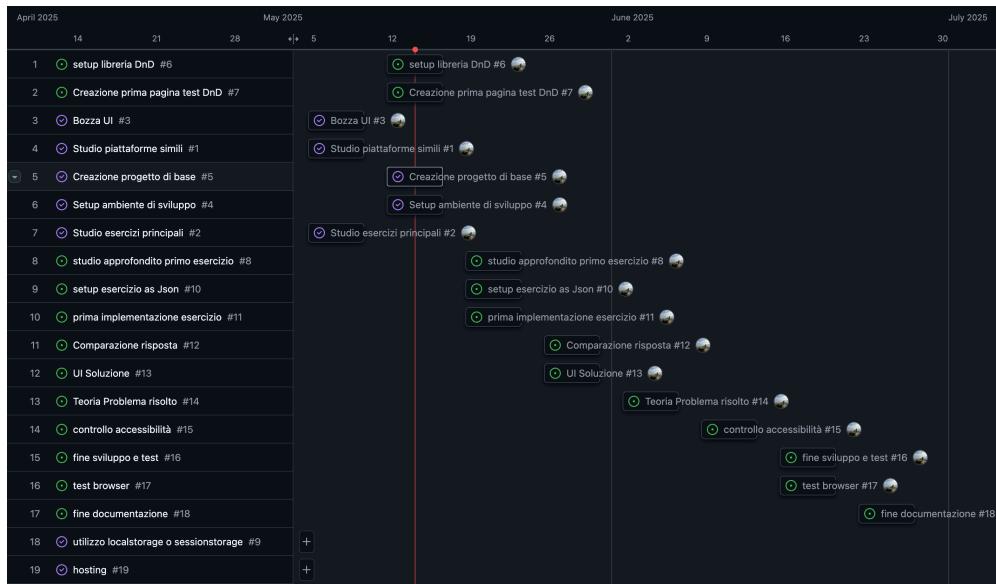


Figura 3.33: Schermata di accesso Admin

Attori coinvolti

- **Attori Primari:** Utente Admin

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'utente Admin ha le credenziali di accesso.

- **Postcondizioni:**

- L'utente Admin è autenticato e può accedere alle funzionalità di amministrazione.

Scenario principale

- L'utente Admin apre l'interfaccia di Thinky direttamente alla pagina di accesso.
- Il sistema mostra il form di accesso.
- L'utente Admin inserisce le credenziali di accesso.
- Il sistema verifica le credenziali.
- Le credenziali sono valide, l'utente Admin viene autenticato.

Estensioni

- UC6.1: Visualizzazione messaggio di errore se le credenziali sono errate.

User Story

- Come utente Admin, voglio poter accedere al sistema per gestire le funzionalità di amministrazione.

