Software Engineering

2023./2024.

Cookbooked

Documentation, Rev. *1*

Group: *Ariel*

Leader: *Cédric LISMONDE*

Date: *17th. november. 2023.*

Teacher: *Nikolina FRID*

Contents

1. Documentation change log

Continuous updating

Rev.	Change description	Authors	Date
0.1	Project Description	Noëlie COMTE	22.10.2023
0.2	Refining Project Description, functional requirements during a team meeting	Noëlie COMTE, Sonia HAMDI, Florian PAILHE	23.10.2023
0.3	Use case description during a team meeting	Sonia HAMDI, Florian PAILHE	26.10.2023
0.4	Refining Use case description and use case diagram	Sonia HAMDI	27.10.2023
0.5	Software Specification Sequence diagrams	Cédric LISMONDE	05.11.2023

Continued on next page

Continued from previous page

Rev.	**Change description**	**Authors**	**Date**
0.6	Architecture Table description and Class diagram during a team meeting	Noëlie COMTE, Sonia HAMDI, Florian PAILHE	12.11.2023
0.7	Architecture Database Diagram	Lara FERNANDES	16.11.2023
0.8	Architecture and design description and Refining Class diagram	Noëlie COMTE, Sonia HAMDI, Lara FERNANDES, Théau MARGOUTI	17.11.2023
0.11			
0.12.1			
0.12.2			
1.0			
1.1			
1.2			
1.3			
1.5			
1.5.1			
2.0			

Continued on next page

Continued from previous page

Rev.	Change description	Authors	Date

There must be major revisions of documents 1.0 and 2.0 at the end of the first and second cycles. Between these revisions, there may be minor revisions depending on how the document is updated. It is expected that after each significant change (addition, modification, removal of parts of the text and accompanying graphic content) of the document, this will be recorded as a revision. For example, revisions within the first cycle will be labeled 0.1, 0.2,. . . , 0.9, 0.10, 0.11 .. until the final revision of the first cycle 1.0. In the second cycle, revisions 1.1, 1.2, etc. are continued.

2. Project assignment description

The objective of this project is to create and implement a website and everything surrounding it, whether it is the coding of the functionalities or the user interface.

CookBooked is a web platform that connects lovers of cooking and baking. CookBooked users will be able to share their favorite recipes, discover new ones and interact with the recipe authors. This platform aims to connect all cooks together while simplifying communication and sharing between users. The recipes are available to all visitors of the website, but registration is required to access all the features provided by the platform. To validate registration, users must input one valid e-mail address, a username, a password, first name and family name which will appear on their profile. Once registered, users will be able to log in by entering their password as well as their username or email address.

Non-registered users can only browse recipes according to the different categories available, types of cuisine or specific ingredients. appetizers, starters, main courses, desserts, pastries etc. constitute the different categories that can be found on the platform. Among the types of cuisine, users will find Mediterranean, Japanese, Indian, French, American and more...

People who register will be able to access all the functionalities of the platform. They will be able to post their own cooking or baking recipes by adding a title to the recipe, and by specifying the various ingredients and the different steps of the recipe. The website will also let them add cooking times and tags such as vegetarian, vegan, gluten-free, etc... Recipe authors will even be able to add photos and videos to the recipe they shared with the community.

Logged-in users can chat with other users. Recipe authors can choose if they are available for the various communication options available on the website. Users can communicate via messaging, integrated chat and video calls. Authors can indicate their availability with precise days or times.

Registered users can also like a recipe, make comments or save it for later. They can also follow the recipe authors to receive updates and new recipes notifications.

Registered users have a public and a private profile. The public profile cannot be modified and is accessible to other members of the CookBooked community. It contains all the recipes published by the registered user, the people he or she follows and those who follow him or her as an author. Kind of like an Instagram profile, with recipes instead of posts.

The private profile, instead, can be modified. This allows the user to manage their personal information (first name, last name, username, profil picture, email...), their communication preferences, but also to access their messaging notifications and updates related to recipes or authors.

The website is maintained by administrators, who are not considered as users. They can manage individual users and recipes. For example, they can change the category of a recipe, delete the recipe or delete some messages in the chat. They can also temporarily or permanently ban a user for inappropriate activity.

The Cookbooked website respects data protection according to GDPR regulation.

To program this site we had to choose object-oriented language python or java. And then use CSS, HTML or JavaScript for the front-end development of the website

Our group decided to program the site in Python and make the user interface in HTML or CSS.

The website must be compatible with different machines, such as mobile phones, tablets and computers, to allow as many people as possible to use it.

3. Software specification

3.1 Functional requirements

Stakeholders:

1. Unregistered user
2. Registered user
3. Administrator
4. Developers
5. Google Calendar and Google Meet API providers

Actors and their functional requirements:

1. Unregister user can:
   (a) Create an account to register
   (b) Browse recipes
      i. Search by categories (appetizers, main dishes, desserts...)
      ii. Search by cuisine types (Mediteranean, Japonese, French...)
      iii. Search by specifics ingredients
   (c) Consult recipes
2. Register user can:
   (a) Log in
   (b) Post recipes
      i. Add a title
      ii. Add the category (Appetizers, main dishes, desserts...)
      iii. Specify ingredients
      iv. Specify preparation steps
      v. Add cooking time
      vi. Add tags (vegetarian, vegan, gluten-free...)
      vii. Add image(s) related to the recipe
      viii. Add video(s) related to the recipe

(c) Consult recipes
   i. Like a recipe
   ii. Comment a recipe
   iii. Save a recipe
   iv. Interact with authors
      A. Messaging
      B. Chat
      C. Video call

(d) Consult authors
   i. Subscribe to users
      A. Receive notifications for new post and message
   ii. See all the posts of the author
   iii. Interact with the author and other users
      A. Messaging
      B. Video call
   iv. Access to his/her public profile

