**Final Year Project**

**FYP-2 Final-Evaluation Report**

**Pakalo – Artificially Intelligent Cooking Assistant**

**FYP Team**

**Abid Waqar i16-0229**

**Mashood Ur Rehman i16-0063**

**Muhammad Nauraiz Mushtaq i16-0106**

**Supervised By**

**Dr. Omer Beg**

**FAST School of Computing**

**National University of Computer and Emerging Sciences**

**Islamabad, Pakistan**

**2020**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Students’ Submission**

FYP-2 Final-Evaluation Report - Pakalo

**Anti-Plagiarism Declaration**

This is to declare that the above FYP report produced under the:

**Title:** FYP-2 Final-Evaluation Report - Pakalo

is the sole contribution of the author(s) and no part hereof has been reproduced on **as it is** basis (cut and paste) which can be considered as **Plagiarism**. All referenced parts have been used to argue the idea and have been cited properly. I/We will be responsible and liable for any consequence if violation of this declaration is determined.

Date: June 10th, 2020 Student 1

Name: Abid Waqar

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student 2

Name: Mashood Ur Rehman

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student 3

Name: Muhammad Nauraiz Mushtaq

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Supervisor (Faculty)

Name: Dr. Omer Beg

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Table of Contents**

Students’ Submission iii

Anti-Plagiarism Declaration iii

1. Introduction2
   1. Project Overview 2
2. Competitive Analysis3
3. Project Vision4
   1. Problem Statement 4
   2. Business Opportunity4
   3. Objectives 5
   4. Project Scope5
   5. Constraints6
   6. Stakeholders Description6
      1. Stakeholders Summary 6
      2. Key High Level Goals and Problems of Stakeholders 7
4. Software Requirement Specifications8
   1. List of Features8
   2. Functional Requirements8
   3. Quality Attributes 9
   4. Non-Functional Requirements9
5. High Level Use Cases 11
   1. Use Case Diagram 11
   2. List of Use Cases 11
   3. High Level Use Case Descriptions 12
6. Iteration Plan 18
7. Domain Model 19
8. Class Diagram 19
9. Architecture Diagram 20
10. Package Diagram 21
11. Deployment Diagram 22
12. Iteration 1 23
    1. Expanded Use Cases 23
    2. Activity Diagram26
    3. System Sequence Diagram27
    4. Operation Contracts28
    5. Sequence Diagrams29
13. Iteration 2 31
    1. Expanded Use Cases 31
    2. Activity Diagram34
    3. System Sequence Diagram36
    4. Operation Contracts37
    5. Sequence Diagrams38
14. Iteration 3 and 4 40
    1. Expanded Use Cases 40
    2. Activity Diagram47
    3. System Sequence Diagram52
    4. Operation Contracts55
    5. Sequence Diagrams57
15. Implementation Details 60

References 62

Appendices 63

1. **Introduction**

One of the most basic activities common amongst every human is the act of cooking. In some way, shape, or form, we need to cook food for consumption. Even though this activity can be perceived as vital to the survival of the human race, it has also gained a reputation as an art form. To put together delicious dishes is a craft many spend years learning and perfecting.

Beit for survival, or for the pursuit of art, many individuals, fascinated by this act, find it difficult to pursue their interests in cooking. They are often intimidated by the perceived difficulty of cooking, or because they have previously failed to produced favourable results in their cooking endeavors.

Pakalo aims to tear down this formidable wall and guide the up and coming chefs, by taking them step by step through recipes they wish to cook, and aid them in their culinary venture.

* 1. **Project Overview**

Pakalo is an application that contains recipes and has a chatbot. The chatbot uses speech and text input from its users and understands it to be able to find recipes, go step by step throughout the recipe alongside the user, suggest recipes according to the user’s interests and the limited ingredients a user has, mention ingredients that are required to make a particular recipe, suggest halal substitute ingredients for recipes with their opposite counterparts and more.

Apart from the chatbot, users will have his or her own account through which users can upload their own recipes. Users can also comment, rate, and share recipes of other users that they liked.

As the user will communicate with the chatbot through either speech or text, the chatbot will use Natural Language Processing to deduce what the user is trying to say. The chatbot will then check the database for helping material and then take action accordingly.

1. **Competitive Analysis**

In the current market, there are plenty of applications and websites that aim to help its users cook in various ways. There are apps like *Allrecipes* and *BigOven* that offer a large selection of recipes, with detailed instructions, and even videos, along with a social aspect where users can add their own recipes and leave feedback on other ones. Then there are apps like *Halal Food Trucks* that make it easier for a user to decide what and where they want to eat by recommending restaurants and diners around them. However, despite the numerous applications and websites that aid their consumers with cooking, none of them offer a chatbot that acts as a culinary guide and guides its users along a recipe at the user’s own pace.

Culinary chatbots do exist, but they are not a guide to the user in a way that Pakalo is. A chatbot called *Betty Crocker* is an app offered for Amazon’s Alexa. This chatbot helps the user by answering culinary questions and recommending substitutes. Another chatbot for the Alexa is *Make It Soy*, which helps the user find vegan substitutes for ingredients. *Seamless* is another chatbot that helps its user order food quickly. Several examples of culinary chatbots like these exist, but none of them offer the user guidance, at their own pace, along the recipe. It is obvious that Pakalo fills a massive gap in the culinary market.

1. **Project Vision**

Pakalo, with its bot as a core feature, will aim to revolutionize cooking. More and more smart home solutions are coming into fruition and this bot will be a significant addition to the smart kitchen. The application will be user friendly, similar to the Google Assistant, only it will be focused primarily on cooking.

* 1. **Problem Statement**

Cooking is not just something humans do to survive, it is an activity that unites them, despite superficial differences. And despite its basic nature, cooking has evolved into a form of expression and art. Even though almost everyone knows how to cook in some form, there are still those who find it difficult to break into this medium. These people include, but are not limited to, students living away from home and without their home cooked meals, young and aspiring chefs who are fascinated with the concept of fine dining and the art of cooking et cetera.

The reason many people find cooking to be a difficult medium to break into is that they lack a proper guide; a culinary teacher that will help them step by step to cook what they want to cook, and help them in any way they need. Even though there exist conventional means of learning how to cook, such as written recipes and video tutorials, these methods have major flaws.

The process of cooking demands attention and focus, and requires the cook to multitask. With this amount of work, it is significantly difficult for a cook to go back and forth from reading recipes, making notes, pausing or rewinding the videos, to actually cooking. In a kitchen environment, where the hands often get dirty, recipe books, notes, smartphones, and tablets are sure to get dirty too as a result of the cook often touching them.

* 1. **Business Opportunity**

As discussed in the competitive analysis, even though many culinary applications and culinary chatbots exist, none of them aim to solve the problem that Pakalo does. These applications, exceptional though they are, are simply not designed to tackle the issue of helping aspiring chefs head on. There is a clear gap in the market. And there is a clear need for a cooking assistant that truly does help a user cook. Not only will Pakalo guide the user with their cooking, but it will also offer a social platform for aspiring cooks to communicate and help each other along their journey.

This is where Pakalo will benefit the most. An appropriate analogy to describe how Pakalo could shake up the current market is that of Siri. Though the functionality of the two chatbots differ significantly, the way Siri shook up the market by providing a chatbot that allowed users to carry out tasks through one interface, is a potential future Pakalo could find itself in. Not to mention, since Pakalo is an application for smartphones, it will further add to the rising trend of smart home features integrated into smartphones.

The launch of Pakalo could potentially force other culinary apps to compete and recognize Pakalo as a serious competitor as it would be the sole application that provides this unique feature, which is a position many companies dream of finding themselves in.

* 1. **Objectives**
* To provide a step by step guidance at the user’s own pace when cooking recipes they would love.
* To recommend recipes based on a limited set of ingredients, similar interests, and user’s history.
* To have a smart easy to talk to chatbot that the user will be comfortable talking to, and that will aid the user in cooking.
* To make a rich user friendly social experience for cooks, with user generated recipes and ratings.
* To suggest substitution of ingredients in cases where they are not available or undesirable
  1. **Project Scope**

Palalo is an application to be developed for Android. This application will contain a database of recipes, which will be available for users to access. The defining part of Pakalo, however, is the integrated and vital chatbot that will serve as the main medium of interaction with the user. This chatbot will guide the user step by step along the recipe that the user has decided to cook. It will also help the user decide what they should cook by recommending recipes to them, based on their previous likes, or based on what they have on hand at the moment.

To achieve this goal, the project activities includes gathering recipes data, categorization of ingredients, formation of a Natural Language Processing model for the chatbot, and development of a knowledge graph as well as a mobile Android application.

* 1. **Constraints**

This development of Pakalo will be spread across a time period of about 9 months, or two semesters of university, Fall and Spring. The time constraint, therefore, will be for the deliverance of ample updated material on the scheduled FYP-I Mid and Final evaluations, and then on FYP-II Pre-Job Fair and Final Evaluations. The project is also divided into 4 iterations which will be discussed in detail further in Chapter 6.

Since the project is primarily focused on software, no hardware costs are present. However, there is a cost constraint. Because the application will require the use of Google’s Firestore and Google’s Cloud Services, we will need to purchase a package. Google does, however, offer free variants, with a 12-month period wherein we can use $300 worth on these services. Although, whether that $300 will last throughout the 9-month time is still uncertain.

With respect to the project scope, which was discussed in Chapter 3.4, the main challenge faced will be the correct implementation of the knowledge graph, recognizing through the user’s input on which step of the recipe they are on, and an optimized NLP model running within the phone application.

* 1. **Stakeholders Description**
     1. ***Stakeholders Summary***

|  |  |
| --- | --- |
| **Stakeholders** | **Priority** |
| Project Lead | High |
| Project Development Team | High |
| Application User | High |
| Potential Investors | Medium |
| Potential Owner | Low |

