Smart Cuisine

GROUP MEMBERS:

Francesco Saverio Sconocchia Pisoni, 1889241
Giulio Caporro 2149534
Lorenzo Tanzi 2129131

github repository link: https://github.com/LorenzoT8942/SmartCuisine

January 9, 2025

1 Introduction

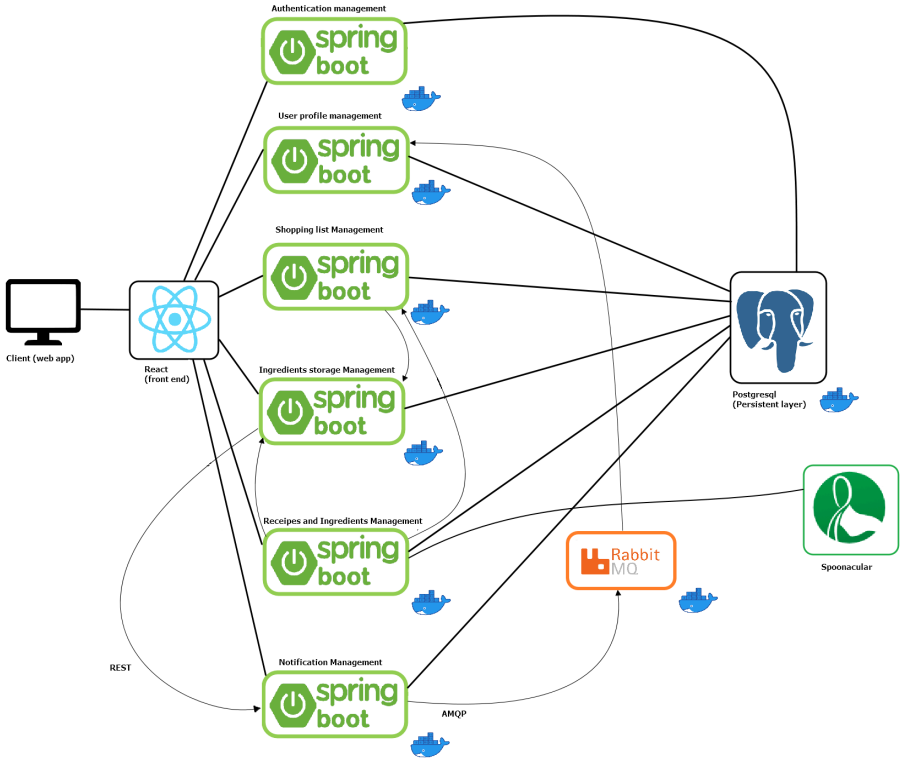
1.1 The idea of the application

Smart Cuisine aims to develop an application to discover new recipes and check their nutritional values in order to monitor the daily intake of macronutrients.
This application aims to improve the quality of the food intake and to give new ideas to people who want to eat in a healthier way, without giving up on tasty meals.
The application meets the needs of the users with dietary restrictions given by health issues, intolerances and religious or moral choices.
To achieve the previous purposes the system allows short-term and medium-term meal planning for the following days.
The type of users which could be interested about this application are personal trainers, people who can't afford a dietist or prefer to plan their meals by themselves.
Another possible user could be a person who uses the application to write down his shopping list and check the nutritional values of the items in the list.

2 Overview

2.1 Overview of the main components

First, is shown an image depicting the system architecture:



This is a microservices system based on Docker containers and is composed by the following components:

- RabbitMQ: It is a messaging broker that allows you to insert messages into queues. In the application it has been used to allow the Storage service to send notifications to the users, when an ingredient is expiring.

- Postgresql: It is the component that takes care of saving all the fundamental data for the application, it is a relational database based on SQL. It is used for example to save user's profile data, their saved recipes and notifications.

- Authentication Management: written in Java 21 using the framework Spring boot, it creates, provides and validates the JWT tokens for user's authentication.

- User Profile Management: written in Java 21 using the framework Spring boot, it handles user-related operations, including registration, login, profile management, and favorite recipes. It allows users to view and delete their profile data, as well as manage their favorite recipes and preferences.

- Shopping List Management: written in Java 21 using the framework Spring boot, it is responsible for managing user shopping lists. It provides functionalities for users to create, retrieve, and delete shopping lists, as well as add or remove ingredients from them. This container ensures that users can efficiently organize and manage their shopping needs, making it easier to keep track of what they want to buy.

- Ingredients Storage Management: written in Java 21 using the framework Spring boot, it is responsible for managing the storage of ingredients for users. It provides functionalities to add, retrieve, update, and delete ingredients in the user's storage. Users can set expiration dates, track ingredient quantities, and move ingredients from their shopping lists to their storage. This container ensures efficient inventory management for users, helping them keep track of their available ingredients.

- Recipes and Ingredients Management: written in Java 21 using the framework Spring boot, it is responsible for managing recipes and ingredients. It allows users to search for recipes by name or specific ingredients, as well as retrieve detailed information about individual ingredients. This container enables efficient management and retrieval of recipe and ingredient data, providing users with ideas for meals and allowing them to plan their cooking based on available ingredients.

- Notification Management: written in Java 21 using the framework Spring boot, it is responsible for managing user notifications. It allows the creation of notifications for users, providing timely alerts such as reminders about ingredient expiration. This container ensures that users stay informed and can act on important updates related to their stored ingredients.

Additionally, the Recipes and Ingredients Management service communicates with Spoonacular via API calls, to receive the recipes and ingredients data, which are the core of the service.