(e) Access and manage his/her private profile
   i. Manage his/her personal information (Username, Name, Surname, Profile Picture, E-mail address)
   ii. Set his/her communication preferences
   iii. Set his/her availability for video calls
   iv. See his/her notifications (messages and video call demands)
   v. Manage his/her posted recipes (Edit and Delete)

3. Administrators can:

(a) Manage users
   i. Temporary ban users
   ii. Permanent ban users
   iii. Delete users

(b) Manage recipe
   i. Change the category of a recipe
   ii. Edit a recipe
   iii. Delete a recipe

(c) Manage chat

i. Delete message

(d) Receive report from registered users

3.1.1 Use cases

Use case description

UC1-Register

- **Main participant:** Unregistered user
- **Goal:** Get access to the application
- **Participants:** Users
- **Prerequisites:** Unregistered
- **Description of the basic course:**
  1. The user input name, surname, email, a picture, a username and set a password
- **Description of possible deviations:**

  2.a The username or email is already taken
  1. Change your username or login

  2.b The password is not allowed
  1. Check the similarity of your password

UC2-Login

- **Main participant:** Registered user
- **Goal:** Get access to everything in the website
- **Participants:** Users
- **Prerequisites:** Register
- **Description of the basic course:**
  1. registered users enters their username and password
  2. if the password is correct the user can access to his profile, see all the recipes and be able to share and exchange with other users.
- **Description of possible deviations:**

  2.a The username or email is already taken
  1. user change his password for a new one

  2.b the username or password is not valid or doesn't exist
  1. user can click on "forgotten password" and change their password
  2. user can re-register again with a new password and username

UC3- Edit user profile

- **Main participant:** Registered user

- **Goal:** change information in a profile
- **Participants:** Users
- **Prerequisites:** user should be in his private profile
- **Description of the basic course:**
    1. User can go in his private profile
    2. Click on the edit button
    3. Change some information : name, surname, username, password, email, profile picture
        (a) to change password the user will have to type the old password first before writing a new one
    4. User can save his changes and be redirected to the main page
- **Description of possible deviations:**
    2.a The user forgot to save his changes
        1. profile information will remain the same as before

UC4-Look at the recipes

- **Main participant:** Registered user, Unregistered user , Administrators
- **Goal:** Discover recipes
- **Participants:** Users
- **Prerequisites:** Go to the app
- **Description of the basic course:**
    1. The users can discover new recipes
- **Description of possible deviations:**
    2.a There is no recipes for your research
        1. Try an other recipes

UC5- Share a recipes

- **Main participant:** Registered user
- **Goal:** Share a recipes
- **Participants:** Users
- **Prerequisites:** Register
- **Description of the basic course:**
    1. Choose a recipe you created
    2. Choose the type of recipe
    3. Post the recipe
- **Description of possible deviations:**

2.a The user leave before posting his recipe
    1. the post is saved but remains in its drafts

2.b The user didn't choose the type of the recipes

UC6-React to recipes

- **Main participant:** Registered user
- **Goal:** Like, comment a recipes
- **Participants:** Users
- **Prerequisites:** Register
- **Description of the basic course:**
    1. Give your opinion about recipes by liking or commenting them
- **Description of possible deviations:**

2.a Please respect our chat policy

UC7- Search for other users

- **Main participant:** Registered user,
- **Goal:** see the profile and recipes of this user
- **Participants:** Users
- **Prerequisites:**
- **Description of the basic course:**
    1. Go to the main page of the website
    2. Click in the search bar
    3. Write the username of the person
    4. Search among the proposed users
- **Description of possible deviations:**

2.a The user doesn't exist

2.b You don't have his username
    1. Type some key words related to his recipes to find the user directly with his posts

UC8- Subscribe to users

- **Main participant:** Registered user,
- **Goal:** Connect with other people and receive notifications about future posts
- **Participants:** Users
- **Prerequisites:** Register and search for the user
- **Description of the basic course:**

1. Click on the user name
2. Click on the ”subscribe” button
3. Wait until the user answer to request

- **Description of possible deviations:**

2.a The user didn't answer

2.b The user refused the request

UC9- Answer to a friend request

- **Main participant:** Registered user
- **Goal:** Mange your friend list
- **Participants:** Users
- **Prerequisites:** Register
- **Description of the basic course:**
  1. Go to your profile
  2. Check the friend request
  3. Accept or refuse the friend request

UC10- Delete friend

- **Main participant:** Registered user
- **Goal:** Manage Friend list
- **Participants:** Users
- **Prerequisites:** Register
- **Description of the basic course:**
  1. Go to a user profile
  2. Delete users

UC11- Contact other users

- **Main participant:** Registered user
- **Goal:** Talk to other users
- **Participants:** Users
- **Prerequisites:** Register
- **Description of the basic course:**
  1. Go to a user profile
  2. Send a message or call the user
- **Description of possible deviations:**

2.a The user is not part of your friend list

UC12- Managing chat

- **Main participant:** Administrators
- **Goal:** Check if the chat rules are respected
- **Participants:** Administrators
- **Prerequisites:** Be an administrators
- **Description of the basic course:**
  1. Manage chat and delete message which are inappropriate
- **Description of possible deviations:**

  2.b This chat as already been deleted

UC13- Managing recipes

- **Main participant:** Administrators
- **Goal:** Check if the content of a recipe is not appropriat
- **Participants:** Administrators
- **Prerequisites:** Be an administrator
- **Description of the basic course:**
  1. change the category of a recipe or delete a recipe
  2. Manage recipes and delete recipes which are inappropriate

UC14- Managing users

- **Main participant:** Administrators
- **Goal:** Manage the behaviour of users
- **Participants:** Administrators
- **Prerequisites:** Be an administrators
- **Description of the basic course:**
  1. temporary ban users or permanent ban
- **Description of possible deviations:**

