BREW BEER OK

Zoran Stojkov, Chengjie Zhou, Paolo Cirilli

APPELLO DI FEBRARIO 2022

1 Premessa

Essendo questo una simulazione di sviluppo del software assegnatoci dai docenti del corso, quest'ultimi verranno definiti con clienti da ora in poi. Abbiamo cercato in più modi di rendere più realistica possibile lo sviluppo di questo progetto cercando di valorizzare i punti di forza di ogni elemento e cercando di lavorare come un team di sviluppo a cui è stato commissionato lo sviluppo di questo progetto con tempo di consegna massima di un mese.

2 Analisi

2.1 Metodologia di sviluppo progettuale

Per sviluppare il software che è stato assegnato al nostro gruppo abbiamo applicato la metodologia di modellazione UP, conosciuto anche come RUP ,è un processo iterativo dove si hanno modelli agili, che si basano su consegna incrementale, con iterazioni brevi e timebox. Si hanno quattro fasi nel modello RUP :

- Avviamento: dove abbiamo posto stime approssimative sui requisti.
- Elaborazione: dove abbiamo definito i requisti e posto stime più ponderate.
- Costruzione: dove abbiamo iniziato l'implementazione dei requisti.
- Transizione: dove abbiamo fatto la parte di testing per poi rilasciare il tag di consegna.

2.2 Requisiti del progetto

I vari requisti sono stati sviluppati in base alla richiesta del cliente.

2.2.1 Requisti funzionali

- Mantenimento lista ricette.
- Mantenimento lista degli ingredienti e equipaggiamento del user.
- Aggiornamento lista ingredienti e equipaggiamento.

- Funzioni CRUD di una ricetta.
- Funzione ci creazione di una nota su una specifica ricetta
- Mantenimento cronologia delle ricette prodotte dal utente
- Implementazione della feature "What should I brew today?"

2.3 Attori

L'attore principale è l'utente registrato che è una generalizzazione di utente (non registrato) che ha la capacità di fare operazioni CRUD delle ricette, inoltre può salvare gli ingredienti e equipaggiamento a disposizione in quel momento.

2.4 Diagramma dei casi d'uso

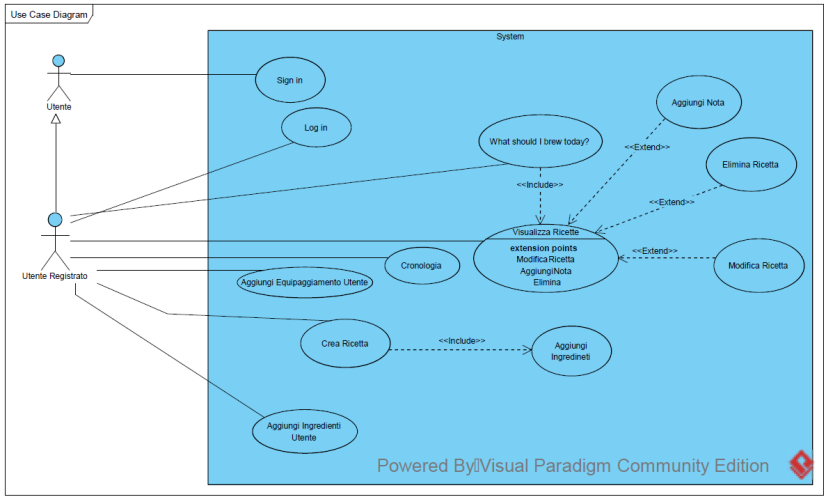

Figura 1: Diagramma dei casi d'uso

2.5 Descrizione dettagliata casi d'uso

2.5.1 Caso d'uso Login

- Caso d'uso: Login.
- Portata: Web app Brew Beer Ok.
- Attore primario: Utente Registrato.
- Parti interessate e interessi : Utente registrato : Vuole accedere al sito.
- Precondizioni: L'utente è già registrato.
- Garanzia di successo: L'utente accede al sito.
- Scenario principale di successo:

1. L'utente accede al sito.
2. L'utente compila il form di login.
3. Il sistema verifica se l'utente è registrato.
4. L'utente accede al sito.

- Estensioni :
  - 1a il sito non è raggiungibile, verrà visualizzato una schermata di errore e la pagina andrà ricaricata.
  - 2a L'utente inserisce nel form campi errati o caratteri incompatibili, verrà segnalato l'errore e l'utente sarà richiesto di inserire i dati.

2.5.2 Caso d'uso SignIn

- Caso d'uso: SingIn.
- Portata: Web app Brew Beer OK.
- Attore primario: Utente.
- Parti interessate e interessi : Utente generico: Vuole fare una registrazione al sito, fruendo di un'interfaccia di facile comprensione.
- Precondizioni: L'utente non è ancora registrato.
- Garanzia di successo: L'utente viene registrato correttamente nel sistema.
- Scenario principale di successo

1. L'utente accede al sito nel quale vuole effettuare la registrazione. L'utente entra nel form di registrazione.
2. L'utente compila con i dati richiesti nel form per la registrazione.
3. Il sistema dopo aver verificato la correttezza dei dati inseriti e di non avere l'utente già presente nel suo database, approva la registrazione.
4. Il sistema registra l'utente.