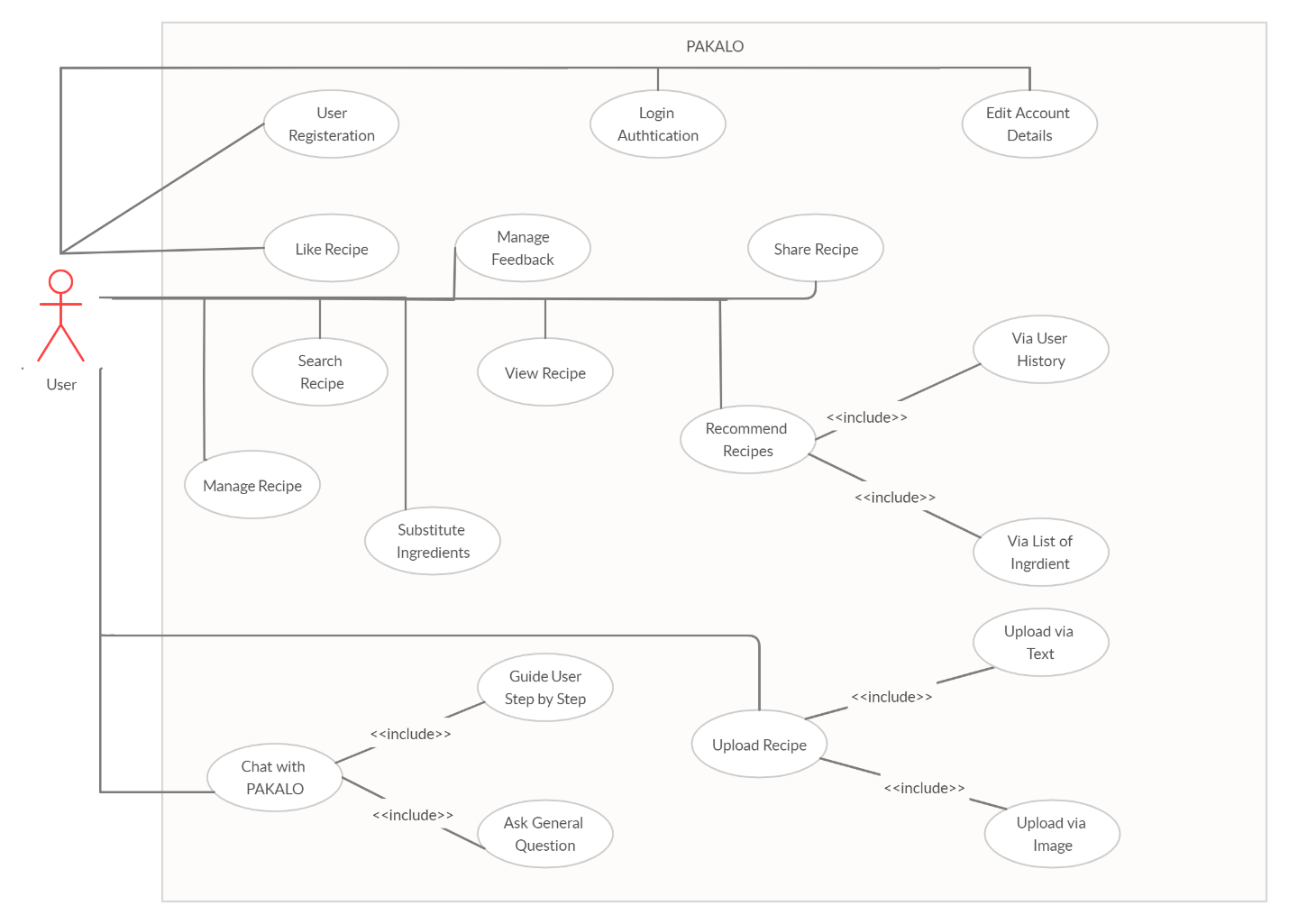
* + 1. ***Key High Level Goals and Problems of Stakeholders***

|  |  |  |
| --- | --- | --- |
| **Stakeholders** | **High Level Goals** | **Problems** |
| Project Lead | Responsible for setting goals and milestones for the project according to the given constraints. Has to ensure the timely completion of deliverables. Is required to guide the development team, and make executive decisions for the project’s development. | Project lead cannot and should not be too strict nor too generous. The lead sometimes restricts the team and does not allow a healthy open discussion regarding the project. The lead sometimes overworks the team. In other scenarios, the lead does not keep check at all. |
| Project Development Team | Responsible for the entire development, from inception to testing to design of the project. Have knowledge of every minute detail regarding the applications. | Need to work in a timely and cooperative fashion. Any hindrance in communication causes massive delays and set-backs. |
| Application User | Consists of the customers and end users, from the target demographic of the application. The general public. | Are very critical of products they use, and may voice their frustration openly, often damaging the morale of the development team. Are the most important stakeholders. Always need to be pleased. Every part of the project exists to please the users. |
| Potential Investors | With every company, investors play a key role. In case Pakalo decides to become a legitimate tech company, these investors will provide funding for operation and future endeavors. | Often restrict the creative freedom of the company as they have specific demands. Need to be pleased as they provide valuable funding. |
| Potential Buyers | As is often with tech companies, bigger companies may approach Pakalo with hopes of acquiring it. Pakalo will therefore serve as a subsidiary of that company, or will rebrand according to the wishes of the company. | Will cause Pakalo to lose its independent status as a company along with all of its creative freedom. |

1. **Software Requirement Specifications**
   1. **List of Features**

* Recipe recommendations, based on history, preferences, and allergies
* Step by step guidance along the recipe
* Provides a list of ingredients for a recipe, and other information about a recipe
* Recommend substitutes for ingredients
* Relation between recipes and sub-recipes using knowledge graph
* Personal account of every user.
* Uploading recipes using text or an image of recipe text
* Ability to rate, comment, upload, and share recipes
  1. **Functional Requirements**
* A user shall be able to register using email and password.
* A user shall be able to login using registered email and password.
* A user shall be able to search for recipes by typing part of the title of recipe.
* A user shall be able to filter recipes in search based on rating and popularity.
* A user shall be able to view a recipe.
* A user shall be able to like a recipe.
* A user shall be able to comment on a recipe.
* A user shall be able to share a recipe.
* A user shall be able to upload a recipe by filling form containing title, ingredients and instructions to make the recipe.
* A user shall be able to upload a recipe through image of recipe text.
* A user shall be able to edit his password and profile picture
* The app shall be able to recommend recipes to the user based on his interests and allergies.
* The user shall be able to search for substitutes of ingredients.
* The chatbot in app shall be able to guide the user through the recipe.
  1. **Quality Attributes**
* Accessibility: The system will be accessible 24/7.
* Agility: The system will be fast in responding to queries.
* Efficiency: The system will be optimized.
* Fault-tolerance: The system will handle errors responsibly.
* Manageability: The backed logic will be easily managed by the development team.
* Mobility: The app will be available anywhere there is an internet connection.
* Responsiveness: The system will be respond to queries under 30 seconds.
* Robustness: The system will respond correctly every time.
* Scalability: The database and backend logic will be scalable
* Simplicity: The frontend will be simple to use, even for laymen.
* Upgradability: The database and backend logic will be upgradable in the future.
  1. **Non-Functional Requirements**
* ***Performance/Response time requirement***
  + The bot should reply to user queries in under 30 seconds.
  + The Chatbot should reply to the user in a timely manner to keep the conversation flowing without interruption. The bot should delay reply so that it feels like talking to a person.
* ***Availability requirement***
  + The chatbot must be available to chat at any time. This is achieved with Azure Cloud platform services. All code for the chatbot is uploaded to Azure.
  + The recipe data is uploaded to firestore which is also available anytime.
* ***Security requirement***
  + The code must be stored securely by Azure Services and also users information must be kept secure. Facebook and Azure use HTTPS (Secure Socket Layer) for all of their communications.
* ***Reliability requirement***
  + The Chatbot must be consistent with its responses. It must behave consistently in a user-acceptable manner when communicating with a user. The information relayed to the user must be consistent with the dialog topic.
* ***Maintainability requirement***
  + The chatbot must be able to be updated with up to date FAQ’s. This allows the chatbot to develop as time goes on and
* ***Portability requirement***
  + The chatbot logic and recipe data are uploaded to cloud so they can be used for different web based, android and IOS frontends.

1. **High Level Use Cases**
   1. **Use case diagram**

****

* 1. **List of Use Cases**

|  |  |
| --- | --- |
| **Use Case ID** | **Use Case Title** |
| UC-01 | General Question and Answers |
| UC-02 | Search Recipe |
| UC-03 | View Recipe |
| UC-04 | Guide User Step by Step |
| UC-05 | Recipe recommendation based on history |
| UC-06 | Recipe recommendation based on ingredients available |
| UC-07 | Substitute Ingredients |
| UC-08 | Register User |
| UC-09 | Login User |
| UC-10 | Edit User Account |
| UC-11 | Like Recipe |
| UC-12 | Manage Feedback on Recipe |
| UC-13 | Share Recipe |
| UC-14 | Upload Recipe via Text |
| UC-15 | Upload Recipe via Image |
| UC-16 | Manage Recipe |

* 1. **High Level Use Case Descriptions**

|  |  |
| --- | --- |
| **Use Case ID** | UC-01 |
| **Use Case Title** | General Questions and Answers |
| **Actors** | * Primary: User |
| **Description** | This use case follows the user asking the bot general questions. This module is primarily focused on the conversational side of the chatbot. A user can ask the bot to tell them a joke or ask about cooking et cetera. The user will be able to converse with the bot in general, apart from finding recipes and cooking them. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-02 |
| **Use Case Title** | Search Recipe |
| **Actors** | * Primary: User |
| **Description** | This use case allows the user to search for recipes by name or by the type of recipe they wish to eat. For example, a user may search for pizza recipes, and the application will return a list of pizza recipes from the database. The user can then choose to access any resulting recipe. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-03 |
| **Use Case Title** | View Recipe |
| **Actors** | * Primary: User |
| **Description** | This use case includes the user viewing the entirety of any recipe that they wish to view, which was a result of the user searching the recipe, or being recommended it. The recipe includes the list of ingredients and instructions and an option for the user to start the step by step guidance of the recipe. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-04 |
| **Use Case Title** | Guide user Step by Step |
| **Actors** | * Primary: User |
| **Description** | In this use case, the user will follow the guidance of the chatbot, as it guides them throughout the instructions of the recipe. A user can ask about the current step of the recipe, confirm when they are ready to move on, skip to the next step, or even go back. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-05 |
| **Use Case Title** | Recipe recommendation based on history |
| **Actors** | * Primary: User |
| **Description** | In this use case, the user is recommended a list of recipes based on the recipes that they have previously searched or cooked, and recipes that they have rated positively. This recommendation is given to the user on the home screen, and especially when a user asks for recommendations. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-06 |
| **Use Case Title** | Recipe recommendation based on ingredients available |
| **Actors** | * Primary: User |
| **Description** | In this use case, the user mentions a list of ingredients that the user currently has on hand, and Pakalo recommends a set of recipes that the user can cook with just these ingredients. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-07 |
| **Use Case Title** | Substitute Ingredients |
| **Actors** | * Primary: User |
| **Description** | This use case includes the user asking the chatbot for substitutes for ingredients in a recipe that they do not wish to use. This happens notably with halal substitutes of recipes. The user simply asks the bot to suggest substitutes for specific ingredients, and the bot returns a list of potential alternative ingredients. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-08 |
| **Use Case Title** | Register User |
| **Actors** | * Primary: User |
| **Description** | This use case deals with registration of a user’s account. Each user has to provide information for getting registered. Once the account is registered, the user can use the login details in the future. In order to interact with Pakalo user needs to be registered. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-09 |
| **Use Case Title** | Login User |
| **Actors** | * Primary: User |
| **Description** | This use case deals with the authentication of users as they sign in to the Pakalo application. Users need to provide their unique username and password in order to gain recipes and interact with the application in general. If login credentials are incorrect, an error message is displayed. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-10 |
| **Use Case Title** | Edit User Account |
| **Actors** | * Primary: User |
| **Description** | This use case includes the user being able to edit their profile for the social aspect of Pakalo. The user can edit their personal information, account details, bio, and profile pictures. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-11 |
| **Use Case Title** | Like Recipe |
| **Actors** | * Primary: User |
| **Description** | In this use case, the user will be able to rate the recipe that they have cooked. This rating system will be a 5 star based rating. A user can leave a star rating that they have deemed fit for a particular recipe. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-12 |
| **Use Case Title** | Manage Feedback on Recipe |
| **Actors** | * Primary: User |
| **Description** | This use case deals with the user leaving a detailed review, or comment on the recipe that they have cooked. The application will provide a feedback section with a user being able to leave a paragraph based feedback and review, within a certain word limit. If and when a user has decided that the feedback that they had left on a recipe is wrong, outdated, or if they have changed their mind, this use case allows them to delete that particular feedback on a particular recipe |

|  |  |
| --- | --- |
| **Use Case ID** | UC-13 |
| **Use Case Title** | Share Recipe |
| **Actors** | * Primary: User |
| **Description** | In this use case, a user can choose to share a recipe that they really like and think other people may enjoy. The user will be able to share that recipe on a number of social media platforms. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-14 |
| **Use Case Title** | Upload Recipe via Text |
| **Actors** | * Primary: User |
| **Description** | In this use case, the user will be able to upload a new recipe by filling in the recipe details such as ingredients and instructions, on a form-based interface. This method of adding a new recipe will require a user to type out every detail of the recipe they wish to upload. |