UC6.1: Visualizzazione messaggio di errore se le credenziali sono errate

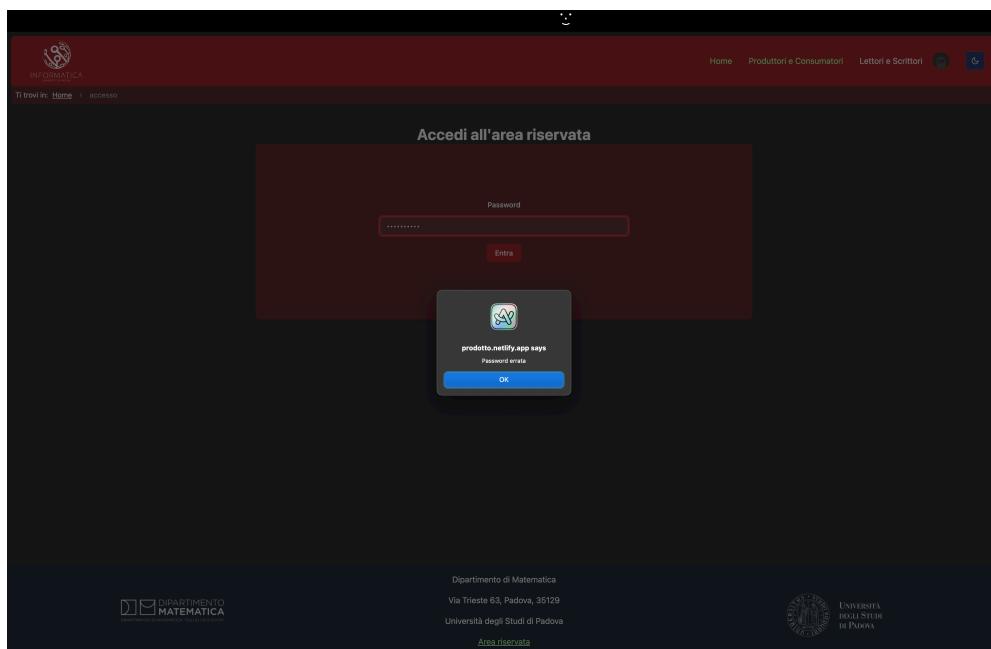


Figura 3.34: Errore di accesso

Attori coinvolti

- **Attori Primari:** Utente Admin

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'utente Admin ha inserito credenziali errate.

- **Postcondizioni:**

- Il sistema visualizza un messaggio di errore.

Scenario principale

- L'utente Admin apre l'interfaccia di Thinky direttamente alla pagina di accesso.
- Il sistema mostra il form di accesso.
- L'utente Admin inserisce le credenziali errate.
- Il sistema verifica le credenziali.
- Le credenziali non sono valide, il sistema visualizza un messaggio di errore.

User Story

- Quando inserisco credenziali errate, il sistema mostra un messaggio di errore così posso correggerle e ripetere l'accesso.

3.7.2 UC07: Visualizzazione homepage Admin

Benvenuto nell'area riservata

Ecco la lista degli utenti registrati:

#	Nome Utente	Scuola	Data di Registrazione
1	Cane Blu	Altro	mar 24 giugno 2025 alle ore 17:42
2	Riccio Lilla	Altro	mar 24 giugno 2025 alle ore 17:46
3	Gatto Nero	Liceo Scientifico	mar 24 giugno 2025 alle ore 18:14

Questa tabella mostra gli utenti registrati, la loro scuola e la data di registrazione.

Dipartimento di Matematica
Via Trieste 63, Padova, 35129
Università degli Studi di Padova

Area riservata

Figura 3.35: Visualizzazione homepage Admin

Attori coinvolti

- **Attori Primari:** Utente Admin

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'utente Admin è autenticato.

- **Postcondizioni:**

- L'utente Admin visualizza la homepage di amministrazione.

Scenario principale

- L'utente ha effettuato l'accesso al sistema.
- Il sistema visualizza la homepage di amministrazione.

Inclusioni

- UC07.1: Visualizzazione tab lista utenti registrati.
- UC07.2: Visualizzazione tab gestione laboratorio.

User Story

- Come utente Admin, voglio visualizzare la homepage di amministrazione per gestire le funzionalità del sistema.

UC07.1: Visualizzazione scheda lista utenti registrati

Ecco la lista degli utenti registrati:			
#	Nome Utente	Scuola	Data di Registrazione
1	Cane Blu	Altro	mar 24 giugno 2025 alle ore 17:42
2	Riccio Lilla	Altro	mar 24 giugno 2025 alle ore 17:46
3	Gufo Nero	Liceo Scientifico	mar 24 giugno 2025 alle ore 18:14

Questa tabella mostra gli utenti registrati, la loro scuola e la data di registrazione.

Figura 3.36: Visualizzazione tabella lista utenti registrati

Attori coinvolti

- **Attori Primari:** Utente Admin
- **Attori Secondari:** GitHub

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'utente Admin è autenticato.
- Le API di GitHub sono disponibili e funzionanti.

- **Postcondizioni:**

- L'utente Admin visualizza la tabella con la lista degli utenti registrati.

Scenario principale

- L'utente Admin accede alla homepage di amministrazione.
- Il sistema visualizza la tabella con la lista degli utenti registrati.

Estensioni

- UC7.1.1: Visualizzazione messaggio per lista vuota.
- UC7.1.2: Visualizzazione messaggio di errore se il caricamento della lista fallisce.

User Story

- Come utente Admin, voglio visualizzare la lista degli utenti registrati per gestire gli utenti del sistema.