2.2 Docker Deployment

In order to realize the structure defined in the previous section, we wrote a docker-compose file that allows to launch all the necessary microservices. The file structure is defined as follows for each service:

• service name: Specifies the name of the service that needs to be configured.
• build: Allows to build a Docker image from a specified Dockerfile.
• volumes: Maps volumes from the local filesystem to the container, for persistence or to share files between the container and the host.
• environment: Defines environment variables for the container.
• ports: It is used to map the Docker container ports to the host ports
Here is an excerpt from the docker-compose.yml file:

```yaml
version: '3'

services:
  postgresql:
    container_name: postgres
    image: postgres:17
    restart: always
    environment:
      POSTGRES_USER: smart_cuisine
      POSTGRES_PASSWORD: forzaMagggicaRoma1
    volumes:
      - ./postgresql/data:/var/lib/postgresql/data
      - ./postgresql/postgresql.conf:/etc/postgresql/postgresql.conf
      - ./postgresql/init-database.sql:/docker-entrypoint-initdb.d/init-database.sql
    ports:
      - "5432:5432"
    command: -c 'config_file=/etc/postgresql/postgresql.conf'
    networks:
```

```
    - app-network
  healthcheck:
    test: ["CMD", "pg_isready", "-U", "smart_cuisine", "-d", "notification_db", "-h", "localhost"]
    interval: 30s
    retries: 3
    start_period: 10s
    timeout: 5s


rabbitmq:
  container_name: rabbitmq
  image: rabbitmq:3-management
  ports:
    - "5672:5672"
    - "15672:15672"
  volumes:
    - ./rabbitmq/rabbitmq_data:/var/lib/rabbitmq
    - ./rabbitmq/rabbitmq.conf:/etc/rabbitmq/rabbitmq.conf
    - ./rabbitmq/definitions.json:/etc/rabbitmq/definitions.json
  networks:
    - rabbit-network




authentication-management:
  container_name: authentication-management
  image: authentication-management
  build:
      context: ./authenticationManagement
  ports:
    - "3000:3000"
  networks:
    - app-network



userprofile-management:
  container_name: userprofile-management
  image: userprofile-management
  build:
```

```
      context: ./userProfileManagement
  ports:
    - "3001:3000"
  networks:
    - rabbit-network
    - app-network
  depends_on:
    postgresql:
      condition: service_healthy
    rabbitmq:
      condition: service_started

shoppinglist-management:
  container_name: shoppinglist-management
  image: shoppinglist-management
  build:
    context: ./shoppingListManagement
  ports:
    - "3002:3000"
  networks:
    - app-network
  depends_on:
    postgresql:
      condition: service_healthy

notification-management:
  container_name: notification-management
  image: notification-management
  build:
    context: ./notificationManagement
  ports:
    - "3003:3000"
  networks:
    - rabbit-network
    - app-network
  depends_on:
    postgresql:
      condition: service_healthy
    rabbitmq:
      condition: service_started
```

```yaml
  recipesingredients-management:
    container_name: recipesingredients-management
    image: recipesingredients-management
    build:
      context: ./recipesAndIngredientManagement
    ports:
      - "3004:3000"
    networks:
      - app-network
    depends_on:
      postgresql:
        condition: service_healthy

  storage-management:
    container_name: storage-management
    image: storage-management
    build:
      context: ./storageManagement
    ports:
      - "3005:3000"
    networks:
      - app-network
    depends_on:
      postgresql:
        condition: service_healthy

  react-app:
    build:
      context: .
      dockerfile: Dockerfile
    ports:
      - "3006:3006"
    restart: always

volumes:
  rabbitmq_data:
    driver: local

networks:
  rabbit-network:
    driver: bridge
```

```
app-network:
  driver: bridge
```

3. Design Techniques

3.1. User Stories

The User Stories are short descriptions of software functionalities from the end-user's perspective. They are used in the context of agile development to capture requirements in a simple and understandable way, without going into technical details.
The structure is composed as follows:
As [type of user] I want to [action] so that [goal].
What follows is an image containing the user stories of the application:

EPIC	USER STORY	PRIORITY	VALUE	RISK	ESTIMATE	RELEASE	ACCEPTANCE	VALUE/RISK MATRIX
SignUp	As a not registered user, I want to subscribe with email and password so that being a registered user	1 High	1 High	2 Med	MD	1.0	1) the user can access a signup form and fill it 2) the process validates the data 3) the process redirects the user to login page after successful registration	Strategic
LogIn	As a registered user, I want to log in with email and password so that logging in my account	1 High	1 High	3 Med	MD	1.0	1) the user can access a login form 2) the process must authenticate the user's credentials 3) the process redirects the user to the homepage after a successful login	Leveraged
Profile managing	As a registered user, I want to access my profile so that I can visualize my personal data	4 Low	4 Low	3 Med	MD	1.0	1) the user can access its profile form 2) the process must display the user's profile data	Routine
Profile managing	As a registered user, I want to delete my account so that being no more a registered user	4 Low	4 Low	4 Low	SM	1.0	1) the user can access its profile form 2) the process must display the user's profile data 3) the user can click on the delete button 4) the process must delete the account's data from its DB	Routine
Profile managing	As a registered user, I want to modify my email address so that I can update it	4 Low	4 Low	3 Med	SM	1.0	1) the user can access its profile form 2) the process must display the user's profile data 3) the user can modify the email field and save 4) the process must modify the email entry from its DB	Routine
Profile managing	As a registered user, I want to modify my password so that I can update it	4 Low	4 Low	3 Med	SM	1.0	1) the user can access its profile form 2) the process must display the user's profile data 3) the user can modify the password field and save 4) the process must modify the password entry from its DB	Routine
Profile managing	As a registered user, I want to modify my gender so that I can update it	4 Low	4 Low	3 Med	SM	1.0	1) the user can access its profile form 2) the process must display the user's profile data 3) the user can modify the gender field and save 4) the process must modify the gender entry from its DB	Routine
Storage Management	As a registered user, I want to add my ingredients to the storage so that i can keep track of the ingredients I have at home	1 High	1 High	4 Low	SM	3.0	1) the user can search the ingredient by name 2) the process must display the ingredients matching with the name 3) the process must add an entry for the storage in the DB	Leveraged