- Estensioni :
  - 1a il sito non è raggiungibile, verrà visualizzato una schermata di errore e la pagina andrà ricaricata.
  - 2a L'utente inserisce nel form campi errati o caratteri incompatibili, verrà segnalato l'errore e l'utente sarà richiesto di inserire i dati.

2.5.3 Crea Ricette

- Caso d'uso: Crea Ricetta.
- Portata: Web app Brew Beer OK.
- Attore primario: Utente Registrato.
- Parti interessate e interessi : Utente e Recpie.
- Precondizioni: L'utente sia registrato.
- Garanzia di successo: Viene create la ricetta.
- Scenario principale di successo:
    1. L'utente preme sul pulsante (+ in basso a destra).
    2. L'utente aggiunge il titolo, viene eseguito il caso d'uso Aggiungi Ingredienti per ricetta e viene selezionato l'equipaggiamento.
    3. Il sistema crea e salva la ricetta.
    4. L'utente viene reindirizzato alla pagina di home.
- Estensioni :
    1. 2a L'utente inserisce i dati errati, il sistema non finalizzerà fino all'inserimento corretto dei dati.
    2. 2b L'utente non ha eseguito il caso d'uso Aggiungi equipaggiamento, il sistema mostrerà un messaggio di errore e inviterà l'utente ad inserire i suoi equipaggiamenti.

2.5.4 Aggiungi Ingredienti

- Caso d'uso: Aggiungi Ingredienti.
- Portata: Web App (Incluso nel Caso d'uso Crea Ricetta).
- Attore primario: Utente Registrato.
- Parti interessate e interessi : Utente e Recipe.
- Precondizioni: Caso d'uso Crea Ricetta svolto con correttezza.
- Garanzia di successo: Vengono aggiunti correttamente gli ingredienti di una ricetta.
- Scenario principale di successo:
    1. Viene mostrato il form di aggiungnta ingredienti.
    2. L'utente compila aggiunge gli ingredienti in modo corretto.
    3. L'utente salva gli ingredienti.

2.5.5 Visualizza Ricette

- Caso d'uso: Visualizza Ricette.
- Portata: Web App Brew Beer OK.
- Attore primario: Utente Registrato.
- Parti interessate e interessi : Utente Registrato vuole visualizzare tutte le ricette disponibili
- Precondizioni: L'Utente è registrato.
- Garanzia di successo: L'utente visualizza le ricette.
- Scenario principale di successo:
    1. L'utente accede al sito.
    2. Il sito carica le ricette sulla pagina principale della web app.

2.5.6 Modifica Ricette

- Caso d'uso: Modifica Ricetta.
- Portata: Web app Brew Beer OK (Estensione Caso d'uso Visualizza Ricette).
- Attore primario : Utente Registrato.
- Parti interessate e interessi : Utente Registrato e il sistema.
- Precondizioni: Ci sai almeno una ricetta salvata.
- Garanzia di successo: la ricetta viene modificata.
- Scenario principale di successo:
    1. L'utente clicca modifica sulla ricetta da modificare.
    2. L'utente modifica ciò che desidera nel form della ricette.
    3. L'utente preme salva.
    4. Il sistema salva le modifiche alla ricetta selezionata.
    5. L'utente sarà reindirizzato alla pagina principale e la ricetta sarà modificata.
- Estensioni :
    1. 2a L'utente inserisce parametri errati, il sistema non finalizzerà aspettando che vengano inseriti i dati corretti.

2.5.7 Aggiungi nota

- Caso d'uso: Aggiungi Nota.
- Portata: Web app Brew Beer OK (Estensione Caso d'uso Visualizza Ricette).
- Attore primario: Utente Registrato.
- Parti interessate e interessi : Utente Registrato e Recipe.
- Precondizioni: Ci sia almeno una ricetta da modificare.
- Garanzia di successo: La nota sia associata alla ricetta.
- Scenario principale di successo:
    1. L'utente clicca su modifica della ricetta.
    2. Verrà mostrato il from di modifica ricetta.
    3. L'utente scrive la nota nella textbox adibita.
    4. L'utente preme salva.
    5. Il sistema salva la nota.
    6. L'utente viene reindirizzato alla pagina principale e viene mostrata la nota nella ricetta a cui è stata aggiunta.

2.5.8 Elimina Ricetta

- Caso d'uso: Elimina.
- Portata: Web app Brew Beer OK (Estensione Caso d'uso Visualizza Ricette).
- Attore primario: Utente Registrato.
- Parti interessate e interessi : Utente,Ricette.
- Precondizioni: Ci sia almeno una ricetta salvata.
- Garanzia di successo: La ricetta viene eliminata.
- Scenario principale di successo:
    1. L'utente preme il pulsante elimina nella pagina principale.
    2. Il sistema trova la ricetta nel database.
    3. La ricetta viene cancellata e l'utente viene reindirizzato alla pagina di home con i dati aggiornati.