UC7.1.1: Visualizzazione messaggio per lista vuota

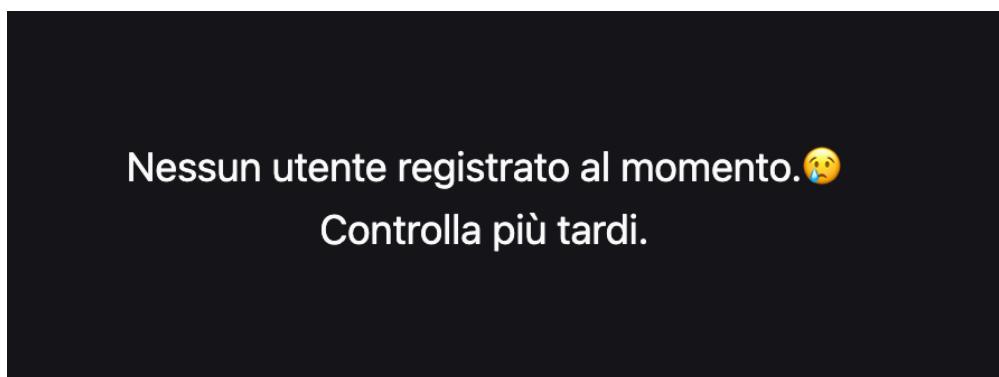


Figura 3.37: Errore lista utenti vuota

Attori coinvolti

- **Attori Primari:** Utente Admin
- **Attori Secondari:** GitHub

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'utente Admin è autenticato.
- Le API di GitHub sono disponibili e funzionanti.
- Non ci sono utenti registrati nel sistema.
- **Postcondizioni:**
 - Il sistema visualizza un messaggio che informa l'utente che la lista è vuota.

Scenario principale

- L'utente Admin accede alla homepage di amministrazione.
- Il sistema verifica se ci sono utenti registrati.
- Non ci sono utenti registrati, il sistema visualizza un messaggio che informa l'utente che la lista è vuota.

User Story

- Quando non ci sono utenti registrati, il sistema mostra un messaggio che informa l'utente Admin che la lista è vuota.

UC7.1.2: Visualizzazione messaggio di errore se il caricamento della lista fallisce

Attori coinvolti

- **Attori Primari:** Utente Admin
- **Attori Secondari:** GitHub

Precondizioni e Postcondizioni

- **Precondizioni:**
 - Il sistema è connesso e funzionante.
 - L'utente Admin è autenticato.
 - Le API di GitHub sono disponibili e funzionanti.
 - Si è verificato un errore durante il caricamento della lista degli utenti registrati.
- **Postcondizioni:**

- Il sistema visualizza un messaggio di errore.

Scenario principale

- L'utente Admin accede alla homepage di amministrazione.
- Il sistema tenta di caricare la lista degli utenti registrati.
- Si verifica un errore durante il caricamento della lista.
- Il sistema visualizza un messaggio di errore.

User Story

- Se il caricamento della lista degli utenti registrati fallisce, il sistema mostra un messaggio di errore per informare l'utente Admin.