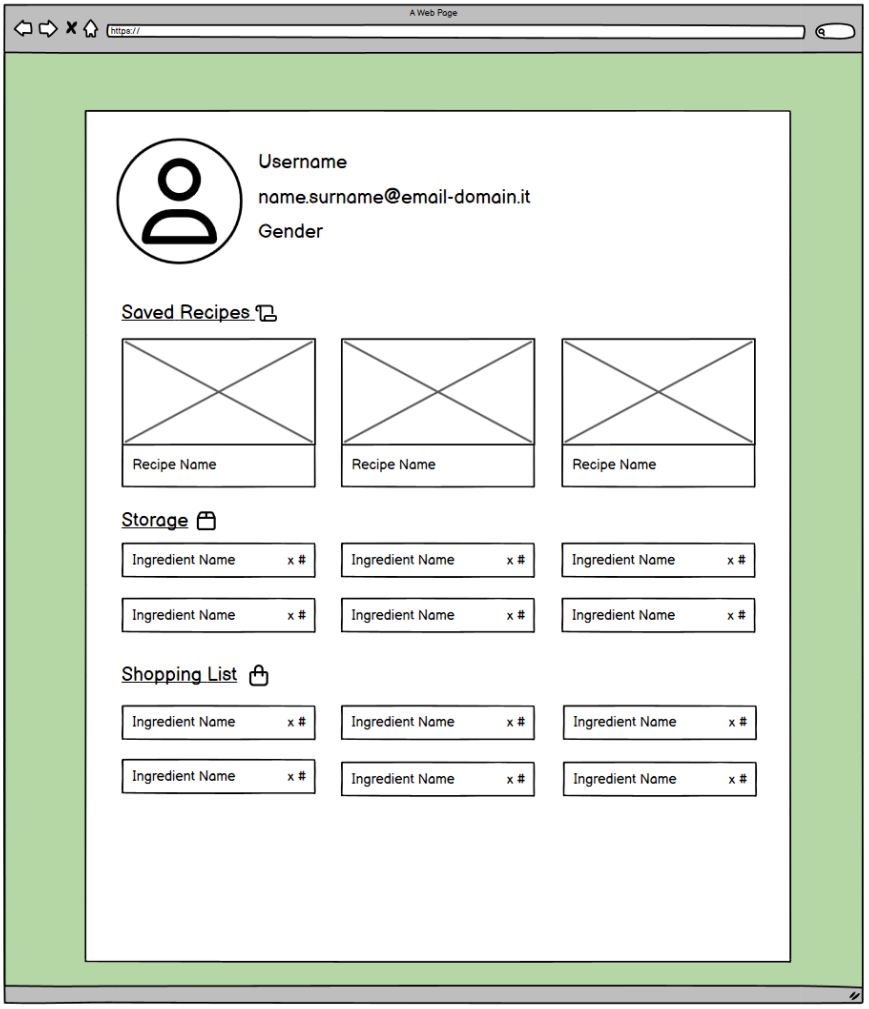
Storage Management	As a registered user, I want to delete an ingredient from my storage, so that I can delete an ingredient I used for a recipe.	1 High	1 High	4 Low	SM	3.0	1) the user can select the ingredient from the storage list of ingredients 2) the user can select multiple ingredients at the same time 3) the process must delete the entries of the DB related to the selected ingredients	Leveraged
Recipes	As a registered user, I want to search for recipes so that I can find ideas for meals	1 High	1 High	3 Med	MD	1.x	1) the user can type the name of a recipe in a search bar 2) the process must display the resulting recipes 3) the user can select a recipe from the search result 4) the process must display the page of the recipe	Leveraged
Recipes	As a registered user, I want to save recipes so that i can access them easily in the future	2 Med	2 Med	4 Low	SM	1.x	1) the user can access the recipe's page 2) the user can click a button to save the recipe as "saved" 3) the process must add an entry in the DB to link the user with the saved recipe. 4) the process must display a visual feedback to inform the user that the recipe is saved.	Leveraged
Recipes	As a registered user, I want to manage the ingredients of a recipe to my shopping list, so that I can keep track of what I need to buy	2 Med	1 High	4 Low	SM	2.0	1) the user can access the page of a receipe 2) the user can decide to "add" the receipe's ingredients in its shopping list 3) the process must add an entry in the shoping list DB table to link the user with each saved ingredients and its quantity	Leveraged
Recipes	As a registered user, I want to delete a saved recipe so that I can remove it if I don't like it	2 Med	1 High	4 Low	SM	1.x	1) the user can access the saved recipes' page. 2) the user can click on a button to remove the recipe from the "saved" recipes. 3) the process removes the entry for the saved recipe from the DB. 4) the process displays a visual feedback to inform the user of the successful removal.	Leveraged
Recipes	As a registered user, I want to search recipes containing a specific ingredient, so that I can find something I can cook with the ingredients I have in storage.	1 High	1 High	3 Med	MD	1.x	1) the user can access the "recipe search" page 2) the user can select the search mode "by ingredient". 3) the user can type the ingredient name 4) the process displays the resulting recipes containing that ingredient.	Leveraged
Shopping List	As a registered user, I want to move the ingredients from my shopping list to my ingredients storage, so that I can keep track of what I bought.	2 Med	1 High	4 Low	LG	2.0	1) the user can access the shopping list page 2) the user can select some ingredients and set the quantities for moving them in the ingredients storage and confirm 3) the process must for each ingredient update the quantity in the corresponding shopping list table (delete if quantity goes to 0) 4) the	Leveraged