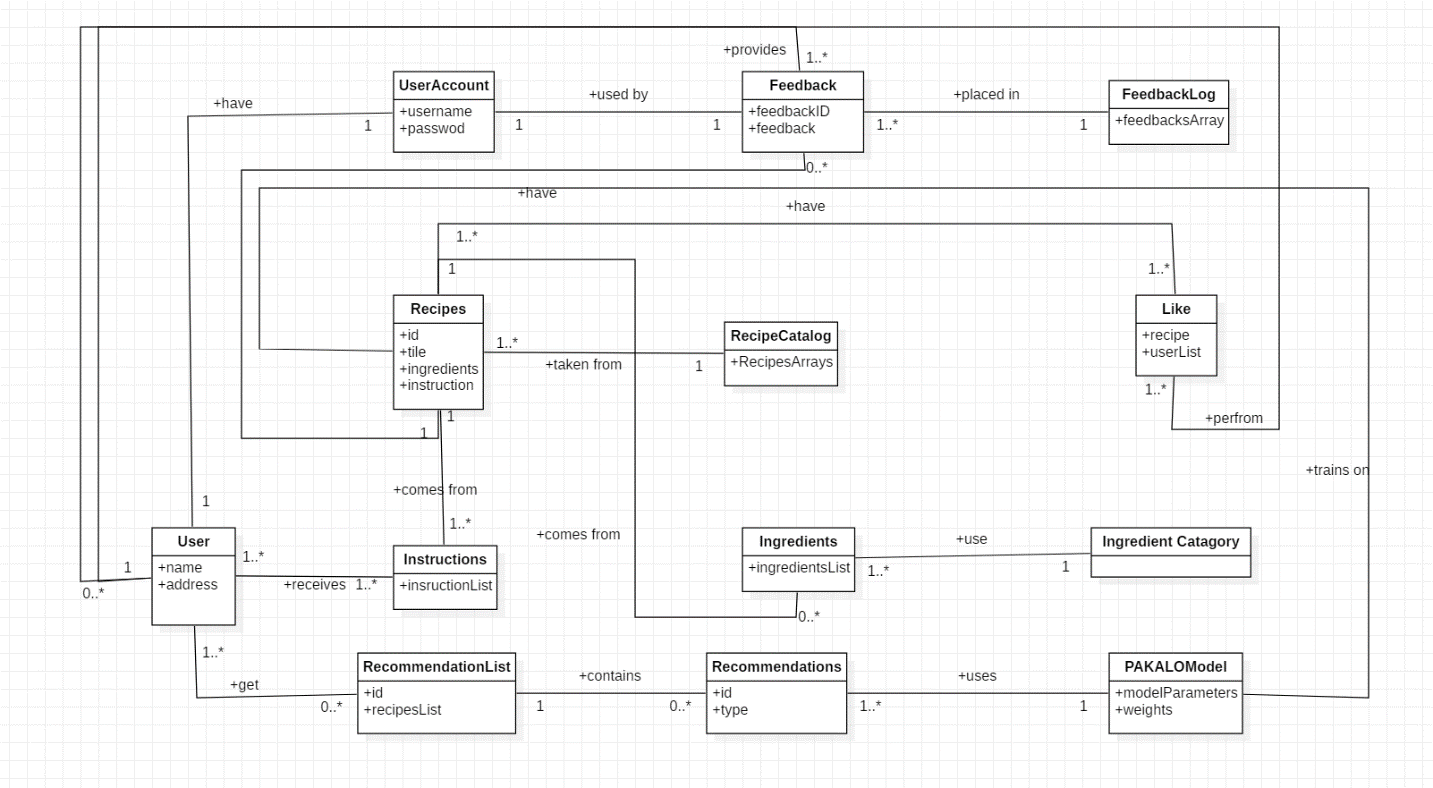
|  |  |
| --- | --- |
| **Use Case ID** | UC-15 |
| **Use Case Title** | Upload Recipe via Image |
| **Actors** | * Primary: User |
| **Description** | In this use case, the user will be able to upload a new recipe by taking a picture of an existing recipe from a cookbook, or by taking a picture of their own written recipes, with ingredients, and instructions. This image data will be fed to a network that will then capture the text data and fill it within the text based form. The user can then edit that form an correct any mistakes. |

|  |  |
| --- | --- |
| **Use Case ID** | UC-16 |
| **Use Case Title** | Manage Recipe |
| **Actors** | * Primary: User |
| **Description** | A user can manage the recipes that they have uploaded to Pakalo. This use case allows the users to delete the recipes that they have uploaded, and to modify and update the recipe if they think it needs to be refreshed. |

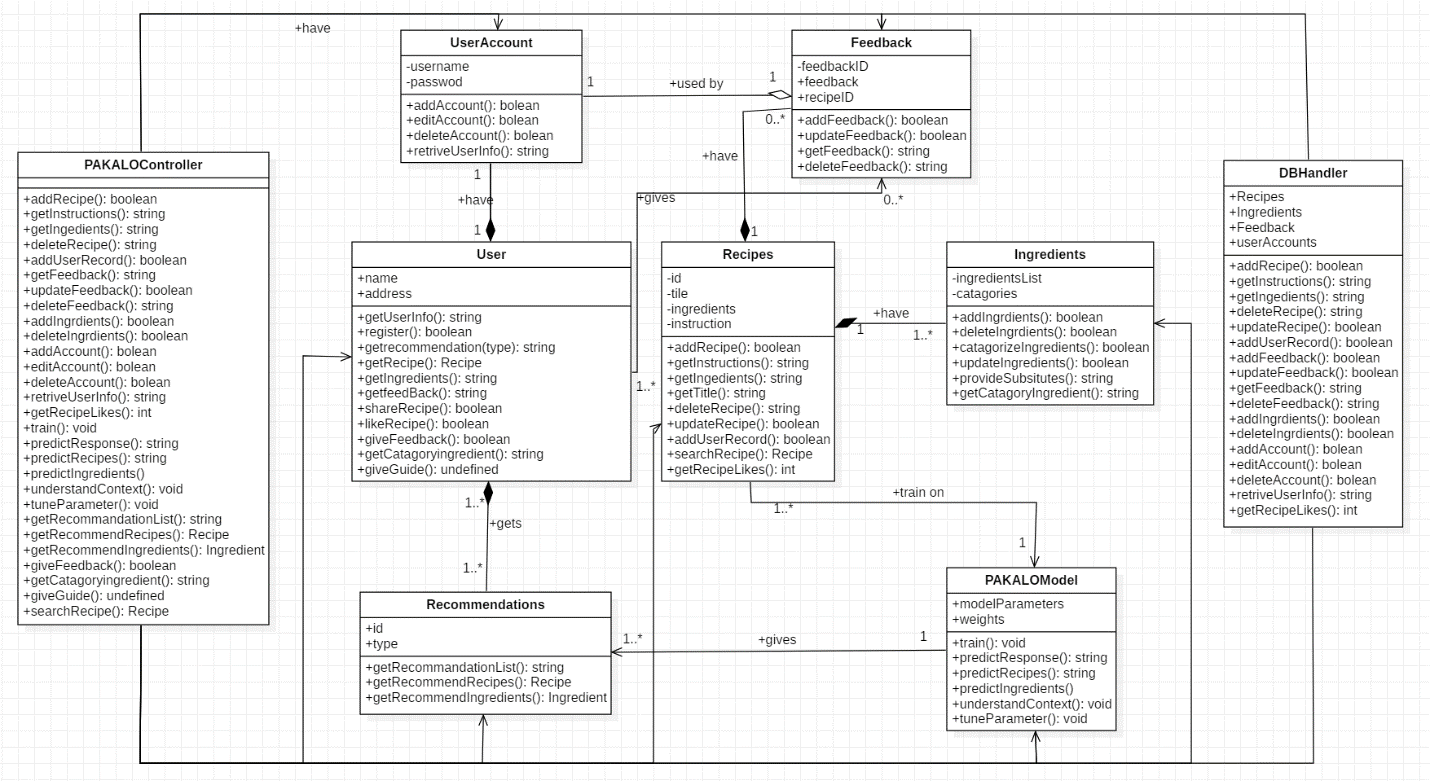
1. **Iteration Plan**

|  |  |  |  |
| --- | --- | --- | --- |
| **Iteration No.** | **Duration** | **Activities** | **Brief Description** |
| One | September | * Data Scraping * Preprocessing * Categorization | This iteration will consist of gathering recipe data from various sources, as well as properly storing this data into usable formats. We will also categorize ingredients according to their food types. |
| Two | October, November | * Model Formation * Model Training * Result Ranking | This iteration will primarily consist of building and training a model for our Natural Language Programming tasks. We will also compare and rank multiple models. |
| Three | December, January, February | * Dynamic Learning * Building Mobile Application | In this iteration, the major focus will be on constructing the Android Application, and developing a pleasant user experience and interface. |
| Four | February, March, April | * Efficient Responsiveness * Feedback | The last iteration will consist of adding the social media aspect to the application, along with a feedback system. The application will also be continually improved upon. |

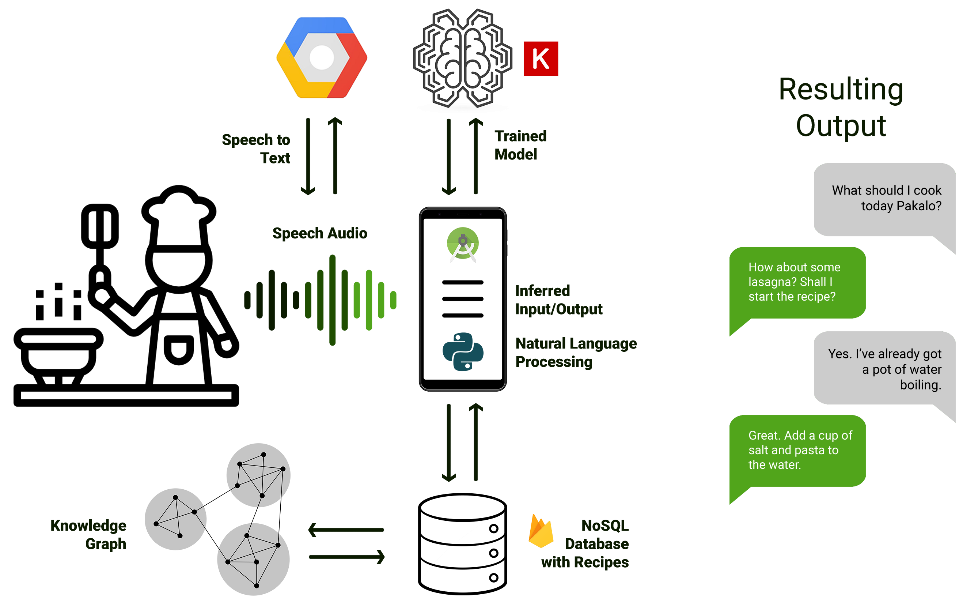
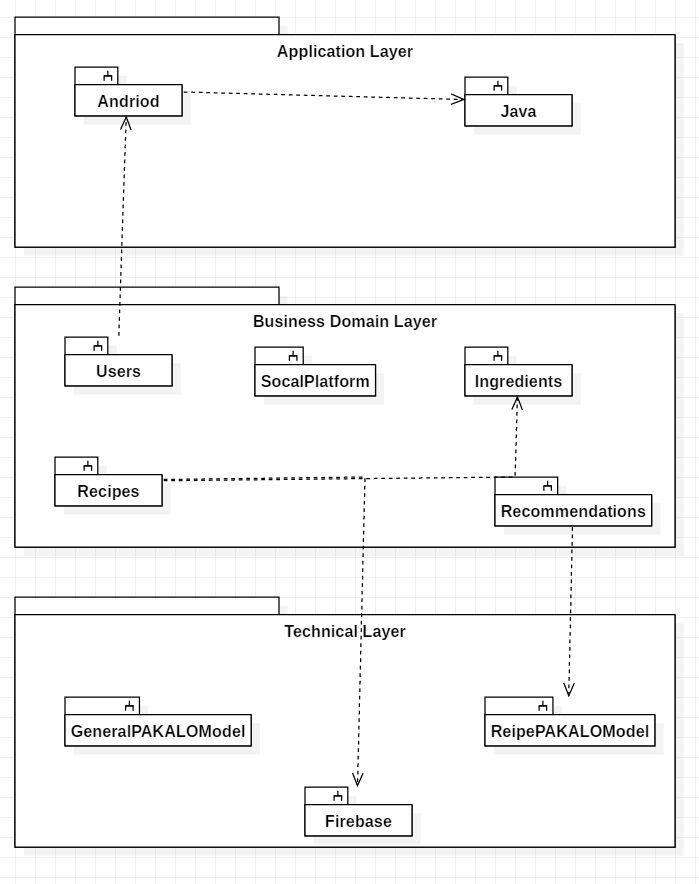
1. **Domain Model**