  2.a This user is already ban

UC15- Report users

- **Main participant:** Registered user
- **Goal:** avoid inappropriate content
- **Participants:** Users
- **Prerequisites:** Register
- **Description of the basic course:**
  1. if a user sees inappropriate content in a post, whether in chat or in the recipe itself, they can tap on it and report it

Use case diagrams

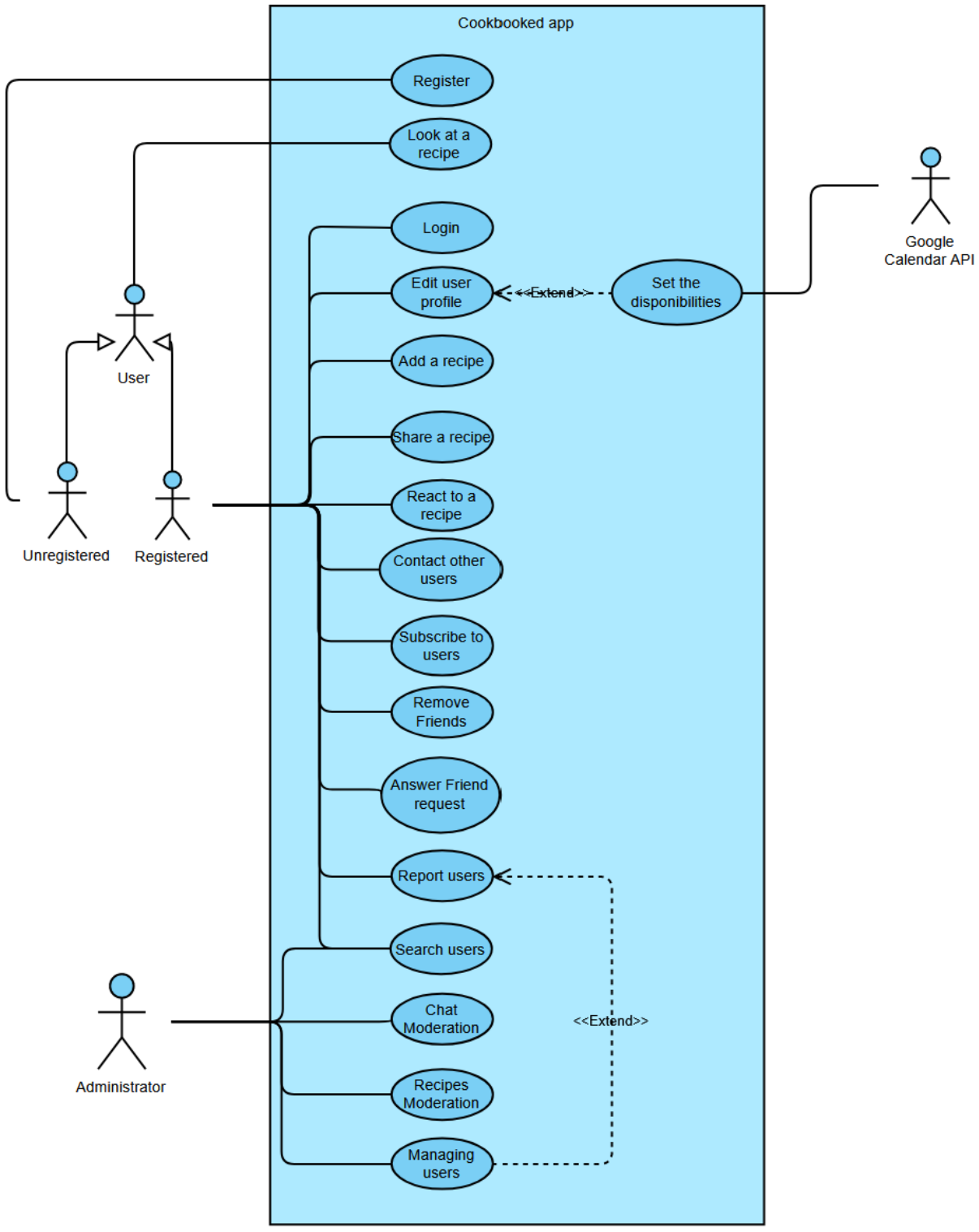

Figure 3.1: Use Case diagrams

3.1.2 Sequence diagrams

Unregistered users can only browse recipes, cuisine types, or specific ingredients.