Shopping List	As a registered user, I want to delete an ingredient from my shopping list, so that I can remove it if I already have it.	2 Med	1 High	4 Low	MD	2.x	1) the user can access the "shopping list" page 2) the user can select multiple ingredients from the shopping list 3) the user can delete the selected ingredient from the shopping list	Leveraged
Shopping List	As a registered user, I want to create a shopping list, so that I can keep track of what I want to buy.	3 Med	2 Med	4 Low	MD	2.x	1) The user can access the "shopping list" page 2) The user can create a new shopping list and name it.	Leveraged
Shopping List	As a registered user, I want to delete a shopping list, so that I can keep track of what I want to buy.	4 Low	2 Med	4 Low	SM	2.x	1) The user can access the "shopping list" page 2) The user can select and delete an exiisting shopping list.	Leveraged
Shopping List	As a registered user, I want to read my shopping lists, so that I know what to buy.	2 Med	1 High	4 Low	SM	2.x	1) The user can access the "shopping list" page 2) The user can view all existing shopping lists, including details of the items in each list.	Leveraged
Ingredient Storage	As a registered user, I want to set the expiration date for the ingredient in my storage, so that I can remember when they will expire.	4 Low	3 Med	4 Low	SM	2.x	1) the user can access the "ingredient's storage" page 2) the user can select an ingredient and set the expiration date 3) the process must set/update the expiration date value of the ingredient in the DB	Routine
Ingredient Storage	As a registered user, I want to receive notifications 1 day before my ingredients expire, so that I have time to cook them before the expiration date.	2 Med	2 Med	1 High	LG	3.0	1) the process elaborates the notification 2) the process displays the notification	Strategic
Ingredient Storage	As a registered user, I want to set the quantity for the ingredient in my storage, so that I can know the quantities of each ingredient in the storage.	4 Low	3 Med	4 Low	SM	3.x	1) the user can access the "ingredient's storage" page 2) the user can select an ingredient and set the quantity 3) the process must set/update the quantity value of the ingredient	Routine
Ingredient Storage	As a registered user, I want to check my storage, so that I know what I have at home.	1 High	1 High	4 Low	MD	3.x	1) The user can access the "ingredient storage" page 2) The user can view a list of ingredients available in their storage.	Leveraged

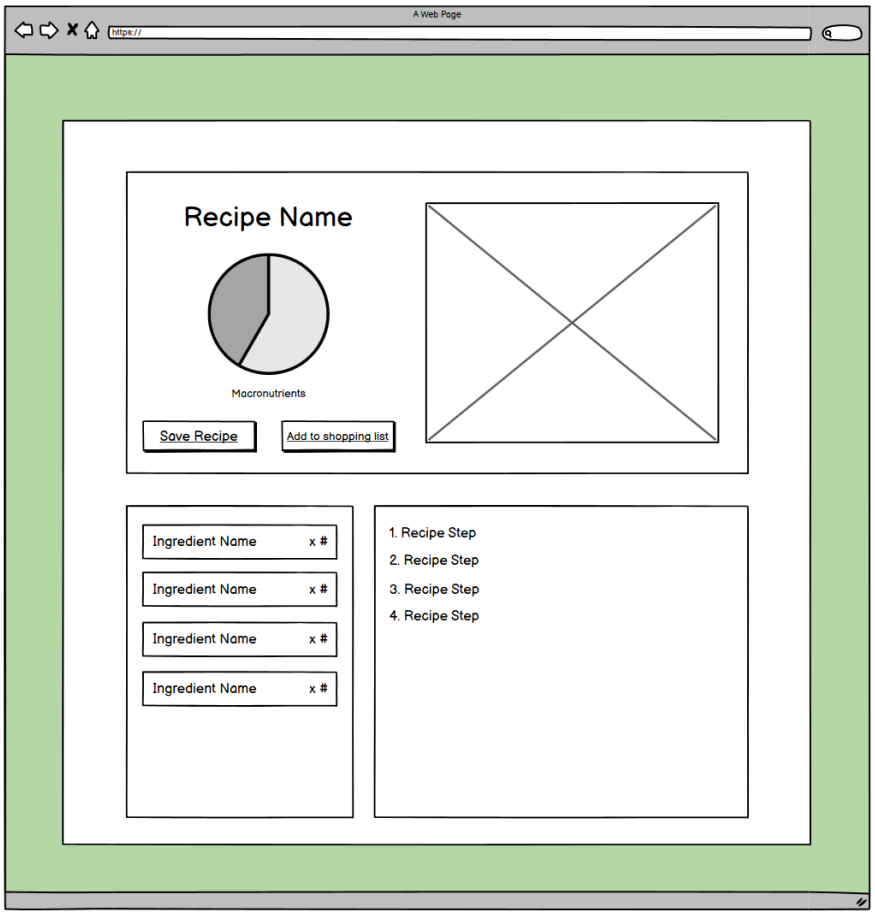
3.2 Mockups

Mockups are graphical representations of the application used to design and visualize the user interface (UI) and user experience (UX) before starting development.
They show how the interface elements, such as buttons, menus, and windows, will be arranged and help to identify and to resolve design or usability issues before the actual coding. In this section, some of the mockups used as the first prototype of the application are shown. The mockups are realized using Balsamiq software.