UC07.2: Visualizzazione tab gestione laboratorio

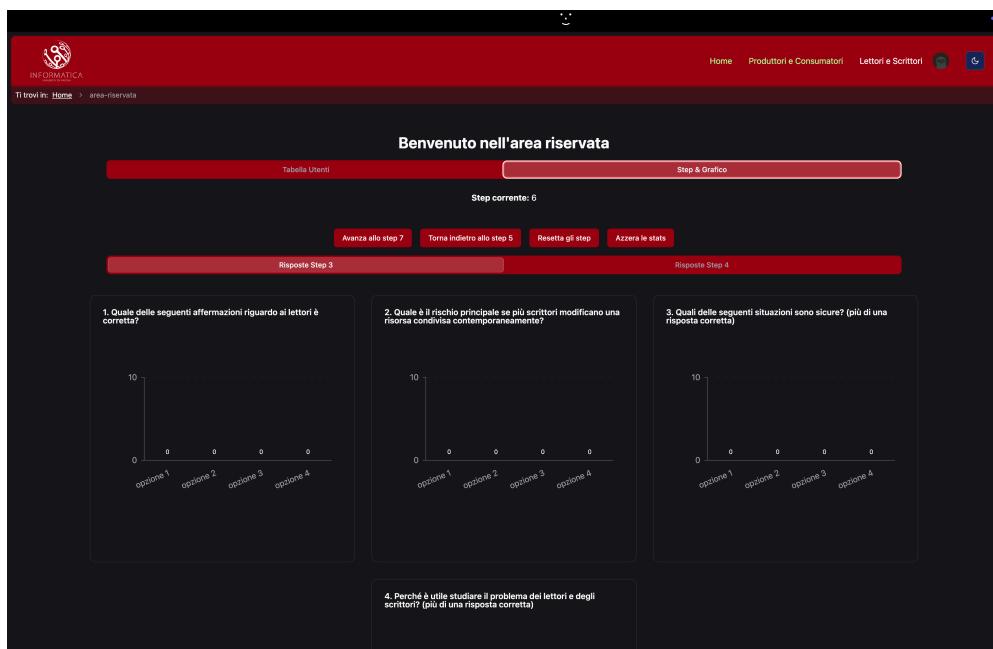


Figura 3.38: visualizzazione tab gestione laboratorio

Attori coinvolti

- **Attori Primari:** Utente Admin
- **Attori Secondari:** Firebase, GitHub

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'utente Admin è autenticato.
- Le API di GitHub sono disponibili e funzionanti.

- **Postcondizioni:**

- L'utente Admin visualizza la Tab con le informazioni di laboratorio.

Scenario principale

- L'utente Admin accede alla homepage di amministrazione.
- Il sistema visualizza la Tab con le informazioni di laboratorio.

Inclusioni

- UC07.2.1: Visualizzazione pulsanti della gestione step del laboratorio.
- UC07.2.2: Visualizzazione Tab Grafici Step 3
- UC07.2.3: Visualizzazione Tab Grafici Step 4

User Story

- Come utente Admin, voglio visualizzare la Tab con le informazioni di laboratorio per gestire gli step del laboratorio e i grafici delle risposte date nei laboratori.

UC07.2.1: Visualizzazione pulsanti della gestione step del laboratorio



Figura 3.39: Pulsanti per la gestione degli step del laboratorio

Attori coinvolti

- **Attori Primari:** Utente Admin
- **Attori Secondari:** Firebase, GitHub

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'utente Admin è autenticato.
- Le API di GitHub sono disponibili e funzionanti.

- **Postcondizioni:**

- L'utente Admin visualizza i pulsanti per la gestione degli step del laboratorio.

Scenario principale

- L'utente Admin accede alla Tab con le informazioni di laboratorio.
- Il sistema visualizza i pulsanti per la gestione degli step del laboratorio.
- L'utente Admin può interagire con i pulsanti per gestire gli step del laboratorio.

Generalizzazioni

- UC07.2.1.1: Visualizzazione pulsante per l'avanzamento ad un nuovo step.
- UC07.2.1.2: Visualizzazione pulsante per ritornare allo step precedente.
- UC07.2.1.3: Visualizzazione pulsante per azzeramento statistiche.
- UC07.2.1.4: Visualizzazione pulsante per il reset degli step.

User Story

- Come utente Admin, voglio visualizzare i pulsanti per la gestione degli step del laboratorio per poterli gestire in modo efficiente.

UC07.2.1.1: Visualizzazione pulsante per l'avanzamento ad un nuovo step

Attori coinvolti

- **Attori Primari:** Utente Admin

- **Attori Secondari:** Firebase

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'utente Admin è autenticato.
- Il sistema ha caricato *currentStep* correttamente.

- **Postcondizioni:**

- L'utente Admin visualizza il pulsante per l'avanzamento ad un nuovo step.

Scenario principale

- L'utente Admin accede alla Tab con le informazioni di laboratorio.
- Il sistema visualizza il pulsante per l'avanzamento ad un nuovo step.
- L'utente Admin può interagire con il pulsante per avanzare ad un nuovo step.

User Story

- Come utente Admin, voglio visualizzare il pulsante per l'avanzamento ad un nuovo step per poter gestire gli step del laboratorio in modo efficiente.

UC07.2.1.2: Visualizzazione pulsante per ritornare allo step precedente

Attori coinvolti

- **Attori Primari:** Utente Admin
- **Attori Secondari:** Firebase

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'utente Admin è autenticato.

- Il sistema ha caricato *currentStep* correttamente.
- *currentStep* ≥ 1
- **Postcondizioni:**
 - Il pulsante per tornare indietro di uno step è cliccabile.

Scenario principale

- L'utente Admin accede alla Tab con le informazioni di laboratorio.
- Il sistema visualizza il pulsante per tornare indietro di uno step.
- L'utente Admin può interagire con il pulsante per tornare indietro di uno step.