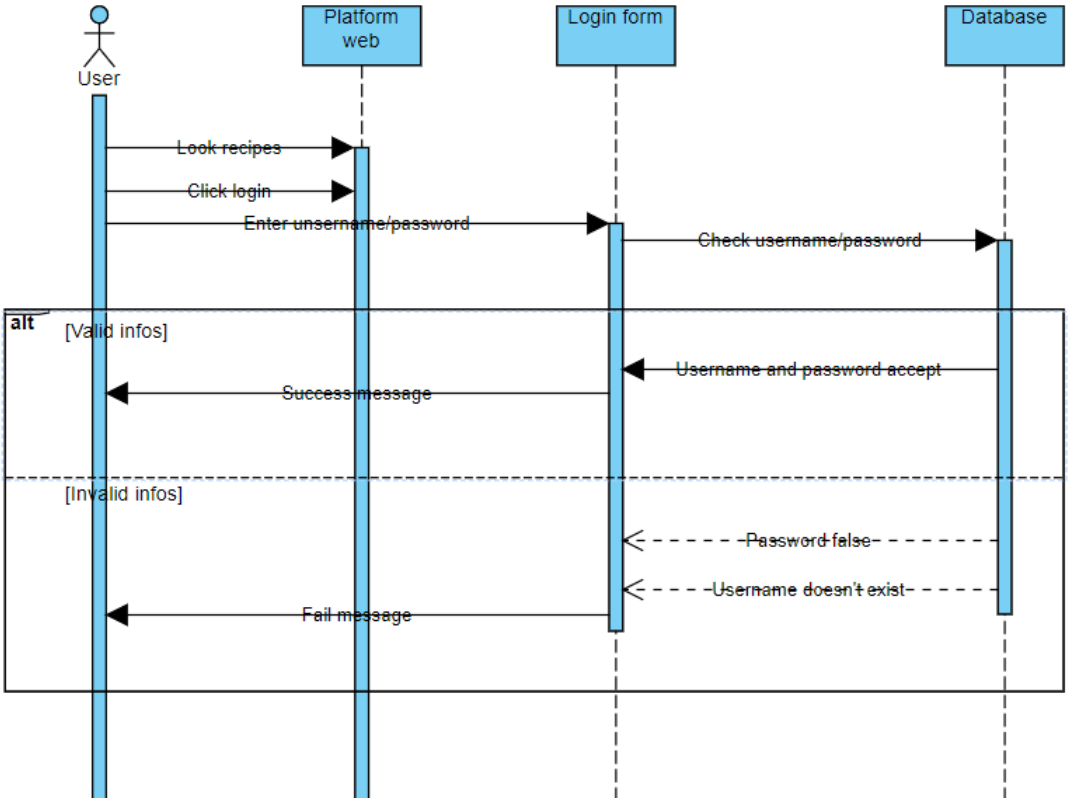

Figure 3.2: Sequence diagram - Login

Once registered, users can post their cooking and baking recipes. They can provide recipe details such as the title, ingredients, preparation steps, cooking time, and tags. Users can also add images and videos related to the recipe.

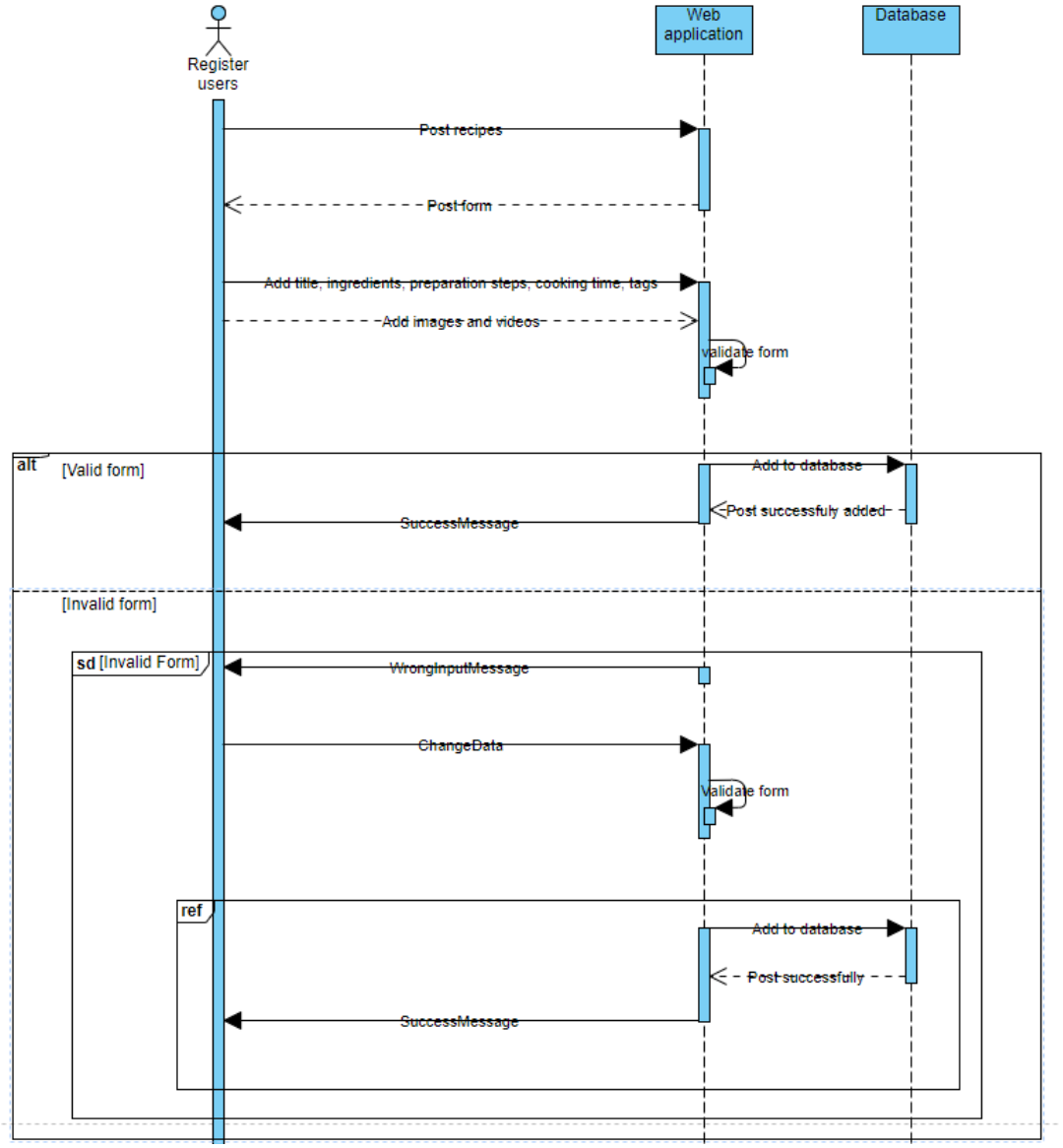

Figure 3.3: Sequence diagram - Add recipe

Registered users can like, comment on, and save recipes for future reference. Users can follow their favorite recipe authors to receive updates on new recipes.