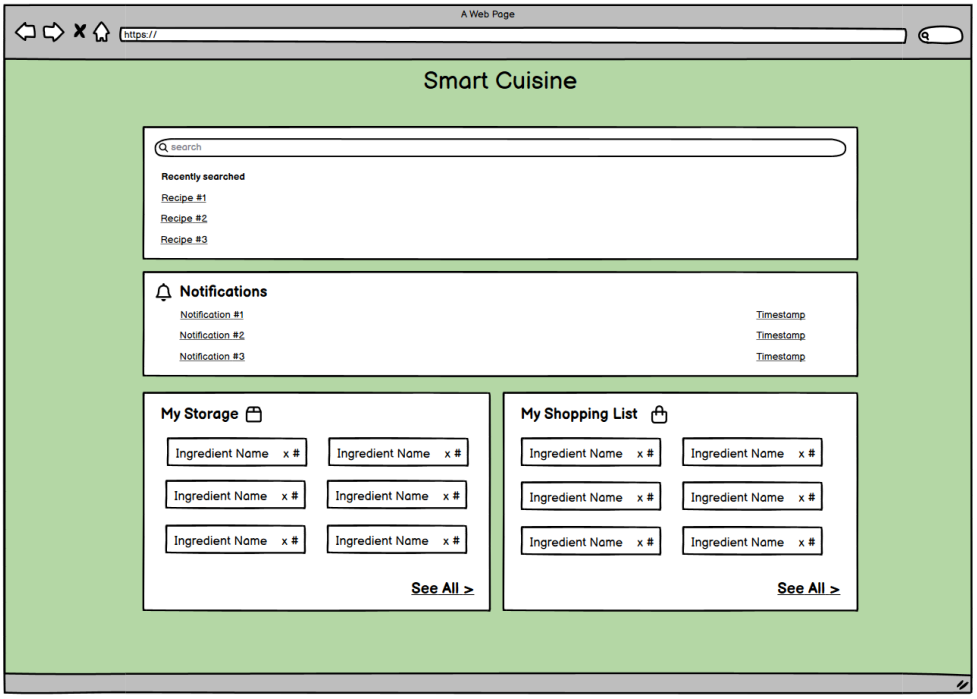
Profile page



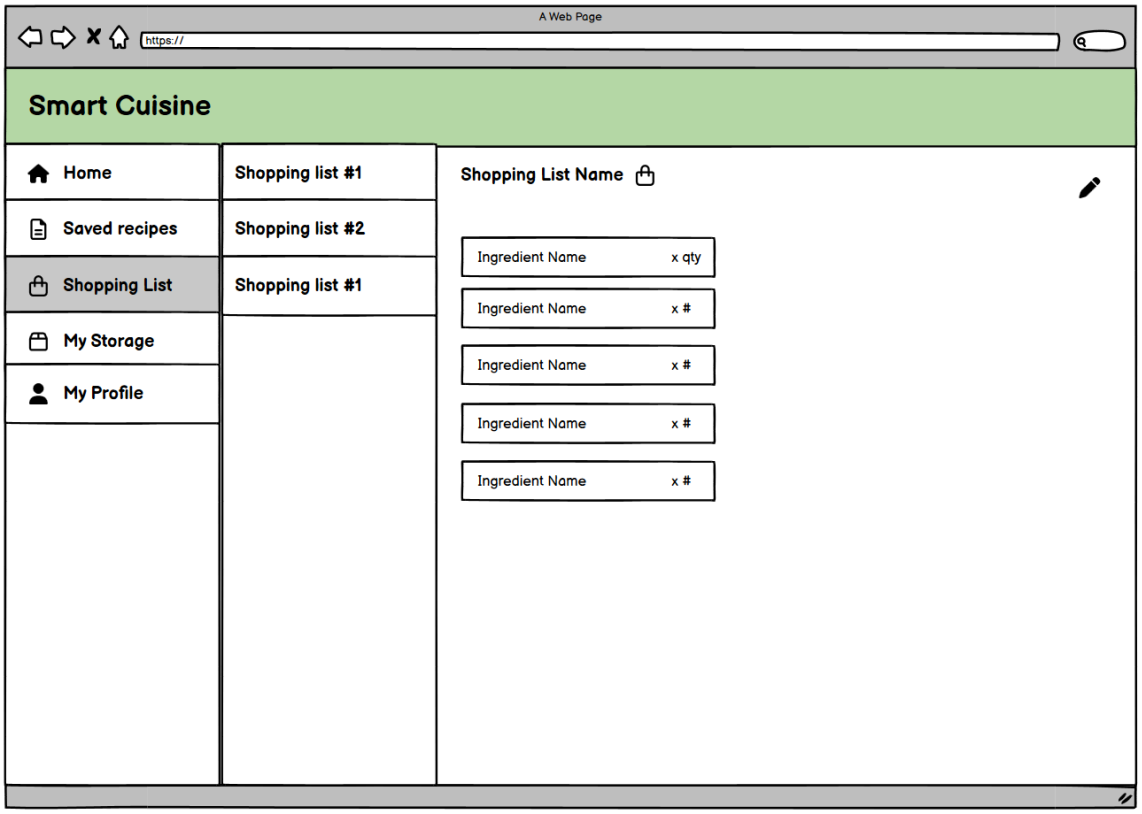
Recipe details view



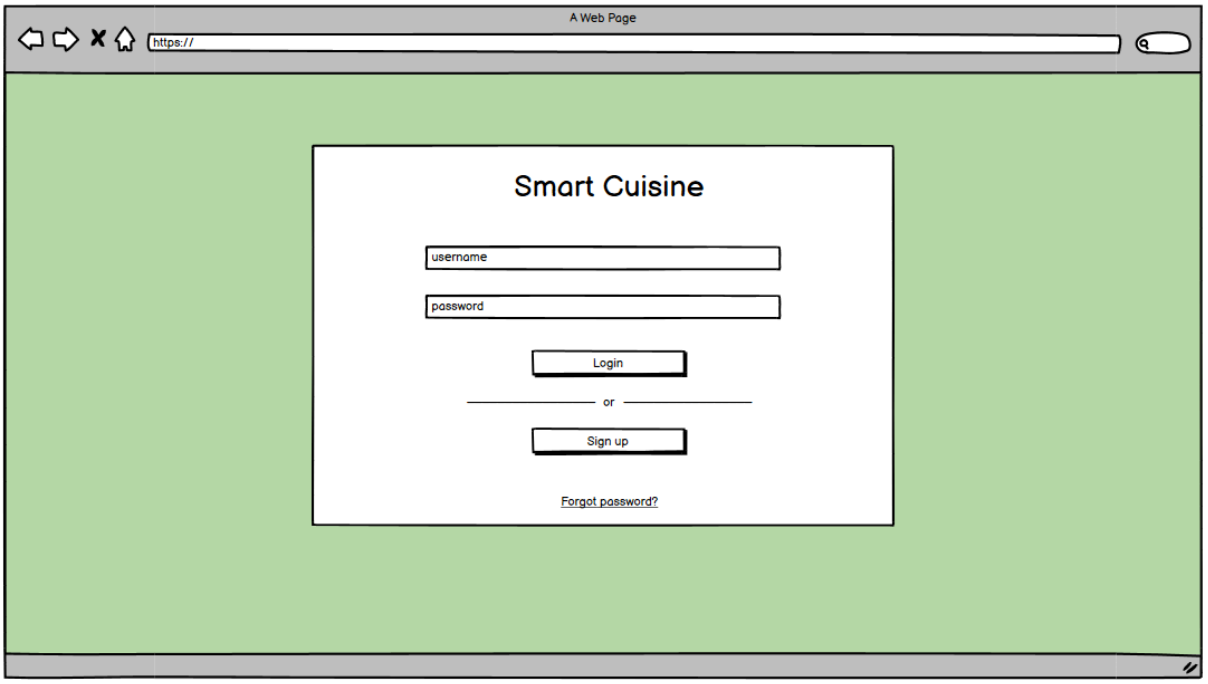
Home view



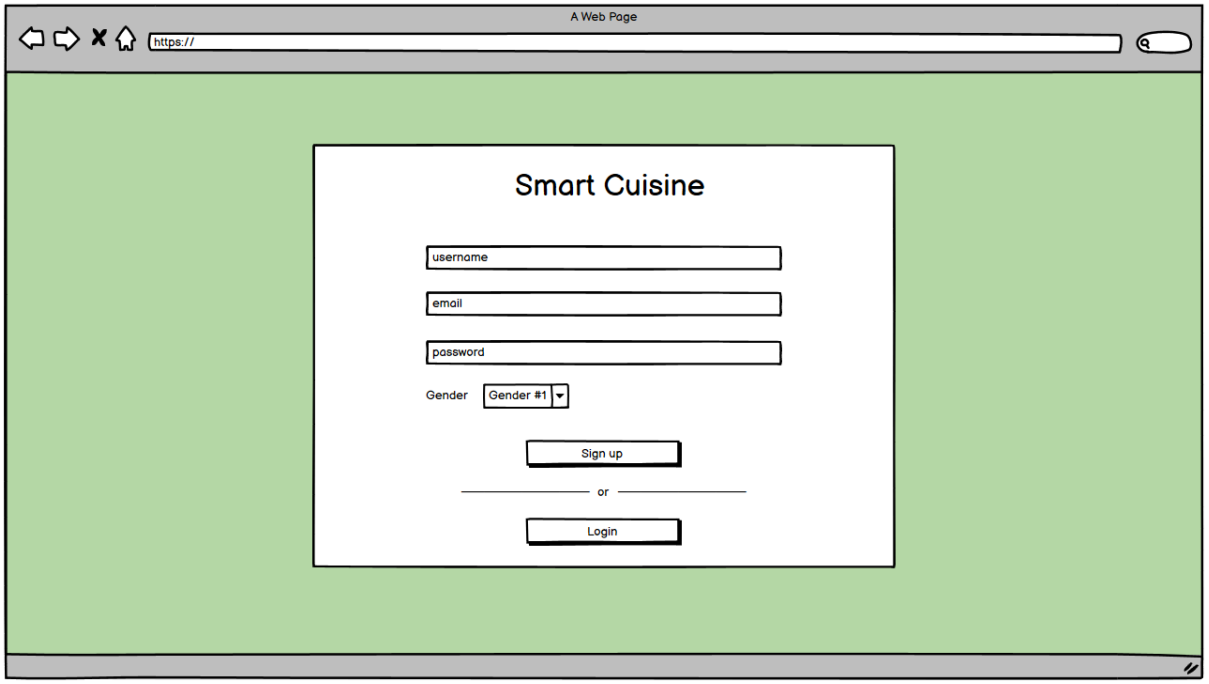
Shopping list view



Login view