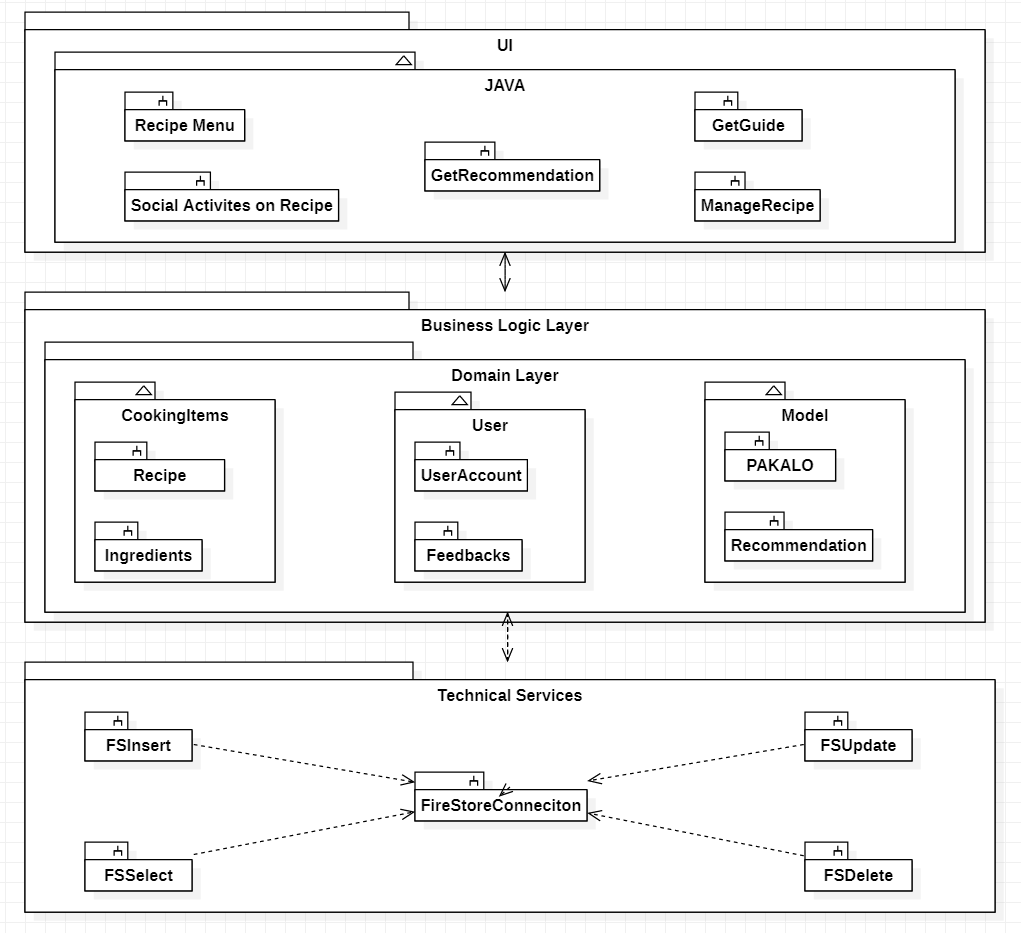
1. **Class Diagram**

****

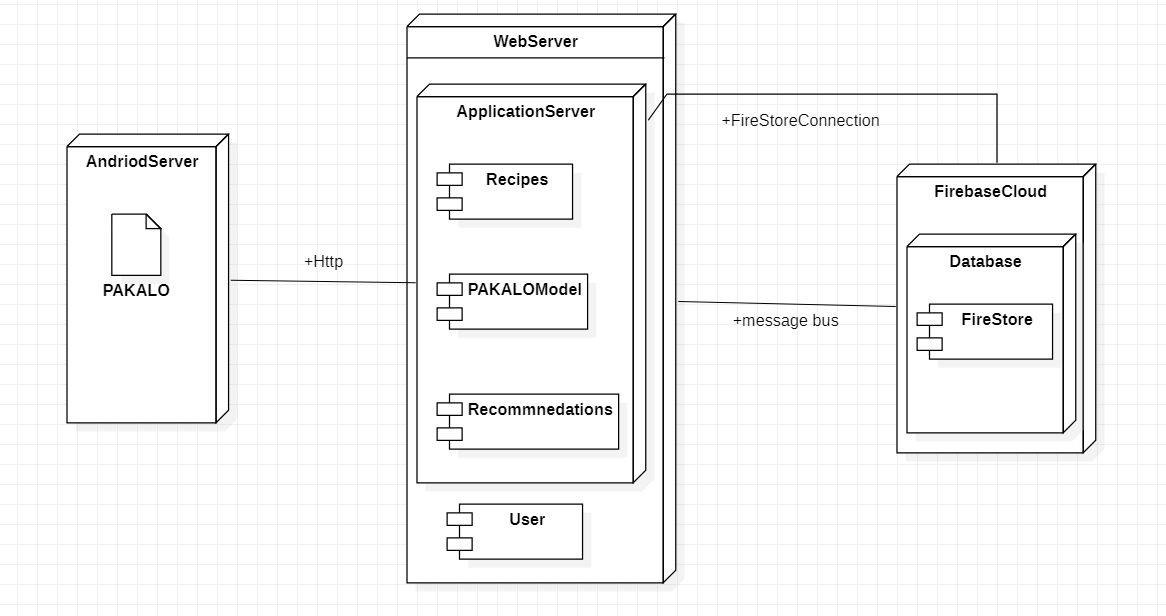
1. **Architecture Diagram**

****

1. **Package Diagram**

****

1. **Deployment Diagram**

****

1. **Iteration 1**
   1. **Expanded Use Cases**

***UC-01: General Question and Answers***

**Scope:** Pakalo

**Level:** User Level

**Actors:** User

**Stakeholders and Interests:** Wants to know about chatbot, and some knowledge related to some computer domain.

**Preconditions:**

1. The user has an account.
2. The user must be logged in.
3. The question should be valid.

**Success Guarantee (Post Conditions):** The User will successfully able to get an answer of question being asked from the Pakalo whether in text or speech.

**Main Success Scenario:**

|  |  |
| --- | --- |
| **Actor Action** | **System Response** |
| 1. The user will open System. |  |
| 1. The user will ask question whether it will be being typed in the message area or/and it will tell the question through speech. |  |
|  | 1. System will recognize the question. |
|  | 1. The system will ask from Pakalo model for answer of question being ask. |
|  | 1. System show the answer and/or it will tell the answer through speech. |
| 1. The User will get the answer. |  |

**Extensions**:

1. If the question being asked is out of context or not valid,
   1. The system will show the message for invalid question or the message could not be understood.
2. If system stops running,
   1. Then the user may have to reopen the system and use it.

**Specific Requirements:**

1. Large text from the system screen should be visible from 500 cm.
2. Understanding of question should be efficient.
3. The question being asked should be valid.

**Technology and Data variation:** Data entering is through typing and/or through speech.

**Frequency of Occurrence:** Near continuous.

**Miscellaneous:** Require Internet Connection

***UC-02: Search Recipe***

**Scope:** Pakalo

**Level:** User Level

**Actors:** User

**Stakeholders and Interests:** Wants to search recipes with the specified name so that they could cook that dish easily and in an efficient way.

**Preconditions:**

1. User has an account.
2. The user must be logged in.
3. At least one recipe including the searched name should exist.

**Success Guarantee (Post Conditions):** The User will successfully able to get a list of recipes related to what they have searched.

**Main Success Scenario:**

|  |  |
| --- | --- |
| Actor Action | System Response |
| 1. The user will go to search bar. |  |
| 1. The user will enter the name of recipe which he wants to get. |  |
|  | 1. System will look for all the recipes that is related to the searched keyword |
|  | 1. System will show the list of recipes related to the searched keyword that it has retrieved. |
| 1. The user will get the list of recipes relating to the keyword they searched. |  |

**Extensions**:

1. If no recipe with the specified name could be found
2. Then the system will show the message to check the recipe name.
3. If there is no spelling mistake, then the system will show the message that recipe could not be found.
4. The system may not respond, then the user may need to restart the system.

**Specific Requirements:**

1. Large text from the recipe on display. Visible from 500 cm.
2. Recipe search from database should be efficient.
3. The name of recipe provided should be valid.

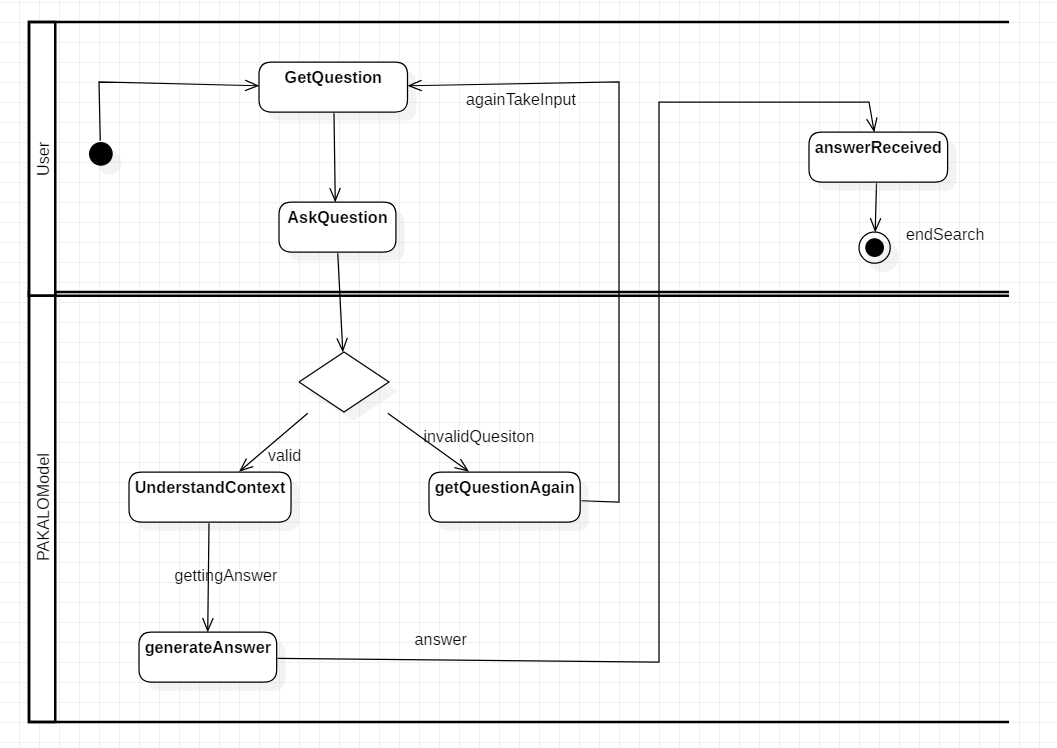
**Technology and Data variation:** Data entering is through typing.