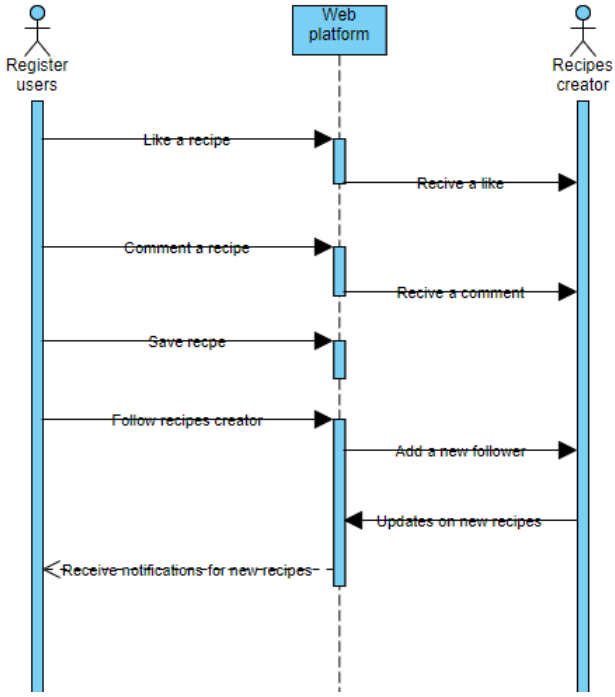

Figure 3.4: Sequence diagram - Interactions with recipes creators

3.2 Other requirements

The website will support only English language and will be accessible on different devices such as mobile phones, computers and tablets. The website is only available on web browser such as Chrome, Firefox, Safari, Edge, etc. The maximum supported number of users is approximately of 1000 users. To protect our users, we will use encrypted passwords.

4. System architecture and design

Our project is a strong and extensible web application designed with a modern architecture. The combination of React for frontend, Django for the backend, PostgreSQL for the database and Tailwind CSS for styling, creates a powerful user experience.

The client side (frontend) of our application is constructed with React. React is not just a library, it is a important JavaScript framework designed for create a software's user interface. Almost all React architectures include:

- Project Directory: contains all configuration documentation, files and other resources.
- src: source directory where the main coding is developed
- Assets: where we store assets that maintain statics like images.
- Components: directory that store re-usable components.
- Services: directory that manages various services or functionalities that aren't directly tied to user interface components, like authentication services.
- Store: where is commonly utilized handling the state of your application.
- Utils: this directory is frequently utilized for handling utility functions, that can appear in various section of your application.
- Views: where you structure the more advanced components that came from the 'Components' directory and engage with services from 'Services' directory .

The diagram below represents how this architecture works.

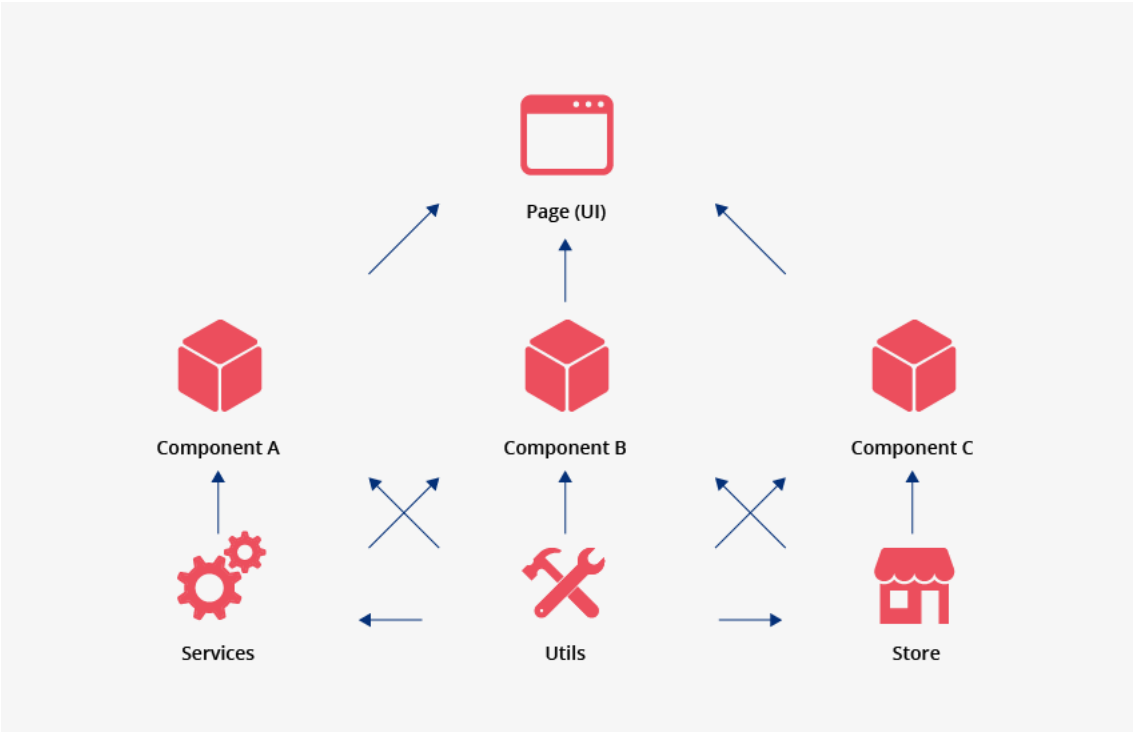

Figure 4.1: React architecture

The backend side of our project was developed using Python Django Framework, more precisely the Model-Template-View architecture pattern. Our innovative web application is divided into 3 parts:

- Model: is responsible to mange the data of the application.
- Template: outlines how to data should be presented to the user
- View: controls what the user sees