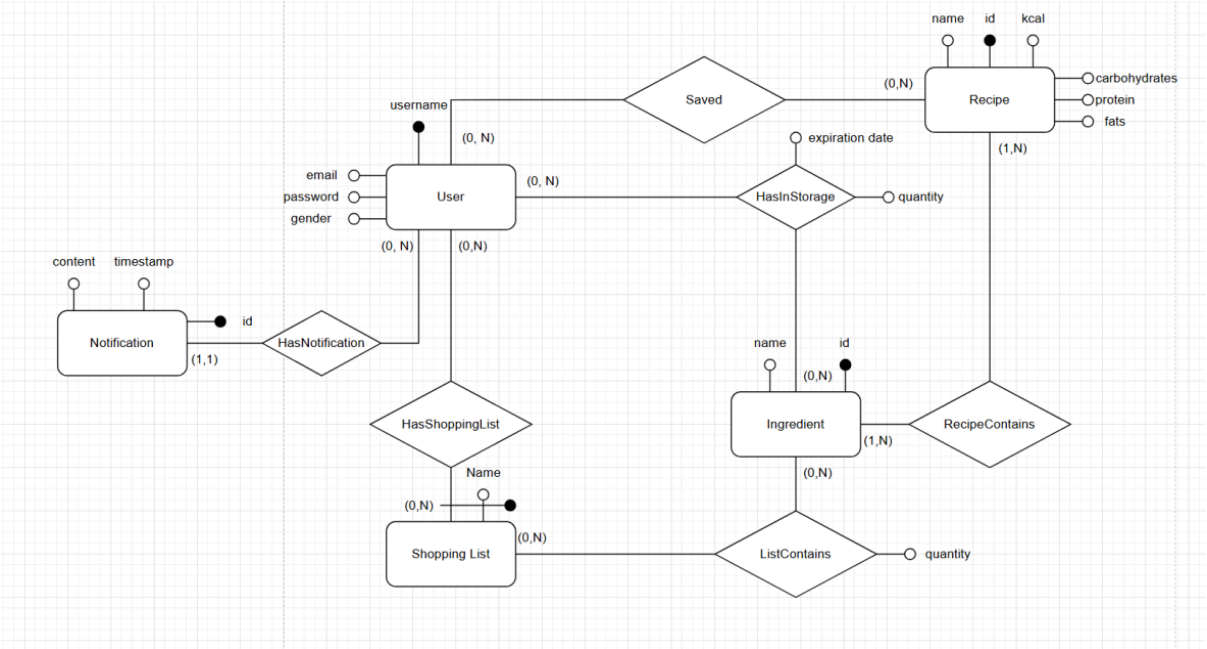


Sign Up view


A Web Page
https://
Smart Cuisine
username
email
password
Gender
Gender #1
Sign up
or
Login

3.3 Database

This section shows the ER diagram used for the application database design.