2.5.9 Aggiungi Ingredienti Utente

- Caso d'uso: Aggiungi Ingredienti.
- Portata: Web App Brew Beer OK.
- Attore primario: Utente Registrato.
- Parti interessate e interessi : Utente e Recipe.
- Precondizioni: Caso d'uso Crea Ricetta svolto con correttezza.
- Garanzia di successo: Vengono aggiunti correttamente gli ingredienti di una ricetta.
- Scenario principale di successo:
  1. L'utente clicca il My Ingridients nella app bar.
  2. Viene mostrato il form di aggiunta ingredienti.
  3. L'utente aggiunge gli ingredienti in modo corretto.
  4. L'utente salva gli ingredienti.
  5. Il sistema salva gli ingredienti.
- Estensioni :
  - 2a L'utente inserisce i dati sbagliati, il sistema non finalizzerà fino ad inserimento corretto dei dati.

2.5.10 Aggiungi Equipaggiamento Utente

- Caso d'uso: Aggiungi Equipaggiamento.
- Portata: Web App Brew Beer OK.
- Attore primario: Utente Registrato.
- Parti interessate e interessi : Utente e Equipaggiamento.
- Garanzia di successo: Vengono aggiunti correttamente gli Equipaggiamento dell'utente.
- Scenario principale di successo:
  1. L'utente clicca il My Equipments nella app bar.
  2. Viene mostrato il form di aggiunta equipaggiamento.
  3. L'utente compila il form in modo corretto.
  4. L'utente clicca su salva.
  5. Il sistema salva l'equipaggiamento e vieni reindirizzato alla pagina degli equipment.

- Estensioni :
  - 2a L'utente inserisce i dati sbagliati, il sistema non finalizzerà fino ad inserimento corretto dei dati.

2.5.11 What should i brew today?

- Caso d'uso: What should i brew today?.
- Portata: Web app Brew Beer OK.
- Attore primario: Utente Registrato.
- Parti interessate e interessi : Utente Registrato e UserIngridients.
- Precondizioni: Il form di lista ingredienti sia stato compilato e salvato.
- Garanzia di successo: Vengono restituita la birra che ottimizza gli ingredienti del utente.
- Scenario principale di successo:
  1. L'utente preme il pulsante What should i brew today?.
  2. Il sistema prende i dati in Lista ingredienti e sceglie la birra tra quelle salvate che ottimizzerà gli ingredienti dell'utente.
  3. L'utente viene reindirizzato nella pagina che mostra la birra scelta dal sistema.

2.5.12 Cronologia

- Caso d'uso: Cronologia.
- Portata: Web app Brew Beer OK.
- Attore primario: Utenete Registrato.
- Parti interessate e interessi : Utente e Recipe.
- Precondizioni: Sia stata prodotta almeno una birra (altrimenti la pagina sara vuota).
- Garanzia di successo: Vengono visualizzate le recipe prodotte.
- Scenario principale di successo:
  1. L'utente preme sul pulsante MyChronology.
  2. L'utente viene reindirizzato alla pagina di cronologia dove saranno mostrate le ricette prodotte con la data di produzione.