**Frequency of Occurrence:** Near continuous.

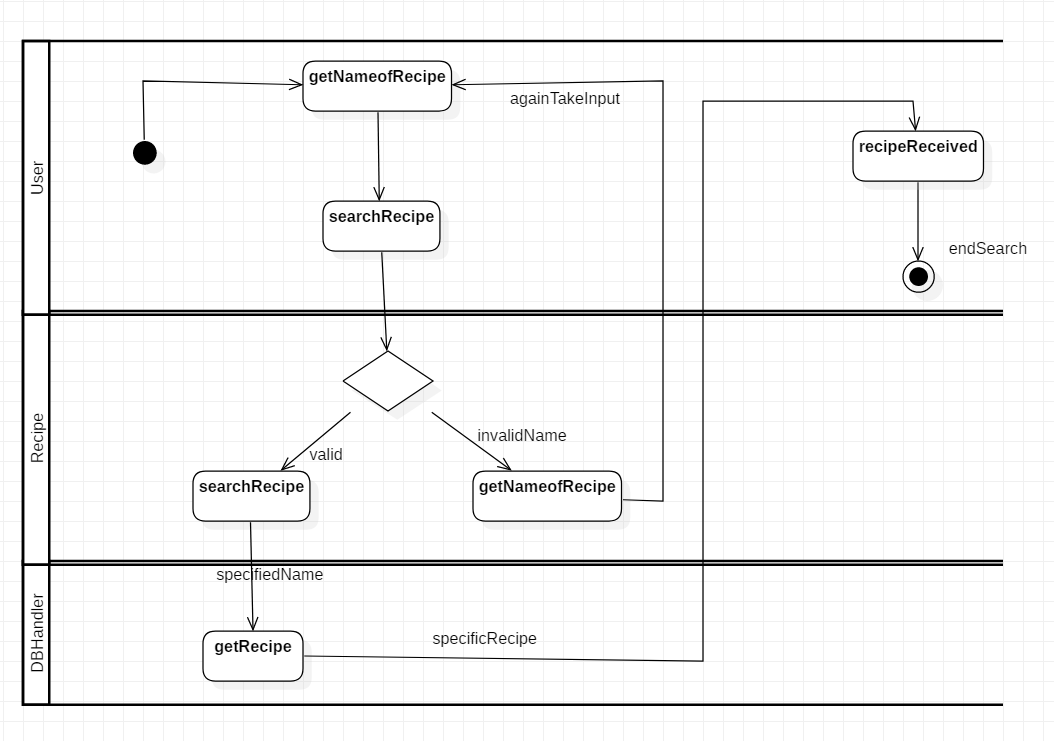
**Miscellaneous:** Require Internet Connection.

* 1. **Activity Diagrams**

***UC-01: General Questions and Answers***

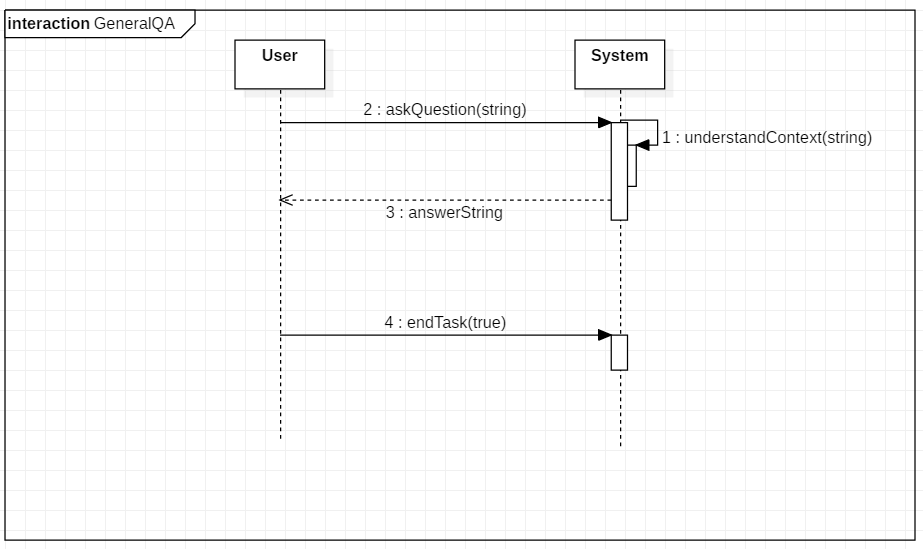
******

***UC-02: Search Recipes***

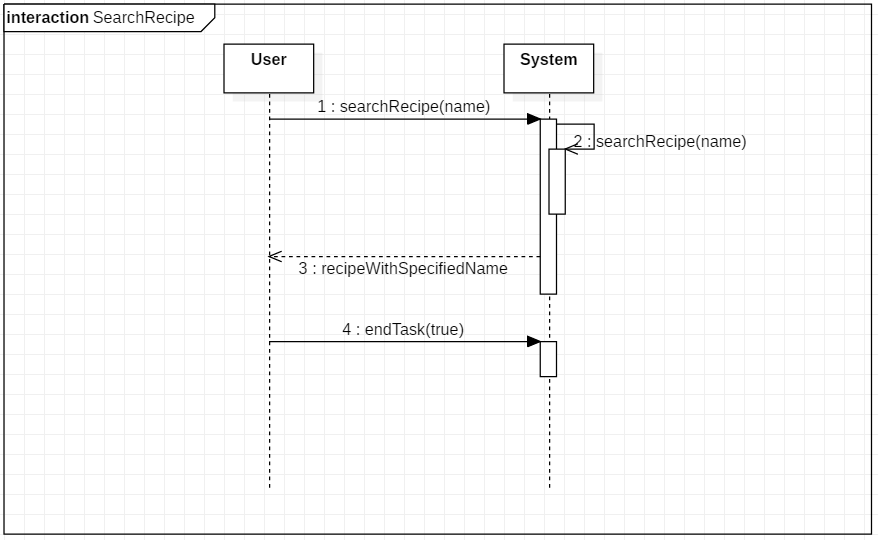
******

* 1. **System Sequence Diagrams**

***UC-01: General Questions and Answers***

******

***UC-02: Search Recipes***

******

* 1. **Operation Contracts**

***Contract CO1: SearchRecipe***

**Operation:** searchRecipe(name)

**Responsibility:** Finds the recipes with the specified name and returns them.

**Cross References:** Search Recipe

**Preconditions:** User must be logged in.

**Post conditions:**

1. A recipe instance r is created.
2. Recipe r was associated with DBHandler.
3. Attributes of r were initialized.
4. r.title became title.
5. Every created recipe instance r is returned in a list.