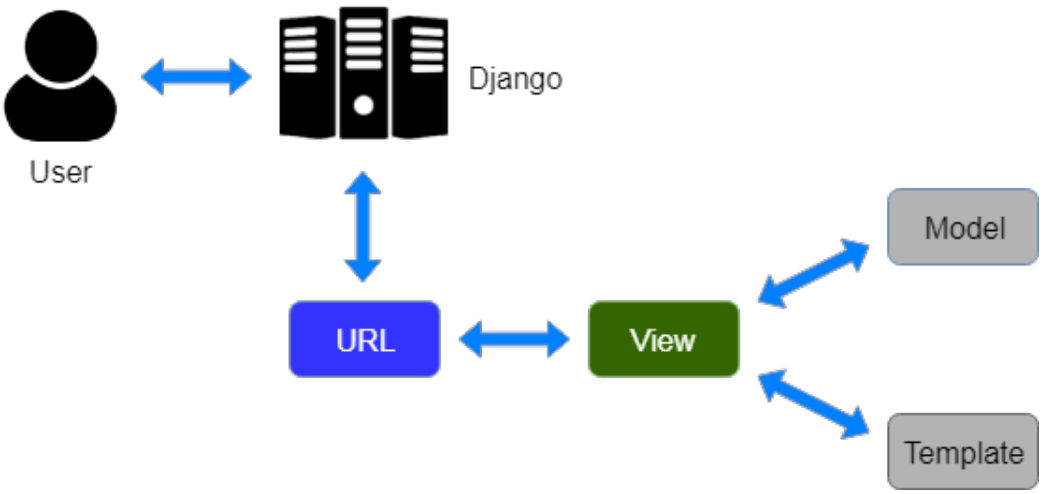

Figure 4.2: Model-Template-View architecture

4.1 Database

We use PostgreSQL database on our server for storing structured data.

4.1.1 Table description

Primary keys are in light green and foreign keys in blue.

Our first table is Register User. Register user is defined by its id, username, name and surname, and email. He needs to have a password and can have a profile picture. He also have the possibility to be limited or banned.

Register User		
user_id	INT	ID user
username	VARCHAR	username
password	VARCHAR	user password
email	VARCHAR	usermail
name	VARCHAR	user name
surname	VARCHAR	user surname
profil picture	VARCHAR	picture of the user
isbanned	BOOLEAN	if the user is banned from the website, boolean response
islimited	BOOLEAN	if the user is limited on the website, boolean response

The recipe table is defined by its id and is connected with user table by user_id but also with ingredients_id from the ingredient table. A recipe have a title, the category, cuisine type, cooking time and preparation steps. The recipe is also connected to tag table. Recipe can also get images and videos and likes.

Recipe		
recipe_id	INT	recipe_id

Continued on next page

Continued from previous page

Recipe		
user_id	VARCHAR	user_id
ingredients_id	VARCHAR	ingredient
recipe_title	VARCHAR	recipe_title
category	VARCHAR	category
cuisine_type	VARCHAR	Type of cuisine
cooking_time	VARCHAR	cooking_time
preparation_step	VARCHAR	preparation_step
tag_list	TAG	tags
image	VARCHAR	recipe image
video	VARCHAR	recipe video
nb_like	INT	number of likes

The ingredient table is defined by the id and contains the ingredient name. This table is useful to permit the the user to add missing ingredients in the database when he creates a recipe and will be useful to define some tags automatically.

Ingredient		
ingredients_id	INT	recipe_id
ingredient_name	VARCHAR	ingredient name
_tag_list	TAG	tag to which the ingredient is linked

The tag table is defined by its id and name. And as the ingredient table, this table is useful to add missing tags in the database from the creation of recipe.

Tag		
tag_id	INT	the tag id
tag_name	INT	the name of the tag

The table admin is defined by the id. The admin table don't have other variable because we don't need more.

Admin		
admin_id	INT	the administrator id

4.1.2 Database diagram

The database diagram presents the graphical structure of the database. The primary and foreign keys are highlighted and tables linked.
We create a many-to-one relationship indicating that for a single user we can have multiple recipes.
Creating the table JointRecipeIngredient, we indicate that multiple ingredients can be associated to a single recipe and that multiple recipes can use the same ingredient. For example, if we have a ”Fruit Salad” and a ”Smoothie” they can use the same ingredient and ”Fruit Salad” can have as many as ingredients does it need.
Creating the table JointRecipeTag, we indicate that we can associate multiple tags with a single recipe and that multiple tasks can have the same tag. For example, if we have multiple recipes related we can tag them the same tag like ”Fast Food” and a unique recipe can have different tags like ”Healthy” , ”Vegetarian”, etc. The graphic was developed using draw.io, as show below.