3.4 Function Points

Function Points (FP) are a standardized measure used to estimate the size and complexity of a software system based on its functionalities. Function Points quantify the work required to develop a software system by evaluating the functionalities it offers to the end user, rather than the code used to implement it.
The main components are:

- External Inputs (EI): Data flows or commands that the system receives from the outside (inputs).

- External Outputs (EO): Results produced by the system and sent to the outside, such as reports or messages.

- External Inquiries (EQ): Requests that the system processes to provide data to the user.

- Internal Logical Files (ILF): Data that the system stores and uses internally.

- External Interface Files (EIF): Files managed by other systems that are used by the software. In this project there are not EIFs.

ILF

	USER	SHOPPINGLIST	NOTIFICATION
RET	1	1	1
DET	4	2	3
RET/DET	Low(7)	Low(5)	Low(7)

IEF

	RECIPE	INGREDIENT		
RET	1	1		
DET	6	2		
RET/DET	Low(7)	Low(5)	FP	31

EI

USER

	INSERT	DELETE	UPDATE		
FTR	1	1	1		
DET	4	1	3		
FTR/DET	Low(3)	Low(3)	Low(3)	FP	9

SHOPPING LIST

	INSERT	DELETE	UPDATE		
FTR	2	1	2		
DET	2	1	2		
FTR/DET	Low(3)	Low(3)	Low(3)	FP	9

NOTIFICATION

	INSERT	DELETE	UPDATE		
FTR	1	1	1		
DET	3	1	3		
FTR/DET	Low(3)	Low(3)	Low(3)	FP	9

EO

	VIEW SAVED RECIEPES	VIEW SHOPPING LIST	VIEW NOTIFICATION			
FTR	2	2	1			
DET	7	4	3			
FTR/DET	Medium(5)	Low(4)	Low(4)		FP	13

EQ

	SEARCH FOR RECIPES	SEARCH FOR INGREDIENTS			
FTR	1	1			
DET	5	2			
FTR/DET	Low(3)	Low(3)		FP	6

FP	77
LOC	2310

The total amount of Function Points (FP) is 77, this result is obtained by summing up the single FPs of each component described above. Once computed the Function Points, we used a table to convert them into Line of Code (LOC).

3.5 Cocomo II

COCOMO II (COnstructive COst MOdel II) is a mathematical model used to estimate the costs, the time and the resources required to develop the software product.

Software Development (Elaboration and Construction)

Effort = 3.5 Person-months
Schedule = 4.4 Months
Cost = $5291

Total Equivalent Size = 3021 SLOC
Effort Adjustment Factor (EAF) = 0.37

Acquisition Phase Distribution

Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	0.2	0.6	0.4	$317
Elaboration	0.8	1.7	0.5	$1270
Construction	2.7	2.8	1.0	$4021
Transition	0.4	0.6	0.8	$635

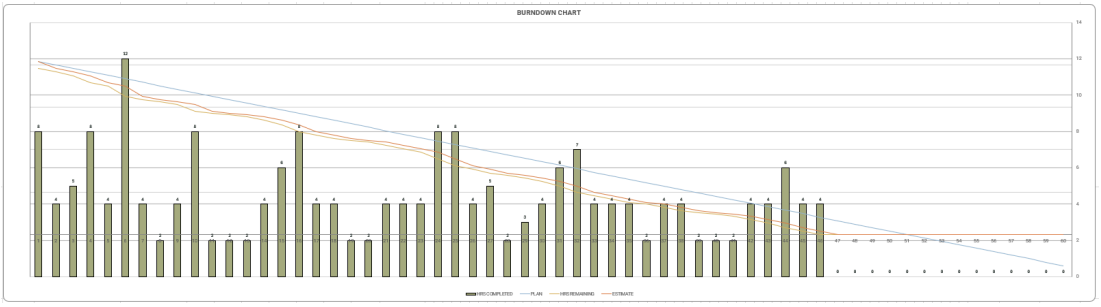
Software Effort Distribution for RUP/MBASE (Person-Months)

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.0	0.1	0.3	0.1
Environment/CM	0.0	0.1	0.1	0.0
Requirements	0.1	0.2	0.2	0.0
Design	0.0	0.3	0.4	0.0
Implementation	0.0	0.1	0.9	0.1
Assessment	0.0	0.1	0.6	0.1
Deployment	0.0	0.0	0.1	0.1

3.6 Scrum

SCRUM is an agile framework that relies on a set of principles and practices to manage complex projects. It is designed to be flexible and adaptable, allowing teams to respond quickly to changes and continuously improve. We worked on 2-week sprints. Each user story was divided into multiple tasks, thus defining the backlog. For each sprint, we assigned a certain number of tasks, based on the estimated number of hours needed to complete each of them. We got 3 sprints, for a total of 1.5 months of estimated work. Below is an extract of the backlog and also an extract of the Gantt chart.