***Contract CO2: AskQuestion***

**Operation:** AskQuesiton(question)

**Responsibility:** Finds answer of the question being asked and returns it.

**Cross References:** Chat with Pakalo

**Preconditions:** User must be logged in.

**Post conditions:**

1. A model instance m was created.
2. Model m was associated with PAKALOModel.
3. Attributes of m were initialized.

***Contract CO3: UnderstandContext***

**Operation:** understandContext(question)

**Responsibility:** Understands what user is talking about and returns the featured words.

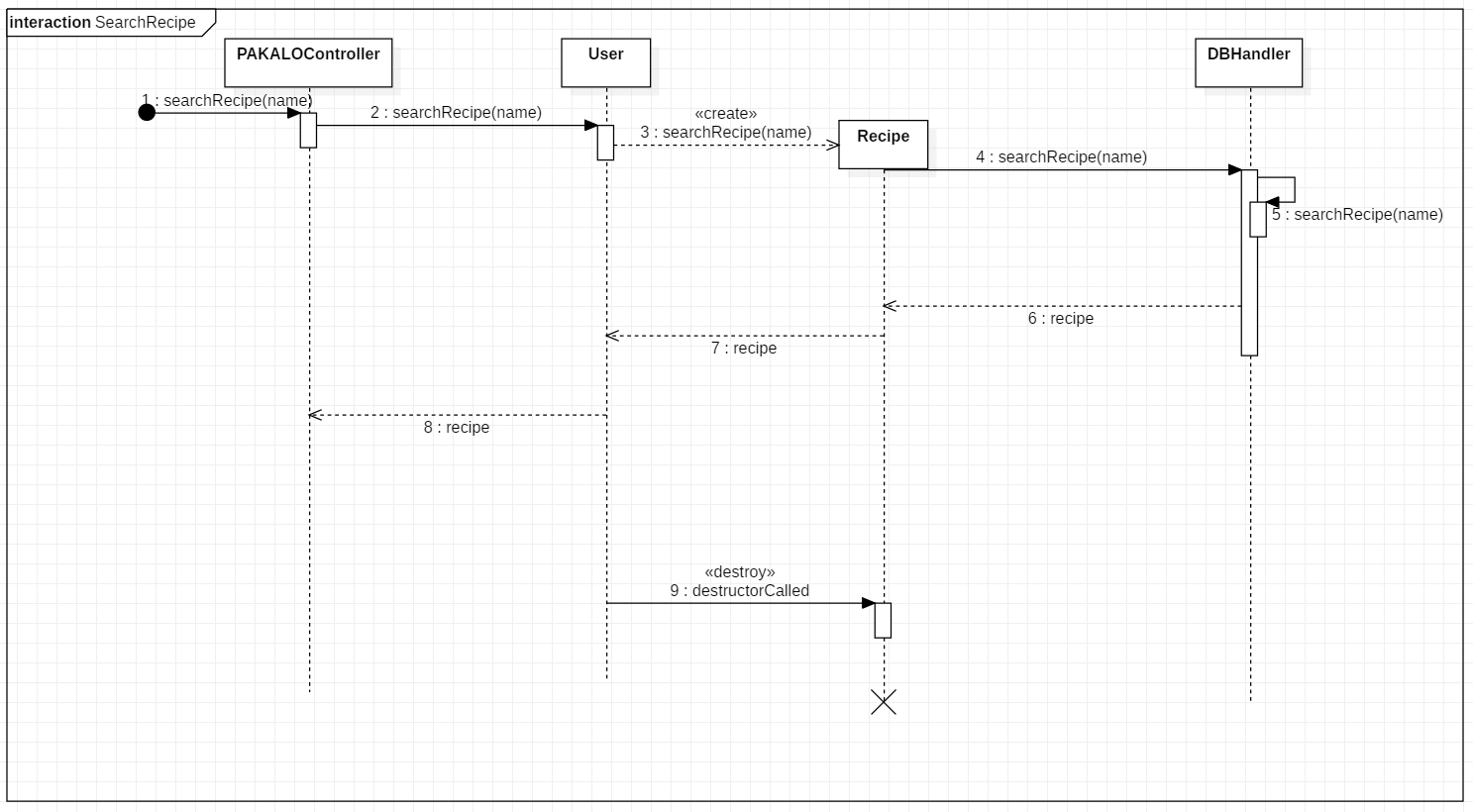
**Cross References:** Chat with Pakalo

**Preconditions:** User must be logged in.

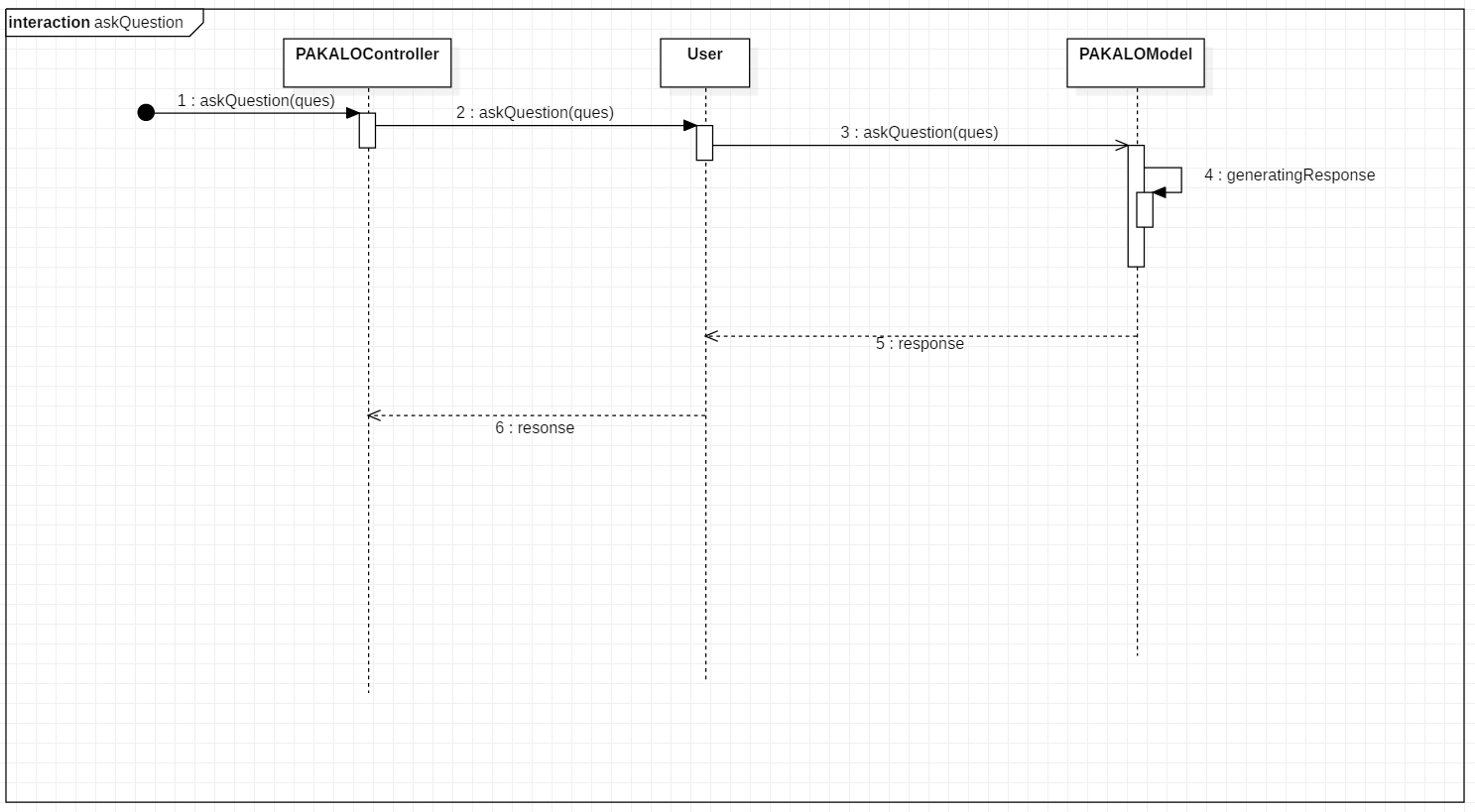
**Post conditions:**

1. A model instance m was created.
2. Model m was associated with PAKALOModel.
3. Model m.string Query became question.
4. Attributes of m were initialized.
   1. **Sequence Diagrams**

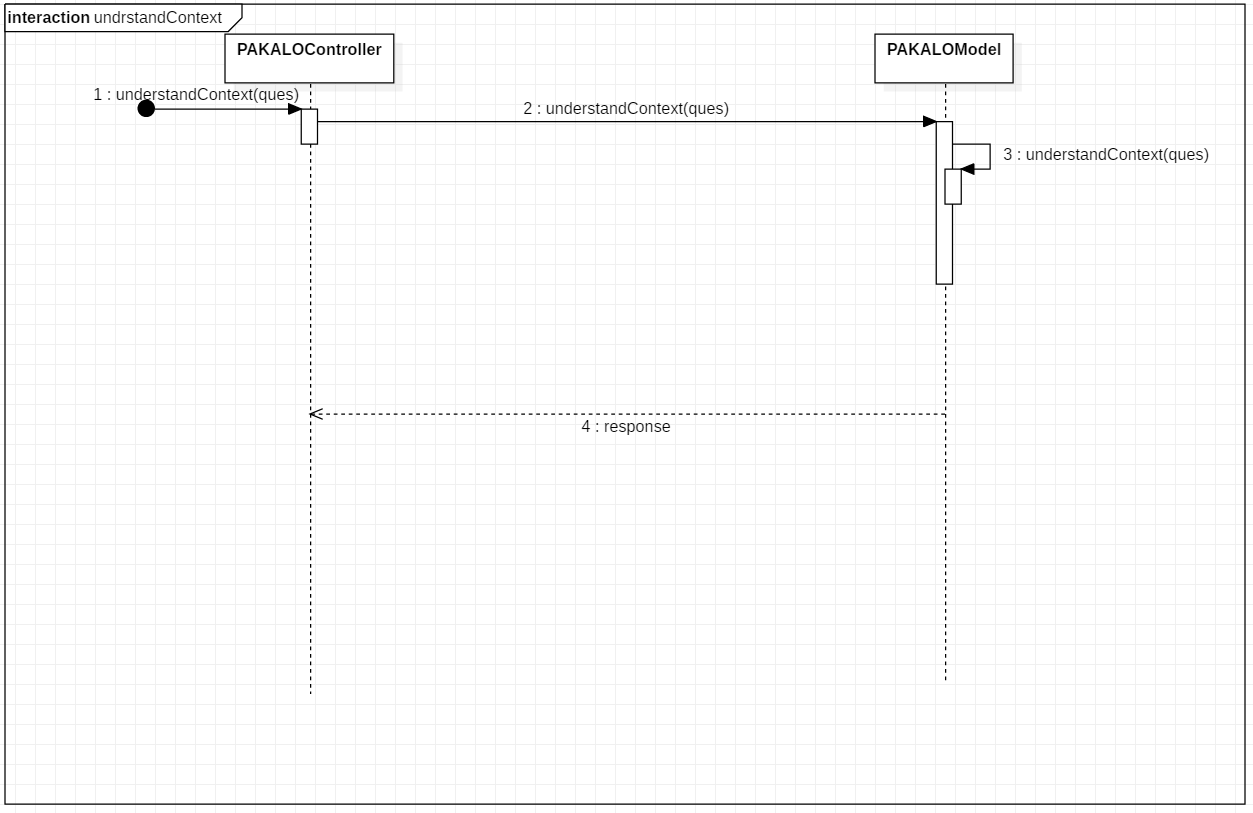
***CO-01: SearchRecipe***

******

***CO-02: AskQuestion***

******

***CO-03: UnderstandContext***

******

1. **Iteration 2**
   1. **Expanded Use Cases**

***UC-03: View Recipe***

**Scope:** Pakalo

**Level:** User Level

**Actors:** User

**Stakeholders and Interests:** Wants to view the full recipe with the specified name so that they could cook that dish easily and in an efficient way by viewing overall recipe.

**Preconditions:**

1. User has an account.
2. The user must be logged in.
3. The recipe name should exist, and should contain all recipe details.

**Success Guarantee (Post Conditions):** The User will successfully able to view the recipe in its full form.

**Main Success Scenario:**

|  |  |
| --- | --- |
| Actor Action | System Response |
| 1. The user will go to the search text box. |  |
| 1. The user will enter the name of recipe which he wants to get and view. |  |
|  | 1. System will show the list of recipes if multiple recipes with the same name exist. |
| 1. The user will choose which recipe he wants. |  |
|  | 1. System will show the complete recipe including both ingredients and instruction. |
| 1. The user will view the recipe. |  |

**Extensions**:

1. If the recipe with the specified name could not be found when a user searches for it
2. Then the system will show the message to check the recipe name.
3. If there is no spelling mistake, then the system will show the message stating the recipe could not be found.
4. The system may not respond; in that case the user may need to restart the system.

**Specific Requirements:**

1. Large text from the recipe on display. Visible from 500 cm.
2. Recipe search from database should be efficient.
3. The name of recipe provided should be valid.

**Technology and Data variation:** Data entering is through typing and/or through speech.

**Frequency of Occurrence:** Near continuous.

**Miscellaneous:** Require Internet Connection.

***UC-04: Guide Step by Step***

**Scope:** Pakalo

**Level:** User Level

**Actors:** User

**Stakeholders and Interests:** Wants the step by step guide throughout the recipe in order cook the specific dish.

**Preconditions:**

1. The user has an account.
2. The user must be logged in.
3. The question should be valid.
4. User has to have started a recipe.

**Success Guarantee (Post Conditions):** The User will successfully able to get his specified step from the Pakalo whether in text or speech.

**Main Success Scenario:**

|  |  |
| --- | --- |
| **Actor Action** | **System Response** |
| 1. The user will open System. |  |
| 1. The user will ask for guidance of specified recipe. |  |
|  | 1. System will recognize the question. |
|  | 1. The system will ask from Pakalo model for getting specified answer being asked. |
|  | 1. System show the step and/or it will tell the answer through speech. |
| 1. The User will get to the specified step. |  |

**Extensions**:

1. If the question being asked is out of context or not valid,

* 1. The system will show the message for invalid question or that it could not be understood.

2. If system stop running,

* 1. Then the user may have to reopen the system and use it.

**Specific Requirements:**

1. Large text from the system screen should be visible from 500 cm.
2. Understanding of question should be efficient
3. The question being asked should be valid.