User Story

- Come utente Admin, voglio visualizzare il pulsante per tornare indietro di uno step per poter gestire gli step del laboratorio in modo efficiente.

UC07.2.1.3: Visualizzazione pulsante per azzeramento statistiche

Attori coinvolti

- **Attori Primari:** Utente Admin
- **Attori Secondari:** Firebase, GitHub

Precondizioni e Postcondizioni

- **Precondizioni:**
 - Il sistema è connesso e funzionante.
 - L'utente Admin è autenticato.
 - Le API di GitHub sono disponibili e funzionanti.
 - Il sistema ha caricato *currentStep* correttamente.
 - L'utente Admin ha cliccato sul pulsante per l'azzeramento delle statistiche.
- **Postcondizioni:**

- Il pulsante per l'azzeramento delle statistiche resetta *users.json* e *chartAnswer.json* (maggiori informazioni su questi due file e il loro scopo sono elencate al [prossimo capitolo](#)).

Scenario Principale

- L'utente Admin accede alla Tab con le informazioni di laboratorio.
- Il sistema visualizza il pulsante per l'azzeramento delle statistiche.
- L'utente Admin può interagire con il pulsante per azzerare le statistiche.
- *users.json* e *chartAnswer.json* vengono resettati.

User Story

- Come utente Admin, voglio visualizzare il pulsante per l'azzeramento delle statistiche per poter resettare le statistiche del laboratorio in modo rapido senza dover accedere al servizio di *information storage* dove sono contenuti i file.

UC07.2.1.4: Visualizzazione pulsante per il reset degli step

Attori coinvolti

- **Attori Primari:** Utente Admin
- **Attori Secondari:** Firebase

Precondizioni e Postcondizioni

- **Precondizioni:**
 - Il sistema è connesso e funzionante.
 - L'utente Admin è autenticato.
 - Il sistema ha caricato *currentStep* correttamente.
 - L'utente Admin ha cliccato sul pulsante per il reset degli step.
- **Postcondizioni:**
 - Il pulsante per il reset degli step resetta *currentStep* a 0.

Scenario principale

- L'utente Admin accede alla Tab con le informazioni di laboratorio.
- Il sistema visualizza il pulsante per il reset degli step.
- L'utente Admin può interagire con il pulsante per resettare gli step in maniera rapida.
- *currentStep* viene resettato a 0.

User Story

- Come utente Admin, voglio visualizzare il pulsante per il reset degli step per poter resettare gli step del laboratorio in modo rapido senza dover accedere al servizio di *information storage* dove è contenuto il file *step.json*.

UC07.2.2: Visualizzazione Tab Grafici Step 3

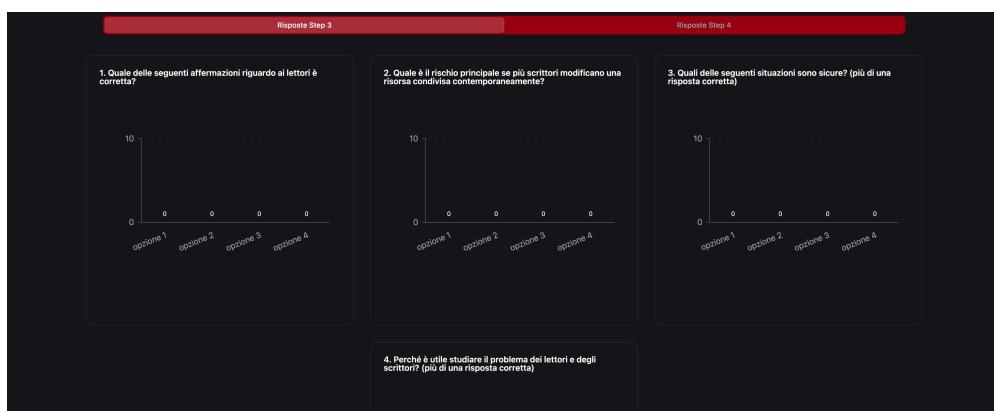


Figura 3.40: UC08: Salvataggio user

Attori coinvolti

- **Attori Primari:** Utente Admin
- **Attori Secondari:** GitHub

Precondizioni e Postcondizioni

• Precondizioni:

- Il sistema è connesso e funzionante.
- L'utente Admin è autenticato.
- Le API di GitHub sono disponibili e funzionanti.
- Il sistema ha caricato i dati di *chartAnswer.json* correttamente.