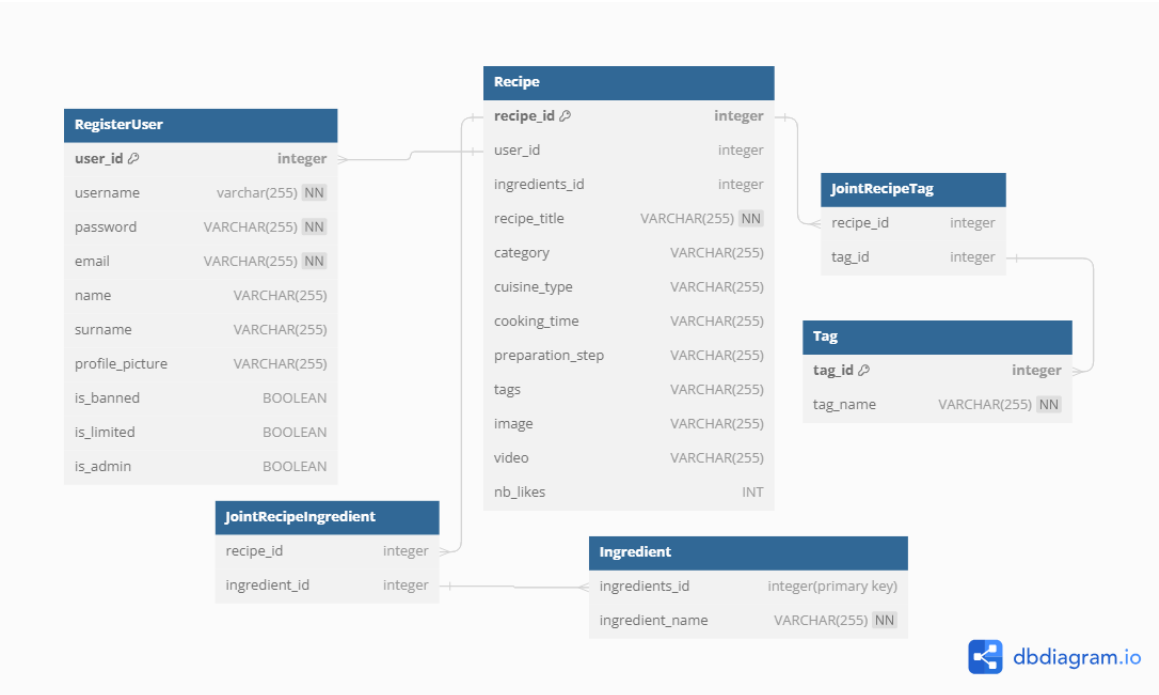

Figure 4.3: Database diagram

4.2 Class diagram

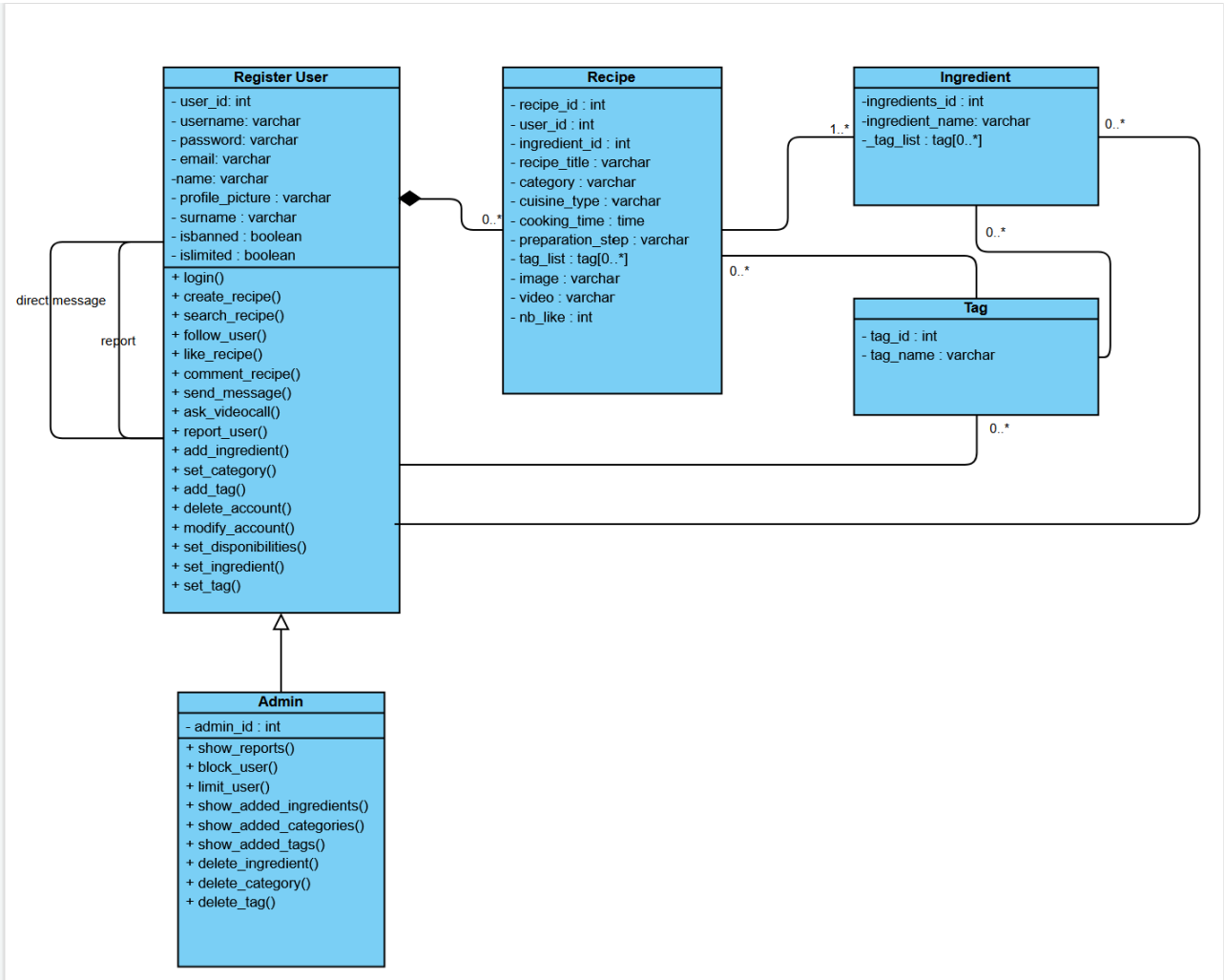

Figure 4.4: Class Diagram

part of the 2nd revision

During the second submission of the project, the class diagram and descriptions must correspond to the actual state of implementation.

4.3 State machine diagram

part of the 2nd revision

*It is necessary to attach a state diagram and describe it. One state diagram showing the **significant part of the functionality** of the system is sufficient. For example, user interface states and the flow of use of some key functionality are a significant part of the system, and registration and login are not.*

4.4 Activity diagram

part of the 2nd revision

It is necessary to enclose a attach of activities with a corresponding description. The activity diagram should show a significant part of the system.

4.5 Component diagram

part of the 2nd revision

A component diagram with the accompanying description must be attached. The component diagram should show the structure of the whole application.