3 Progettazione

3.1 Diagrammi

3.1.1 Diagramma di dominio

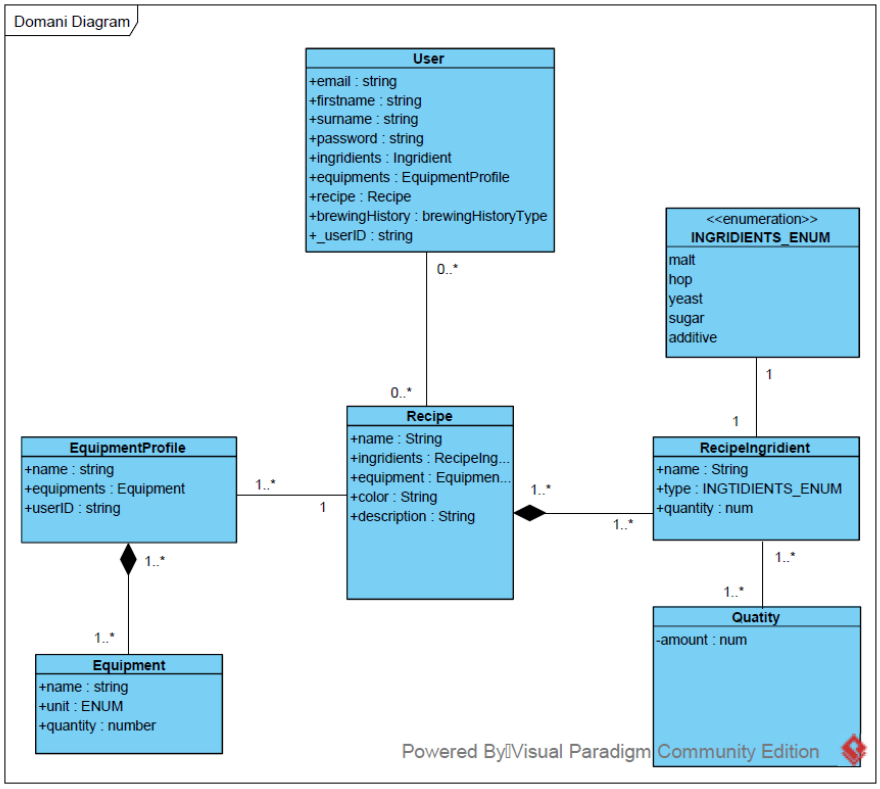

Figura 2: Diagramma delle classi a livello di dominio

3.1.2 Diagramma delle classi di progetto

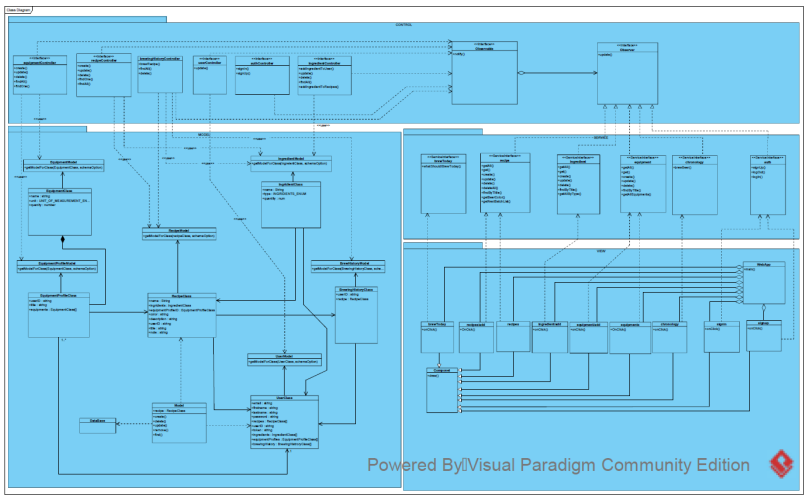

Figura 3: Diagramma delle classi a livello di progettazione

3.1.3 Diagrammi di sequenza di sistema

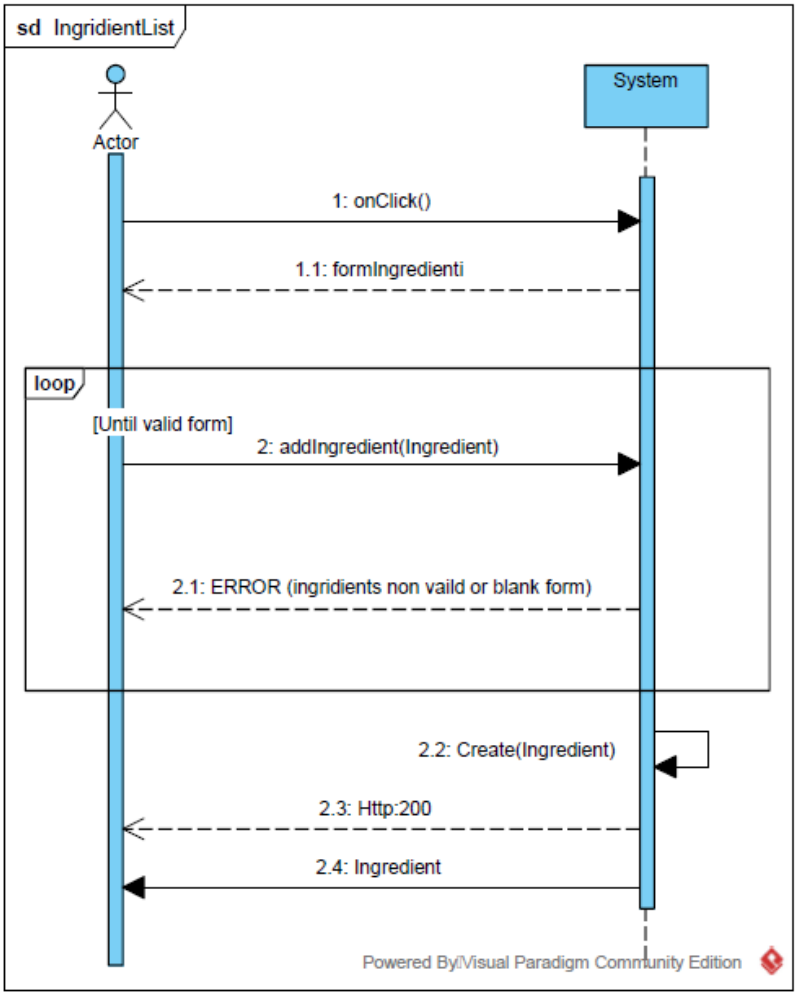

Figura 4: diagramma SSD

3.1.4 Diagrammi di sequenza

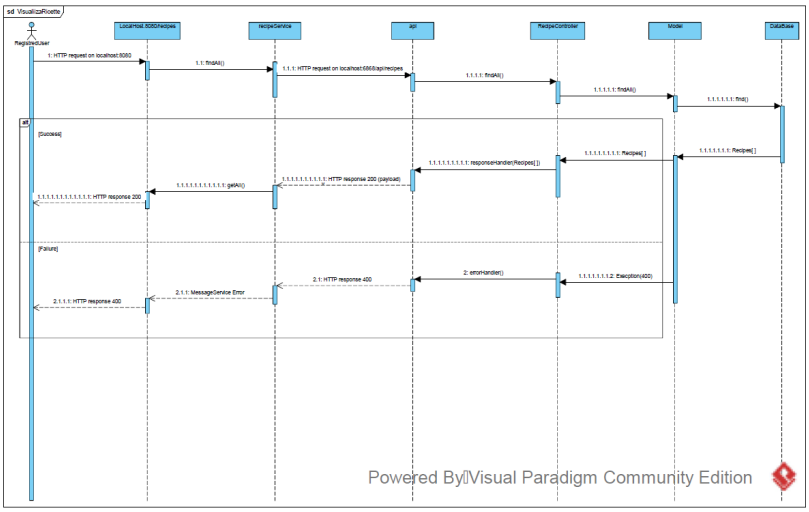

Figura 5: Diagramma di sequenza di progettazione

3.1.5 Diagramma degli stati

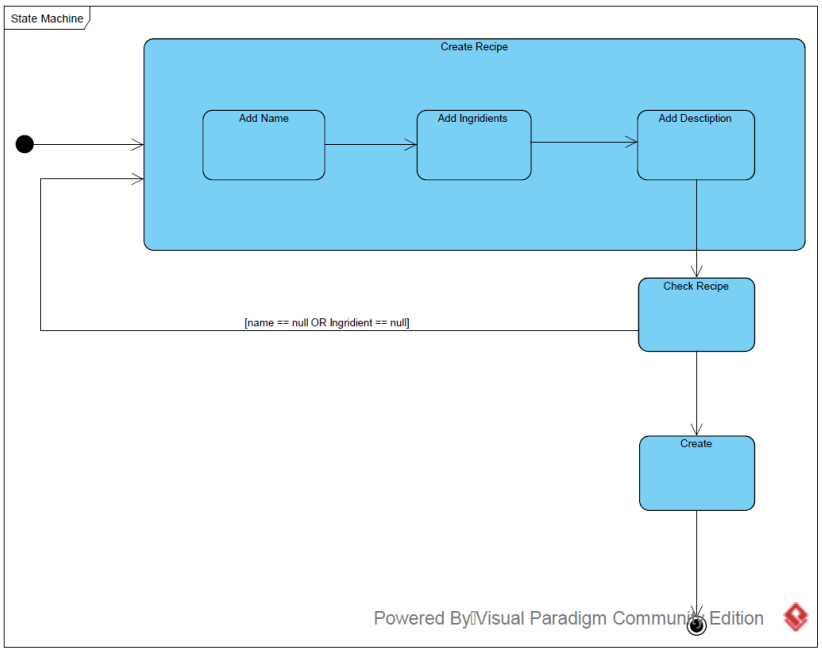

Figura 6: Diagramma degli stati

3.1.6 Diagramma di attività

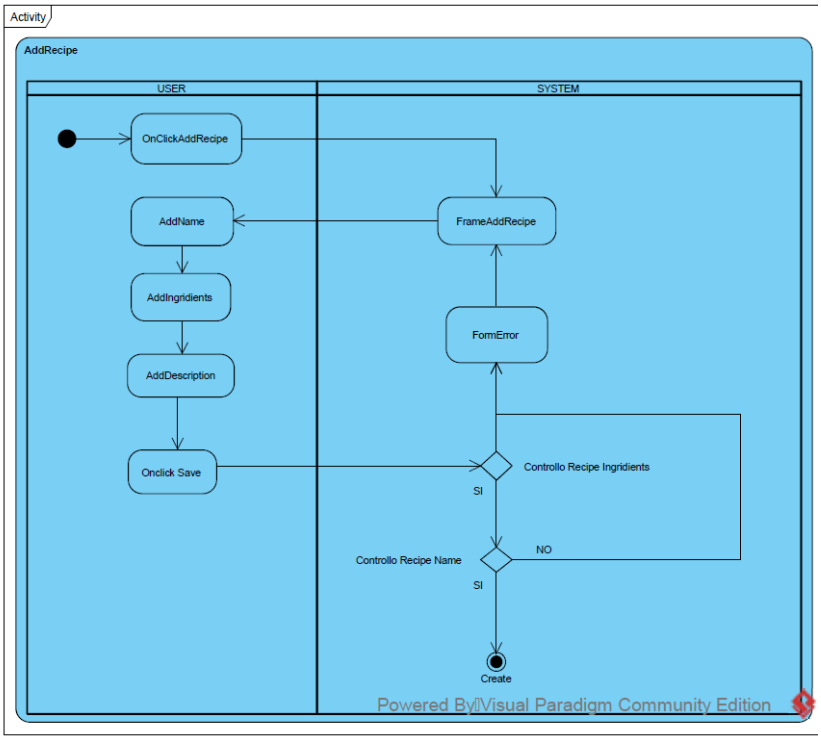

Figura 7: Diagramma di attività

3.1.7 Diagrammi della architettura software

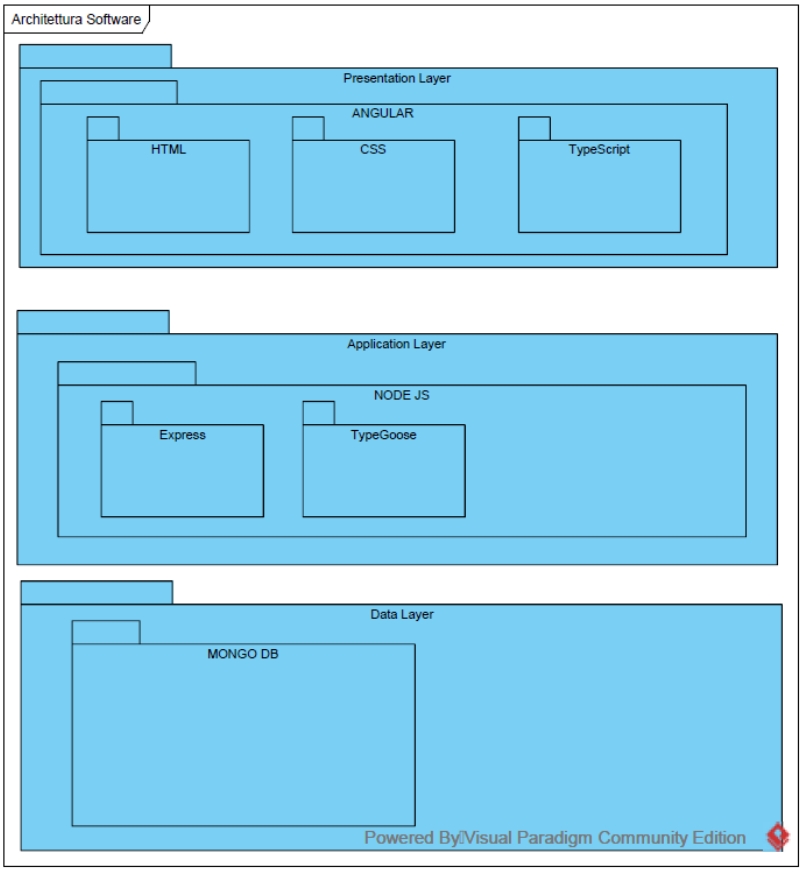

Figura 8: Diagrammi della architettura software

3.2 Tecnologie utilizzate

- Angular : è un framework open source per lo sviluppo di applicazioni web, utilizzato in questo progetto per lo sviluppo del forntend, l'approcio di angular è Model-View-ViewModel per ogni componente è presente un file .ts .html .css .spec, e un service per gestire il flusso di collegamento con il lato di backend.

- MongoDB : è un DBMS non relazionale, orientato ai documenti. Classificato come un database di tipo NoSQL, MongoDB si allontana dalla struttura tradizionale basata su tabelle dei database relazionali in favore di documenti in stile JSON con schema dinamico, rendendo l'integrazione di dati di alcuni tipi di applicazioni più facile e veloce, usato come database unico e principale per la web app.