- **Postcondizioni:**

- L'utente Admin visualizza la Tab con i grafici delle risposte date dagli utenti nello step 3.

Scenario principale

- L'utente Admin accede alla Tab con le informazioni di laboratorio.
- Il sistema visualizza la Tab con i grafici delle risposte date dagli utenti nello step 3.
- L'utente Admin può interagire con i grafici per visualizzare le risposte date dagli utenti nello step 3.

Estensioni

- UC7.2.2.1: Visualizzazione messaggio di informazione se il file `chartAnswer.json` è vuoto o avviene un errore generico.

User Story

- Come utente Admin, voglio visualizzare la Tab con i grafici delle risposte date dagli utenti nello step 3 per analizzare le risposte degli utenti e migliorare l'esperienza del laboratorio.

UC7.2.2.1: Visualizzazione messaggio di informazione se il file

`chartAnswer.json` è vuoto o avviene un errore generico

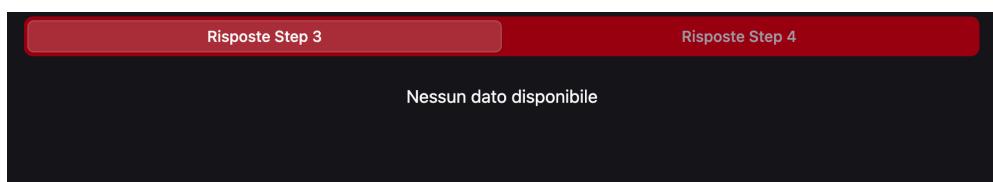


Figura 3.41: Errore durante il caricamento dei dati

Attori coinvolti

- **Attori Primari:** Utente Admin
- **Attori Secondari:** GitHub

Precondizioni e Postcondizioni

- **Precondizioni:**

- Il sistema è connesso e funzionante.
- L'utente Admin è autenticato.
- Il file *chartAnswer.json* è vuoto o si verifica un errore durante il caricamento dei dati.
- **Postcondizioni:**
 - Il sistema visualizza un messaggio di informazione che indica che il file *chartAnswer.json* è vuoto.

Scenario principale

- L'utente Admin accede alla Tab con i grafici delle risposte date dagli utenti nello step 3.
- Il sistema tenta di caricare i dati dal file *chartAnswer.json*.
- Il file *chartAnswer.json* è vuoto o si verifica un errore durante il caricamento dei dati.
- Il sistema visualizza un messaggio di informazione che indica che il file *chartAnswer.json* è vuoto.

User Story

- Quando il file *chartAnswer.json* è vuoto o si verifica un errore durante il caricamento dei dati, il sistema mostra un messaggio di informazione per informare l'utente Admin che non ci sono dati disponibili per lo step 3.