5. Implementation and user interface

5.1 Tools and technologies

part of the 2nd revision

List in detail all technologies and tools used in the development of documentation and applications. Briefly describe them, and state their meaning and place of application. For each of the listed tools and technology it is necessary to ***specify internet link*** *where you can download or learn more about them.*

5.2 Software testing

part of the 2nd revision

In this chapter it is necessary to describe the implementation of testing of implemented functionalities at the level of components and at the level of the whole system with the presentation of selected test cases. Students should examine core functionality and boundary conditions.

5.2.1 Component testing

It is necessary to conduct unit testing on classes that implement basic functionalities. Develop a ***minimum 6 test cases*** *in which regular cases, boundary conditions, and exception throwing will be examined. It is also desirable to create a test case that uses functionalities that are not implemented. It is necessary to enclose the source code of all exam cases and a presentation of the results of the exam in the development environment (passing / failing the exam).*

5.2.2 System testing

The system test should be performed and described using the Selenium footnote `https://www.seleniumhq.org/` *framework. Develop a* ***minimum 4 test cases*** *that will examine regular cases, boundary conditions, and call functionality that is not implemented / cause an error to see how the system responds when something is not fully realized. The test case should consist of an input (eg username and password), the expected output or result, the test step and the output or result obtained.*

The creation of test cases using the Selenium framework can be performed using one of the following two tools:

- *browser extension* ***Selenium IDE*** *- recording user actions for automatic exam repetition*
- ***Selenium WebDriver*** *- support for writing exams in Java, PHP languages using a special programming interface*

Details on the use of the Selenium tool will be presented in a special lecture during the semester.

5.3 Deployment diagram

part of the 2nd revision

*You need to insert a **specification** layout diagram and describe it. It is possible to insert an instance layout diagram instead of a specification layout diagram, provided that this diagram better describes some important part of the system.*

5.4 Deployment instructions

part of the 2nd revision

In this chapter it is necessary to give instructions for deployment of the realized application. For example, for web applications, describe the process by which the source code leads to a fully set up database and server that responds to user queries. For a mobile application, the process by which the application is built and placed on one of the stores. For a desktop application, the process by which an application is installed on a computer. If mobile and desktop applications communicate with the server and / or database, describe the procedure for setting them up. When creating instructions, it is recommended that ***highlight installation steps using hints*** *and use* ***screenshots*** *as much as possible to make the instructions clear and easy to follow.*

The completed application must be running on a publicly available server. Students are encouraged to use one of the following free services: Amazon AWS, Microsoft Azure or Heroku. Mobile apps should be released on the F-Droid, Google Play or Amazon App Store.

6. Conclusion and future work

part of the 2nd revision

In this chapter it is necessary to write a review of the time of project assignment, what technical challenges have been identified, whether they have been solved or how they could be solved, what knowledge was acquired during project development, what knowledge would be especially needed for faster and better project implementation. and what would be the prospects for continuing work in the project team.

It is necessary to accurately list the functionalities that are not implemented in the realized application.

References

Continuous updating
List all references and literature that helped in the realization of the project.

1. Oblikovanje programske potpore, FER ZEMRIS, http://www.fer.hr/predmet/opp
2. I. Sommerville, "Software engineering", 8th ed, Addison Wesley, 2007.
3. T.C.Lethbridge, R.Langaniere, "Object-Oriented Software Engineering", 2nd ed. McGraw-Hill, 2005.
4. I. Marsic, Software engineering book", Department of Electrical and Computer Engineering, Rutgers University, http://www.ece.rutgers.edu/~marsic/books/SE
5. The Unified Modeling Language, https://www.uml-diagrams.org/
6. Astah Community, http://astah.net/editions/uml-new
7. Test Test, 748Test

Index of figures

Appendix: Group activity

Meeting log

Continuous updating

It's necessary to frequently update the meeting log according to the template.

1. meeting
    - Date: October 23, 2023
    - Attendees: Noëlie COMTE, Sonia HAMDI, Florian PAILHE
    - Meeting subjects:
        - Project description
        - Functional requirements
2. meeting
    - Date: October 26, 2023
    - Attendees: Sonia HAMDI, Florian PAILHE
    - Meeting subjects:
        - Use case description
3. meeting
    - Date: November 12, 2023
    - Attendees: Noëlie COMTE, Sonia HAMDI, Florian PAILHE
    - Meeting subjects:
        - Architecture table description
        - Class diagram

Activity table

Continuous updating

Note: Activity contributions should be entered as hours contributed by person and activity.

	Cédric LISMONDE	**Sonia HAMDI**	**Noëlie COMTE**	**Florian PAILHE**	**Théau MARGOUTY**	**Lara FERNANDES**	**Taylan ÜNVER**
Project management		2.5	2.5				
Project task description	0	0.5	2.5	0	0	0	0
Functional requirements	0	2	2	2	0	0	0
Individual patterns description	0	3	0	2	0	0	0
Patterns diagram	0	1.5	0	0.5	0	0	0
Sequence diagram	5	0	0	0	0	0	0
Other requirements description	0	0.5	0.5	0	0	0	0
System architecture and design	0	0	0	0	0	1	0
Database	0	3.5	3.5	3.5	0	2.5	0
Class diagram	0	2.5	2.5	1	1	0	0
State diagram							
Activity diagram							
Components diagram							
Used technologies and tools							
Solution testing							
Layout diagram							
Deployment instructions							

Continued on next page

Continued from previous page

	Cédric LISMONDE	**Sonia HAMDI**	**Noëlie COMTE**	**Florian PAILHE**	**Théau MARGOUTY**	**Lara FERNANDES**	**Taylan ÜNVER**
Meeting log							
Conclusion and future work							
References							
Additional task examples							
Welcome page creation							
Database creation							
Connecting to the database							
back end							

Change log diagrams

part of the second revision

Import generated change log diagrams from GitLab to this chapter. Diagrams can be reached at GitLab at Repository/Contributors.