- Node.js : Node.js è un ambiente runtime single-threaded, open-source e multi piattaforma per la creazione di applicazioni server-side e di networking veloci e scalabili. Gira sul motore di runtime JavaScript V8 e usa un'architettura I/O event-driven e non bloccante, il che lo rende efficiente e adatto ad applicazioni real-time. Abbiamo utilizzato il framework Express.js usato per creare l'api della web app così da rendere bidirezionali le operazioni di CRUD tra frontend e backend.

- GitHub : Git utlizzato usato per versionamento del codice utilizzando come server host GitHub.
- Sonarcloud : SonarCloud è un servizio online che rende possibile individuare Bugs, Code smell e vulnerabilità, svolgere testing, collegato direttamente con la nostra repository GitHub.
- Understand : Understand è stato utilizzato per individuare eventuali antipatter strutturali.

3.3 Design Principle

3.3.1 Principi SOLID

In informatica, l'acronimo SOLID si riferisce ai "primi cinque principi" dello sviluppo del software orientato agli oggetti descritti da Robert C. Martin.

- The single-responsibility principle : Afferma che ogni classe dovrebbe avere una ed una sola responsabilità, interamente incapsulata al suo interno,lato backend ogni ogni classe ha una e una sola responsabilità in quanto le classi servono solo a definire il modello dei dati che vengono incapsulati e passati ai vari model che servono a fare le operazioni sul database, mentre i controller gestiscono tutta la logica di business e i routes interfacciano i services al utente finale.
- The open–closed principle : Un'entità software dovrebbe essere aperta alle estensioni, ma chiusa alle modifiche,prendendo come esempio il fontend estendere una component consuma meno risorse che modificarla.
- The Liskov substitution principle : Gli oggetti dovrebbero poter essere sostituiti con dei loro sottotipi, senza alterare il comportamento del programma che li utilizza. Angular per come è stato strutturato rispetta questo principio.
- The interface segregation principle : Sarebbero preferibili più interfacce specifiche, che una singola generica, nel progetto sono state create interfacce di tipo ServiceInterface per il frontend che ognuna gestisce una componet mentre nel backend sono stati inseriti controller per gestire user,ingredient,recipe,equipaggiamento,autenticazione e la cronologia.
- The dependency inversion principle : Una classe dovrebbe dipendere dalle astrazioni, non da classi concrete. Angular per come è stato strutturato rispetta questo principio.

3.4 PHAME principles

- Encapsulation : Usata per nascondere l'implementazione, utilizzato nei controller che nascondono la logica di business.