UC07.2.3: Visualizzazione Tab Grafici Step 4

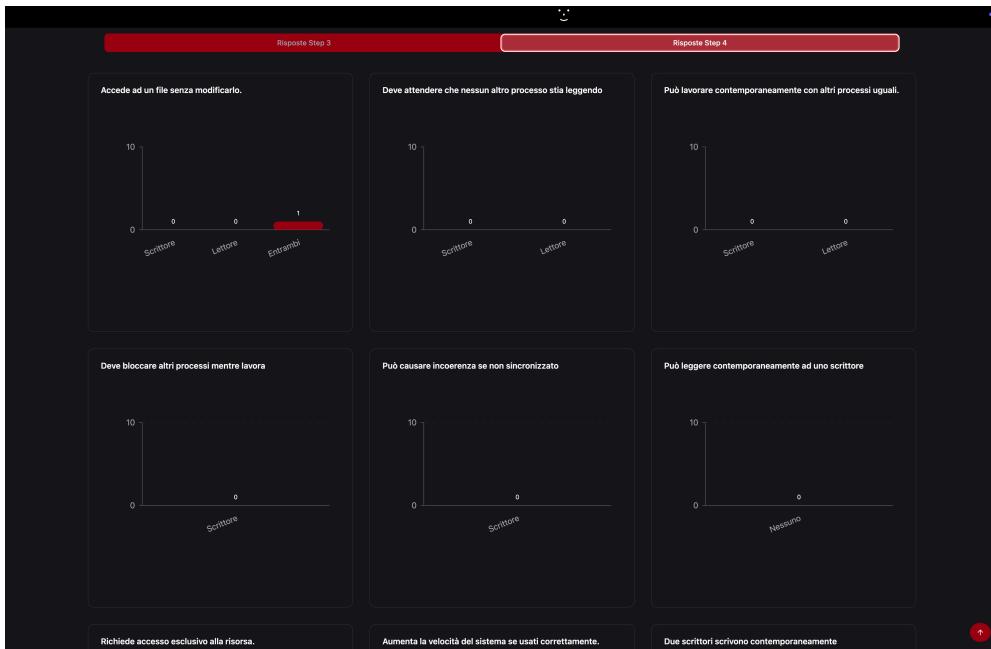


Figura 3.42: UC08: Salvataggio user

Attori coinvolti

- **Attori Primari:** Utente Admin
- **Attori Secondari:** GitHub

Precondizioni e Postcondizioni

- **Precondizioni:**
 - Il sistema è connesso e funzionante.
 - L'utente Admin è autenticato.
 - Le API di GitHub sono disponibili e funzionanti.
 - Il sistema ha caricato i dati di *chartAnswer.json* correttamente.
- **Postcondizioni:**
 - L'utente Admin visualizza la Tab con i grafici delle risposte date dagli utenti nello step 4.

Scenario principale

- L'utente Admin accede alla Tab con le informazioni di laboratorio.

- Il sistema visualizza la Tab con i grafici delle risposte date dagli utenti nello step 4.
- L'utente Admin può interagire con i grafici per visualizzare le risposte date dagli utenti nello step 4.

Estensioni

- UC7.2.2.1: Visualizzazione messaggio di informazione se il file `chartAnswer.json` è vuoto o avviene un errore generico.

User Story

- Come utente Admin, voglio visualizzare la Tab con i grafici delle risposte date dagli utenti nello step 4 per analizzare le risposte degli utenti e migliorare l'esperienza del laboratorio.

Capitolo 4

Implementazione

Espone come le scelte descritte nel capitolo precedente sono state implementate, attraverso una panoramica del codice sorgente, delle feature di accessibilità e delle best practices attuate durante il processo di sviluppo.

Capitolo 5

Retrospettiva finale

Glossario

API: Interfaccia di programmazione delle applicazioni. Permette la comunicazione tra diverse applicazioni o servizi. 12, 16

Firebase: Piattaforma di sviluppo di applicazioni web e mobili che offre servizi come database in tempo reale, autenticazione e hosting. 15

Git: Sistema di controllo versione distribuito. Permette di tenere traccia delle modifiche apportate a file e cartelle nel tempo. 2

GitHub: Piattaforma di hosting per progetti software che utilizza Git come sistema di controllo versione. Permette la collaborazione tra sviluppatori e la gestione del codice sorgente. 2, 12, 15

Kanban: Sistema di gestione del lavoro che utilizza schede per visualizzare il flusso di lavoro e le attività in corso. 3

Piano di Lavoro: Piano di lavoro. Documento che descrive le attività e gli obiettivi di un tirocinio o di un progetto. 2

SEO: ing: Search Engine Optimization. Ottimizzazione per i motori di ricerca. Insieme di tecniche e pratiche per migliorare la visibilità di un sito web sui motori di ricerca. 12

Scrum: Framework Agile per la gestione dei progetti. Si basa su iterazioni brevi (sprint) e su riunioni regolari per monitorare i progressi. 4

WebApp: abbr. Applicazione Web iv

backlog: Backlog. Elenco di attività o funzionalità da completare in un progetto. In questo caso, si riferisce all'elenco delle funzionalità e dei bug da risolvere nel progetto. 3

proponente: Colui che propone un progetto o un'idea. In questo caso, si riferisce all'Università di Padova 11

server-side components: Componenti lato server. Permettono di eseguire il rendering delle pagine sul server prima di inviarle al client, migliorando le prestazioni e l'ottimizzazione SEO. 12

sprint: Sprint. Periodo di tempo definito in cui un team di sviluppo (o in questo caso, un solo dev) lavora per completare un insieme specifico di attività o obiettivi. 3

Bibliografia

- [1] Robert C. Martin, Kent Beck, Martin Fowler, e altri, «Manifesto delle Agile Software Development». [Online]. Disponibile su: <http://agilemanifesto.org/>