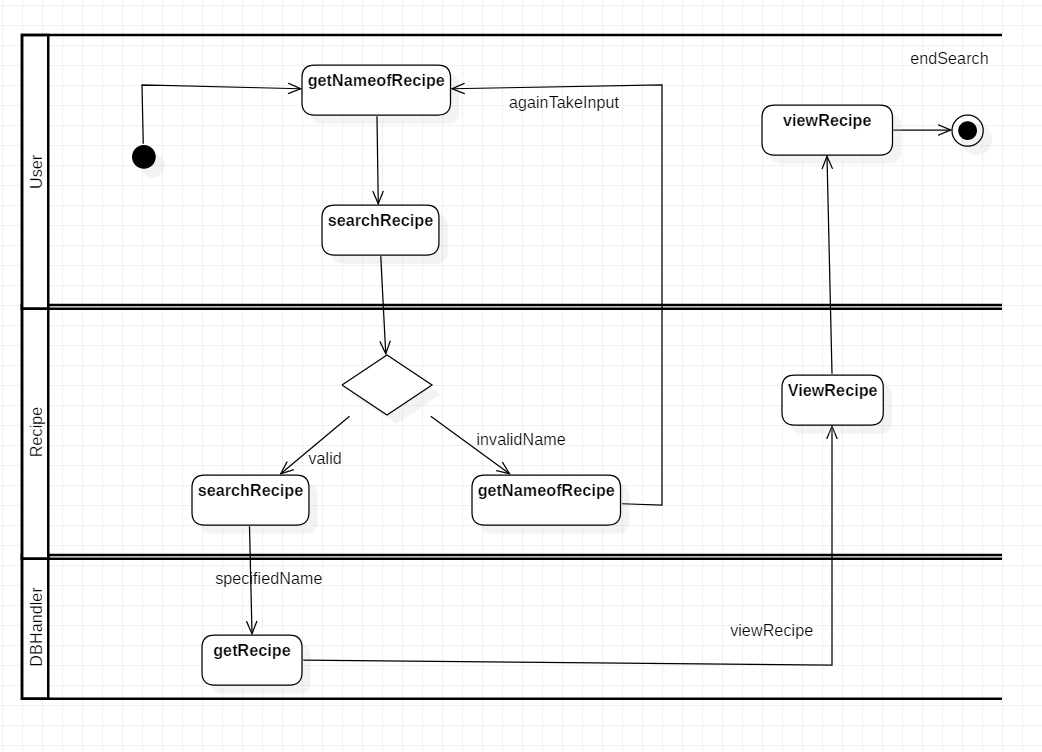
**Technology and Data variation:** Data entering is through typing and/or through speech.

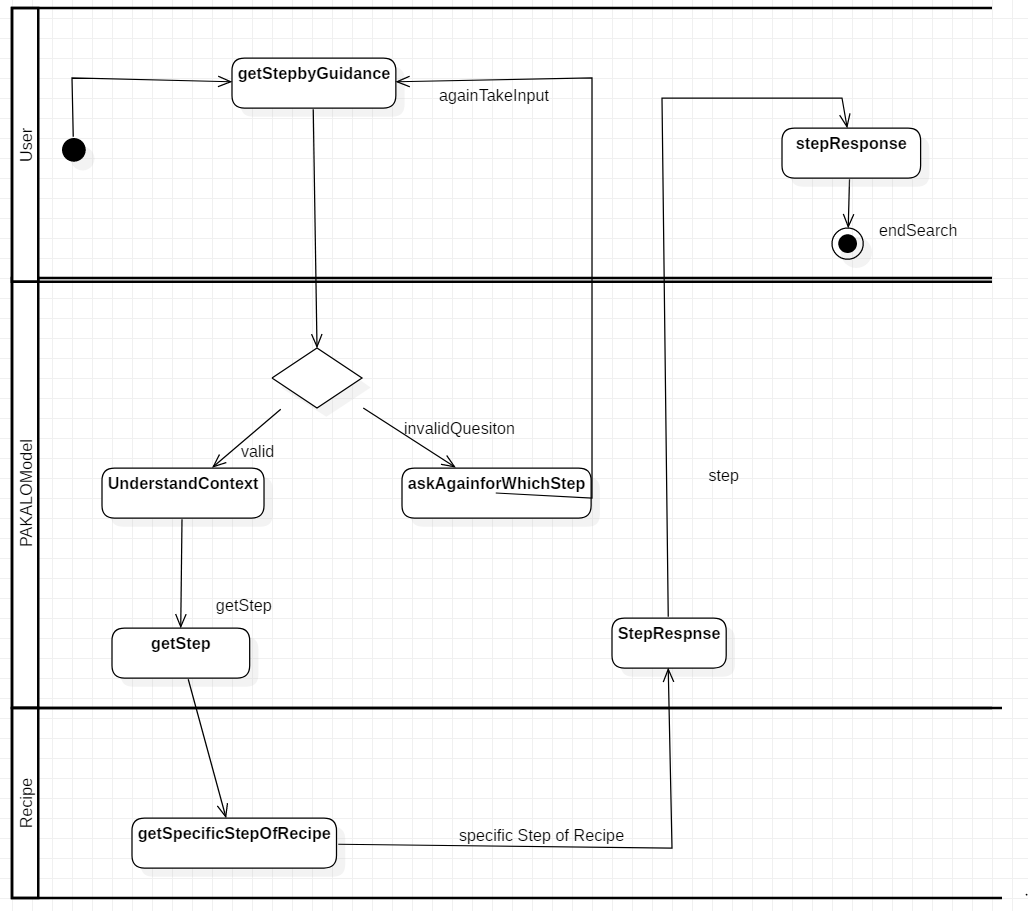
**Frequency of Occurrence:** Near continuous.

**Miscellaneous:** Require Internet Connection

* 1. **Activity Diagrams**

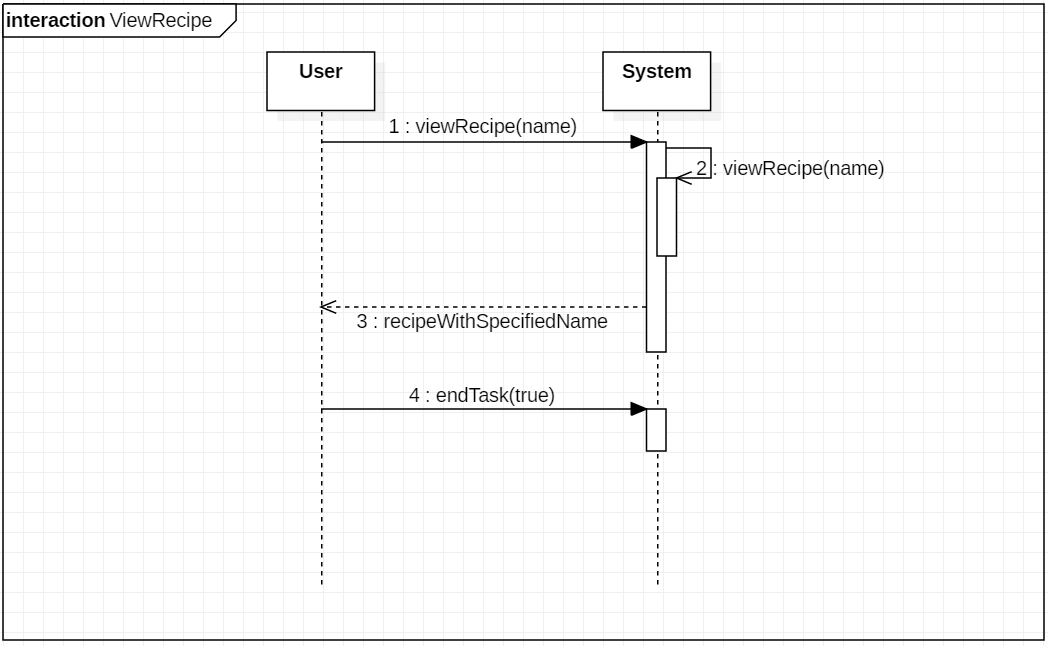
***UC-03: ViewRecipe***

******

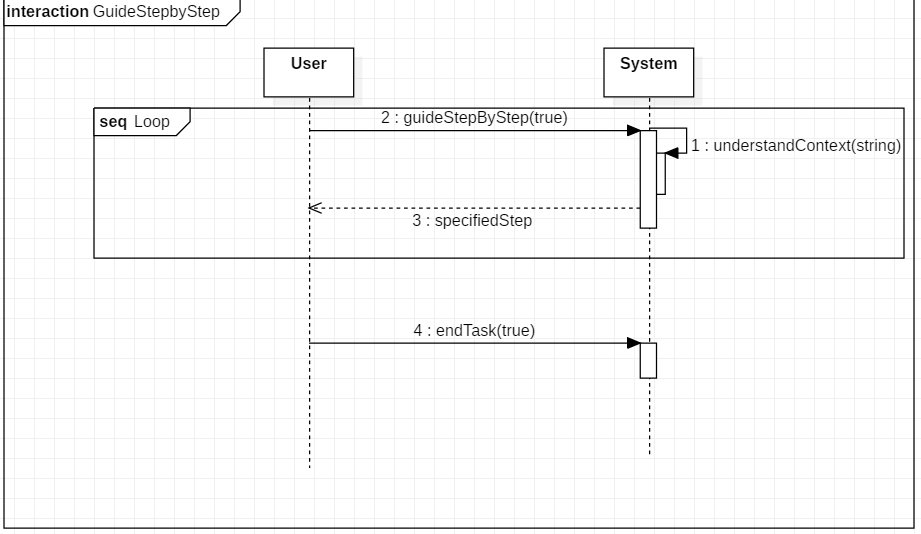
***UC-04: Guide Step by Step***

* 1. **System Sequence Diagrams**

***UC-03: View Recipe***



***UC-04: Guide Step by Step***

******

* 1. **Operation Contracts**

***Contract CO4: ViewRecipe***

**Operation:** viewRecipe(name)

**Responsibility:** Find the recipe with the specified name and displays it.

**Cross References:** View Recipe

**Preconditions:** User must be logged in.

**Post conditions:**

1. Recipe r was associated with DBHandler.
2. Attributes of r were initialized.
3. r.title became title.
4. r.ingredients became ingredients.
5. r.instructions became instruction.

***Contract CO5: guideStepByStep***

**Operation:** GuideStepByStep(name)

**Responsibility:** Guide step by step through out complete recipe with name.

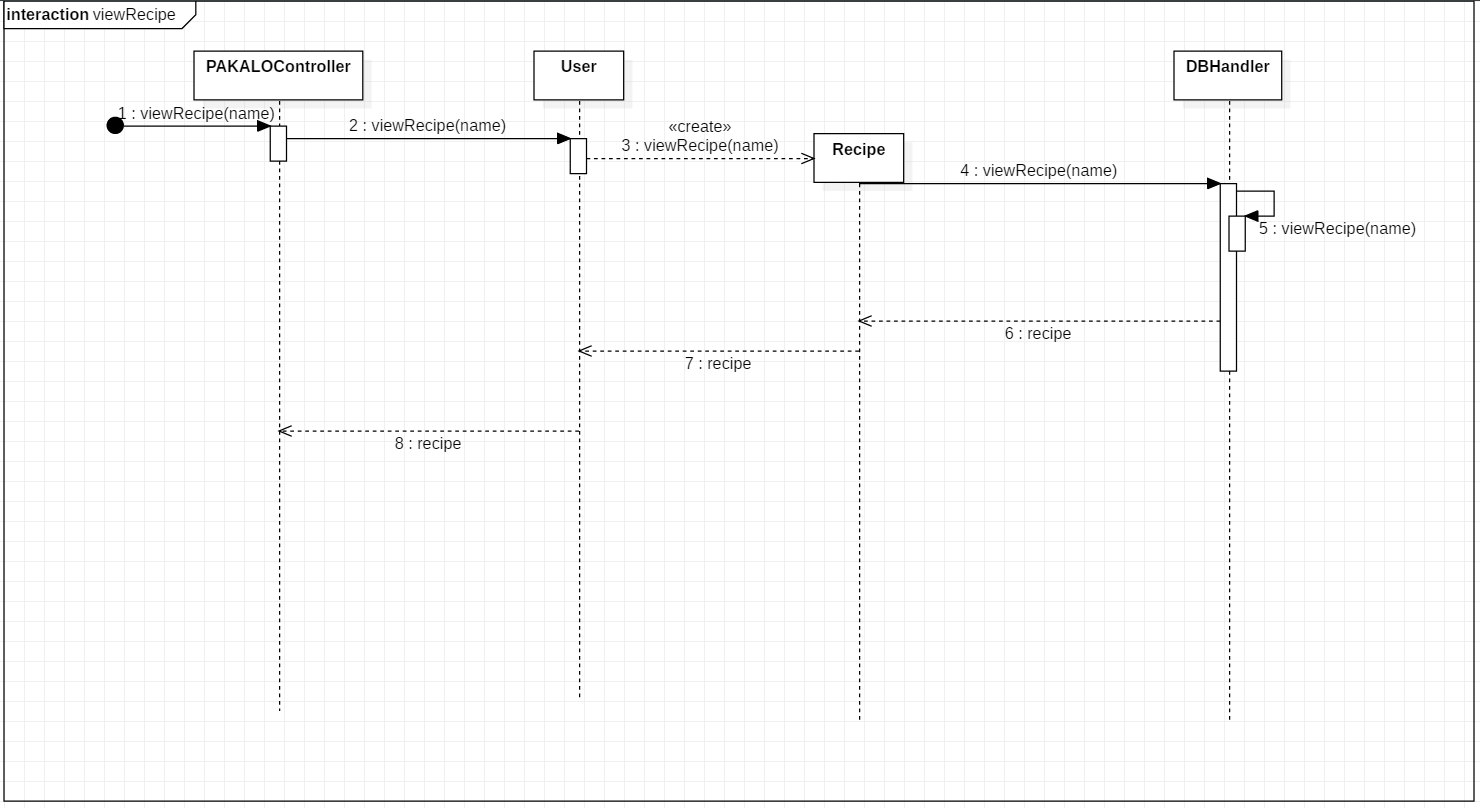
**Cross References:** guide Step by Step