WORK BREAKDOWN	TASK TITLE	TASK OWNER	AMOUNT OF WORK IN HOURS			SPRINT	START DATE	DUE DATE	DURATION	PCT OF TASK COMPLETE
			ESTIMATE	COMPLETED	REMAINING					
1	Authentication service	Francesco Saverio Sconocchia	10	10	0					100%
1.1	DB Creation	Francesco Saverio Sconocchia	1	1	0	1	08/11/2024	09/11/2024	2	100%
1.2	Environment Creation	Francesco Saverio Sconocchia	1	1	0	1	08/11/2024	08/11/2024	1	100%
1.3	JWT setting	Francesco Saverio Sconocchia	2	2	0	1	09/11/2024	09/11/2024	1	100%
1.4	JWT creation and providing	Francesco Saverio Sconocchia	4	4	0	1	08/11/2024	09/11/2024	2	100%
1.5	Tests auth	Francesco Saverio Sconocchia	2	2	0	1	09/11/2024	09/11/2024	1	100%
2	User Profile service	Francesco Saverio Sconocchia	14	14	0					100%
2,1	SignUp form creation	Francesco Saverio Sconocchia	2	2	0	2	05/12/2024	06/12/2024	2	100%
2,2	SignUp Data Validation	Francesco Saverio Sconocchia	1	1	0	1	09/11/2024	09/11/2024	1	100%
2,3	SignIn form creation	Francesco Saverio Sconocchia	2	2	0	2	05/12/2024	06/12/2024	2	100%
2,4	SignIn Data Validation	Francesco Saverio Sconocchia	1	1	0	1	09/11/2024	09/11/2024	1	100%
2,5	User's data insertion	Francesco Saverio Sconocchia	1	1	0	1	09/11/2024	12/11/2024	4	100%
2,6	Profile form creation	Francesco Saverio Sconocchia	2	2	0	2	05/12/2024	06/12/2024	2	100%
2,7	Delete profile	Francesco Saverio Sconocchia	1	1	0	1	17/11/2024	17/11/2024	1	100%
2,8	Update profile	Francesco Saverio Sconocchia	4	4	0	1	18/11/2024	20/11/2024	3	100%
3	RabbitMQ Service	Francesco Saverio Sconocchia	9	9	0					100%
3,1	RabbitMQ service	Francesco Saverio Sconocchia	4	4	0	1	12/11/2024	13/11/2024	2	100%
3,2	Message consumer	Francesco Saverio Sconocchia	4	4	0	1	13/11/2024	13/11/2024	1	100%
3,3	Message insertion	Francesco Saverio Sconocchia	1	1	0	1	13/11/2024	13/11/2024	1	100%
4	Notification Service	Francesco Saverio Sconocchia	4	4	0					100%
4,1	Notification service	Francesco Saverio Sconocchia	2	2	0	1	14/11/2024	14/11/2024	1	100%
4,2	Filters	Francesco Saverio Sconocchia	1	1	0	1	14/11/2024	14/11/2024	1	100%
4,3	Notification implementation	Francesco Saverio Sconocchia	1	1	0	1	14/11/2024	15/11/2024	2	100%
5	Shopping List Service	Giulio Caporro	32	32	0					100%
5,1	Create Shopping List	Giulio Caporro	6	6	0	1	08/11/2024	09/11/2024	2	100%
5,2	Update Shopping List	Giulio Caporro	5	5	0	1	08/11/2024	10/11/2024	3	100%
5,3	Delete Shopping List	Giulio Caporro	4	4	0	1	11/11/2024	13/11/2024	3	100%
5,4	Add Quantity Selection	Giulio Caporro	3	3	0	1	12/11/2024	13/11/2024	2	100%
5,5	Searching shoppingList	Giulio Caporro	3	3	0	1	13/11/2024	14/11/2024	2	100%
5,6	Create Delete Option in Shopping List	Giulio Caporro	5	5	0	1	14/11/2024	15/11/2024	2	100%
5,7	Implement Ingredient Addition	Giulio Caporro	6	6	0	1	15/11/2024	16/11/2024	2	100%
6	Storage Service	Giulio Caporro	80	80	0					100%
6,1	Create Storage Model	Giulio Caporro	6	6	0	2	01/12/2024	01/12/2024	1	100%
6,2	Implement CRUD operations	Giulio Caporro	8	8	0	2	02/12/2024	02/12/2024	1	100%
6,3	JWT Integration for Security	Giulio Caporro	7	7	0	2	03/12/2024	03/12/2024	1	100%
6,4	Create Scheduled Task Service	Giulio Caporro	7	7	0	2	03/12/2024	05/12/2024	3	100%
6,5	Notification Integration	Giulio Caporro	8	8	0	2	06/12/2024	11/12/2024	6	100%
6,6	Add Quantity Selection for Storage	Giulio Caporro	2	2	0	2	07/12/2024	08/12/2024	2	100%
6.7	Enable Notifications for Expirations	Giulio Caporro	25	25	0	2	12/12/2024	13/12/2024	2	100%
6.8	Display Ingredient Storage	Giulio Caporro	5	5	0	2	13/12/2024	15/12/2024	3	100%
6.9	Implement UI	Lorenzo Tanzi	12	12	0	3	18/12/2024	09/01/2025	23	100%
7	Recipes&Ingredients Service	Lorenzo Tanzi	55	55	0					100%
7.1	Home page view	Lorenzo Tanzi	10	10	0	3	18/12/2024	19/12/2024	2	100%
7.2	Search view	Lorenzo Tanzi	7	7	0	3	18/12/2024	19/12/2024	2	100%
7.3	Recipe API request "by recipe name"	Lorenzo Tanzi	10	10	0	3	18/12/2024	20/12/2024	3	100%
7.4	Recipe API request "by ingredient"	Lorenzo Tanzi	10	10	0	3	27/12/2024	05/01/2025	10	100%
7.5	Insertion of saved recipe	Lorenzo Tanzi	4	4	0	3	05/01/2025	08/01/2025	4	100%
7.6	Saved recipes view	Lorenzo Tanzi	4	4	0	3	05/01/2025	09/01/2025	5	100%
7.7	Recipe view	Lorenzo Tanzi	6	6	0	3	07/01/2025	08/01/2025	2	100%
7.8	Removal of saved recipe	Lorenzo Tanzi	4	4	0	3	08/01/2025	08/01/2025	1	100%



4. Conclusions

In this project we have focussed on the design of a microservices system, implemented by using advanced technologies and design practices to ensure an high-quality product.
The usage of those technologies and design techniques helped to create an efficient and scalable distributed Smart Cuisine application.
The use of the Docker network and RabbitMQ improved the infrastructure management and communication between components (synchronous and asynchronous).
The containerization of Postgresql, has allowed us to be scalable and modular with also the database infrastructure.
React framework provided us a robust tool for the UI development, using the typescript language.
The adopted design techniques, including user stories, mockups, function points, COCOMO II and SCRUM, played an important role to ensure that the application was well-designed, effectively managed and aligned to the user expectations.