3.5 Altri

Separation of concern : getBeerColor in quanto è la composizione di più metodi indipendenti tra di loro.

3.6 Architectural pattern

- Data source architectural patterns: Data mapper - usato per il trasferimento bidirezionale di dati tra la memoria della web app ed la memoria persistente, l'interfaccia mapper includerà i metodi CRUD,
- Security Patterns: Authentication - Usato per il SignIn e SignUp abbiamo implementato l'autenticazione dell'utente attraverso JWT (Json Web Token), è un sistema di cifratura dell'informazioni del utente per utilizzare le varie chiamate ai nostri servizi, nel token sono incapsulate le informazioni necessarie per garantire un utente (iduser,email), noi opzionalmente abbiamo reso il token è valido per un ora. Authorization - il sistema prevede che solo user registrati possano usare la webapp.

3.7 Design Pattern

- Controller: Fa da middleman tra l'interfaccia utente e i servizi ha il compito di gestire le operazioni di sistema, nella sezione di backend abbiamo definito controller di tipo use case, avendo instanziato un controller per l'autenticazione per le recipe, ingredient, equipment.
- Observer: Definisce una relazione 1 a N tra oggetto questo permette l'aggiornamento automatico degli oggetti, usato nel forntend per aggiornare dinamicamente gli oggetti Recipe,Equipment,Ingredient quando l'utente utilizza funzioni CRUD sugli oggetti sopra elencati.

3.8 Code Smell rilevati e risolti

Analizzando il progetto su SonarCloud, abbiamo individuato e risolto diversi tipi di code smell, che erano incentrati principalemnte su codice non utilizzato.

- Comments: rimossi tutti i commenti TODO e quelli che servivano a livello di sviluppo.
- Unused Import: rimossi tutte le librerie non usate.
- Empty Method: rimossi tutti i metodi vuoti.

3.9 Bugs rilevati e risolti

Analizzando il progetto su SonarCloud, abbiamo individuato e risolto diversi, che erano incentrati su errori lessicali

- Corretti alcuni ID mancanti in HTML che potevano dare problemi con il fato forntend del progetto.
- Abbiamo classificiato come falso positivo i bug riguradanti il TAILWIND CSS, in quanto è un problema di VS code e non influisce sulla corretta esecuzione del progetto
- UNICO VERO BUG

3.10 Understand

3.10.1 Backend

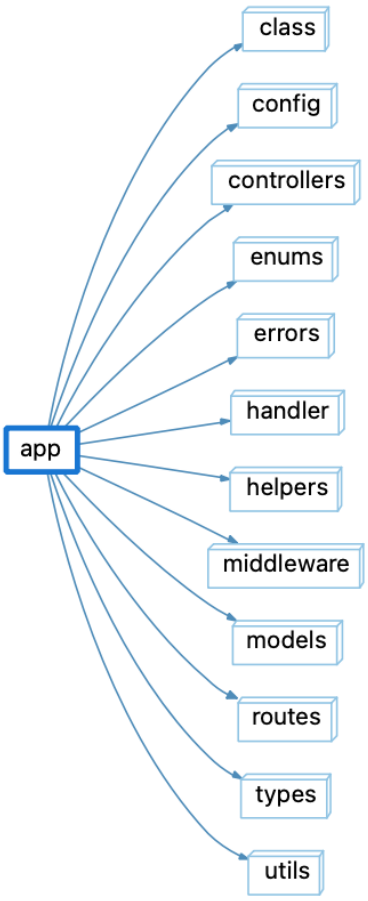

Figura 9: Grafo della architettura

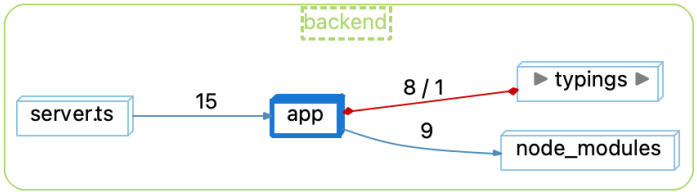

Figura 10: Grafo delle dipendenze

3.11 Frontend

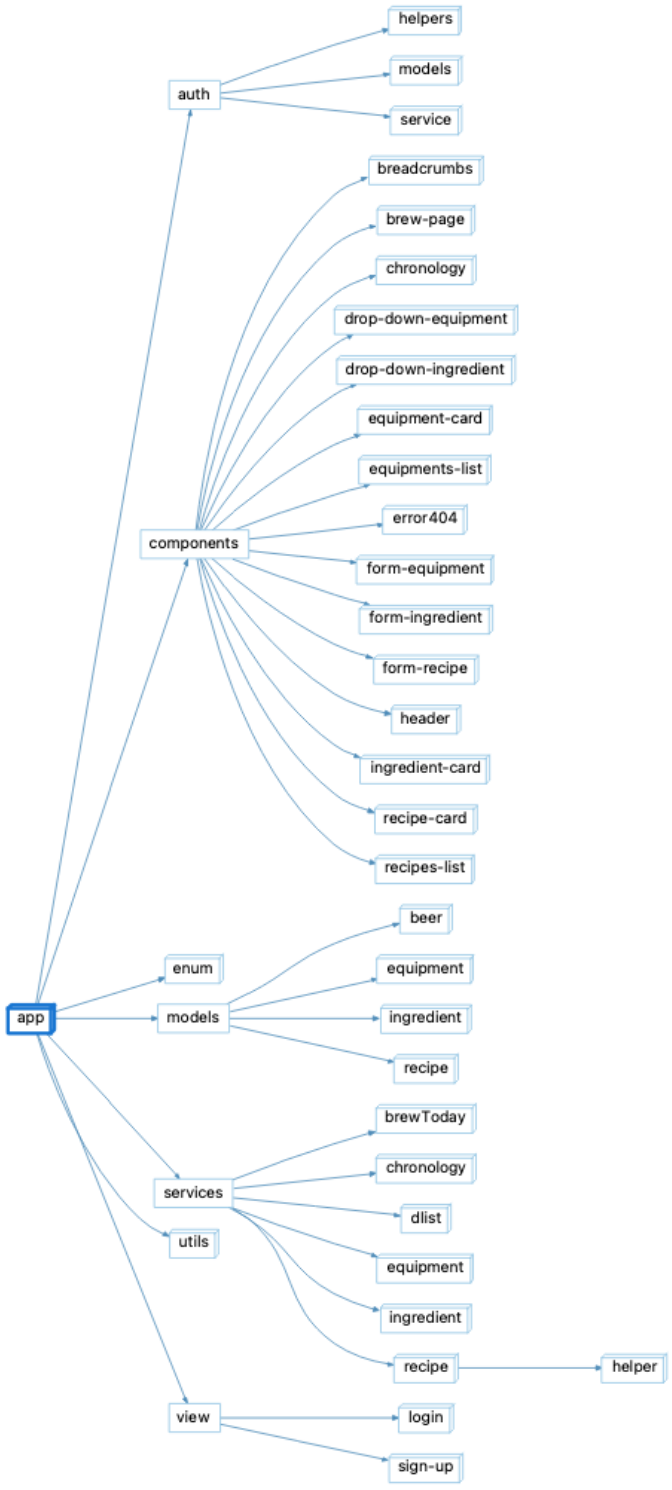

Figura 11: Grafo della architettura lato FE

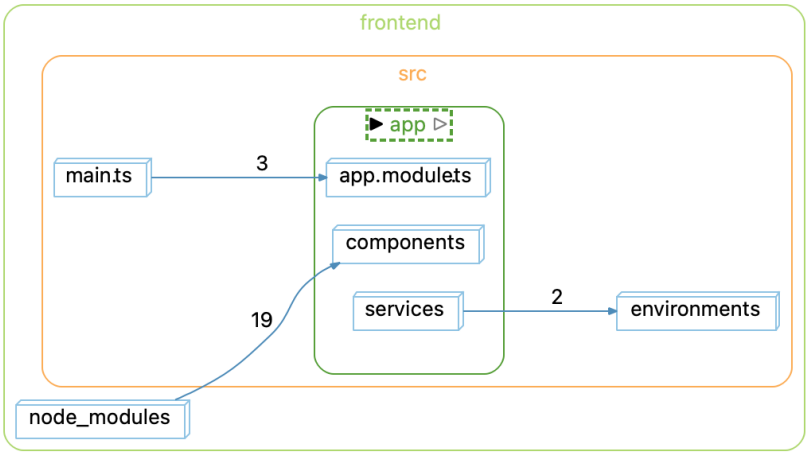

Figura 12: Grafo dipendenze lato FE

3.12 Problemi riscontrati durante la progettazione

Durante la fase di progettazione, abbiamo notato che gestivamo in maniere pesante il salvataggio di dati nel database, in quanto ogni user aveva tutte le sue instanze salvate dentro esso questo portava ad avere file json molti pesanti se si gestivano una quantità di dati elevata, abbiamo risolto il problema ristrutturando il backend e passando agli user le referenze di quello che avevano salvato dentro di esser per alleggerire il carico di dati su ogni file json. Per quanto riguarda il frontend ci siamo resi conti l'elevato numero di tipologie di birre producibili e di ingredienti possibili per rendere possibile per tenere intatto il principio di poter aggiungere all-grain beer, abbiamo impostato noi una serie di ingredienti predefiniti che contengono una elevata diversificazione per rendere possibile tenere il numero di birre salvabili molto elevato, stesso principio impostando ingredienti specifici per la tipologia di birra che targhetta la webapp, questa modifica ha reso possibile determinare direttamente il colore della birra impostando un algoritmo che associa ad ogni malto con una serie di operazioni uno di quaranta colori.

4 Sviluppi Futuri

Il softwere naturalmente è predisposto alla modifica ed estensione, nella visione degli sviluppatori esso può essere esteso da applicazione per home brewer a un social network dove sarà possibile caricare condividere ricette, con foto e descrizione con altri home brewer e la possibilità di essi di valutare la birra con un sistema di stelle, cosi da poter avere un contest con cadenza predefinita per eleggere la migliore birra. Per problemi di tempistiche non siamo riusciti ad implementare la fase di testing, ma verranno svolti durante il proseguimento dello sviluppo in quanto cerando siti simili abbiamo notato un buco nel mercato.