**Preconditions:** User must be logged in.

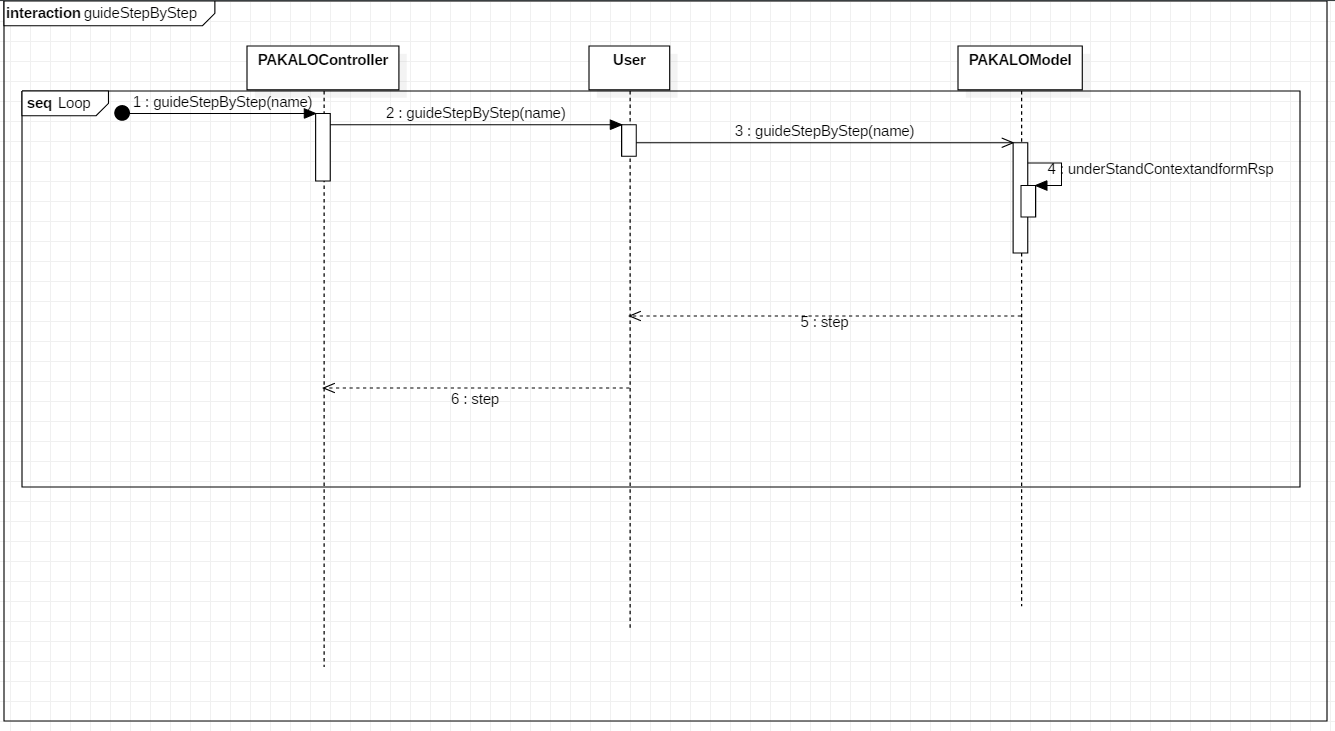
**Post conditions:**

1. A model instance m was created.
2. Model m was associated with PAKALOModel.
3. Attributes of m were initialized.
4. Recipe instance r was initialized with recipe name N.
5. Model m sends each step of recipe at user requirements.
   1. **Sequence Diagrams**

***CO-04: ViewRecipe***

******

***CO-05: guideStepByStep***

******

1. **Iteration 3 and Iteration 4**
   1. **Expanded Use Cases**

***UC-05: Recipe Recommendation based on history***

**Scope:** Pakalo

**Level:** User Level

**Actors:** User

**Stakeholders and Interests:** Wants recommended list of recipes based on the recipes that they have previously searched or cooked, and recipes that they have rated positively.

**Preconditions:**

1. The User must have unique username.
2. User have already been registered.
3. User is logged in to PAKALO.

**Success Guarantee (Post Conditions):** The User will successfully able to get and select recipes recommended by PAKALO.

**Main Success Scenario:**

|  |  |
| --- | --- |
| **Actor Action** | **System Response** |
| 1. The user will open the System. |  |
| 1. The User will click on search recipe |  |
|  | 1. System will shows the recommended recipes that the user have previously opened. |
| 1. The User can select the recipes from recommended list that are based on the history of user. |  |

**Extensions**:

1. If system stops running,
   1. Then the user may have to reopen the system and use it.

**Specific Requirements:**

1. Large text from the system screen should be visible from 500 cm.
2. Understanding of question should be efficient.
3. The question being asked should be valid.

**Technology and Data variation:** Data entering is through typing and/or through speech.

**Frequency of Occurrence:** Near continuous.

**Miscellaneous:** Require Internet Connection

***UC-08: Register User***

**Scope:** Pakalo

**Level:** User Level

**Actors:** User

**Stakeholders and Interests:** Wants to register account in order to interact with PAKALO and get guidance of specific cooking recipe.

**Preconditions:**

1. The User must have unique username.

**Success Guarantee (Post Conditions):** The User will successfully able to register account for using PAKALO.

**Main Success Scenario:**

|  |  |
| --- | --- |
| **Actor Action** | **System Response** |
| 1. The user will open System. |  |
| 1. The User will click on register account button if user doesn’t get registered by the PAKALO. |  |
|  | 1. System will recognize the register action. |
|  | 1. The System will show the digital form for registration of account. |
| 1. The User will fill the details in the form |  |
| 1. User press register or create account button |  |
|  | 1. System will register the user data and shows success popup for successful account registration |

**Extensions**:

1. If username is not unique,

* 1. The system will show the message for invalid username.

1. If system stops running,
   1. Then the user may have to reopen the system and use it.

**Specific Requirements:**

1. Large text from the system screen should be visible from 500 cm.
2. Understanding of question should be efficient.
3. The question being asked should be valid.

**Technology and Data variation:** Data entering is through typing and/or through speech.

**Frequency of Occurrence:** Near continuous.

**Miscellaneous:** Require Internet Connection

***UC-09: Login User***

**Scope:** Pakalo

**Level:** User Level

**Actors:** User

**Stakeholders and Interests:** Wants to login to the PAKALO in order to interact with PAKALO and get guidance of specific cooking recipe.

**Preconditions:**

1. The User must have unique username.
2. User have already been registered

**Success Guarantee (Post Conditions):** The User will successfully able to login and use PAKALO.

**Main Success Scenario:**

|  |  |
| --- | --- |
| **Actor Action** | **System Response** |
| 1. The user will open the System. |  |
| 1. The User will click on Login button if user have already registered by the PAKALO. |  |
|  | 1. System will recognize the login action. |
|  | 1. The System will show the digital form for entering login credentials for user. |
| 1. The User will fill the credentials in the login form. |  |
| 1. User press login button. |  |
|  | 1. System will validate the credentials and ten redirect to PAKALO Home screen. |

**Extensions**:

1. If username is not unique,

* 1. The system will show the message for invalid username.

2. If system stops running,

* 1. Then the user may have to reopen the system and use it.

**Specific Requirements:**

1. Large text from the system screen should be visible from 500 cm.
2. Understanding of question should be efficient.
3. The question being asked should be valid.

**Technology and Data variation:** Data entering is through typing and/or through speech.

**Frequency of Occurrence:** Near continuous.

**Miscellaneous:** Require Internet Connection.

***UC-11: Like Recipe***

**Scope:** Pakalo

**Level:** User Level

**Actors:** User

**Stakeholders and Interests:** Want to leave a star rating that they have deemed fit for a particular recipe.

**Preconditions:**

1. The User must have unique username.
2. User have already been registered.
3. User is logged in to PAKALO.

**Success Guarantee (Post Conditions):** The User will successfully able to like the recipe.

**Main Success Scenario:**

|  |  |
| --- | --- |
| **Actor Action** | **System Response** |
| 1. The user will open the System. |  |
| 1. The User will open the specific recipe. |  |
|  | 1. System will shows the recipe and icons for liking the recipe. |
| 1. The User will click on like icon for liking the recipe. |  |
|  | 1. The system will mark the specific recipe as liked/rated by the user. |

**Extensions**:

1. If system stops running,
   1. Then the user may have to reopen the system and use it.

**Specific Requirements:**

1. Large text from the system screen should be visible from 500 cm.
2. Understanding of question should be efficient.
3. The question being asked should be valid.

**Technology and Data variation:** Data entering is through typing and/or through speech.

**Frequency of Occurrence:** Near continuous.

**Miscellaneous:** Require Internet Connection

***UC-13: Share Recipe***

**Scope:** Pakalo

**Level:** User Level

**Actors:** User

**Stakeholders and Interests:** Wants choose to share a recipe that they really like and think other people may enjoy.

**Preconditions:**

1. The User must have unique username.

1. User have already been registered.
2. User is logged in to PAKALO.

**Success Guarantee (Post Conditions):** The User will successfully able to share the recipe.

**Main Success Scenario:**

|  |  |
| --- | --- |
| **Actor Action** | **System Response** |
| 1. The user will open the System. |  |
| 1. The User will open the specific recipe. |  |
|  | 1. System will shows the recipe and icons for sharing the recipe. |
| 1. The User will click on share icon for sharing the recipe. |  |
|  | 1. The system will mark the specific recipe as share and shows the shared url of that recipe. |

**Extensions**:

1. If system stops running,
   1. Then the user may have to reopen the system and use it.

**Specific Requirements:**

1. Large text from the system screen should be visible from 500 cm.
2. Understanding of question should be efficient.
3. The question being asked should be valid.

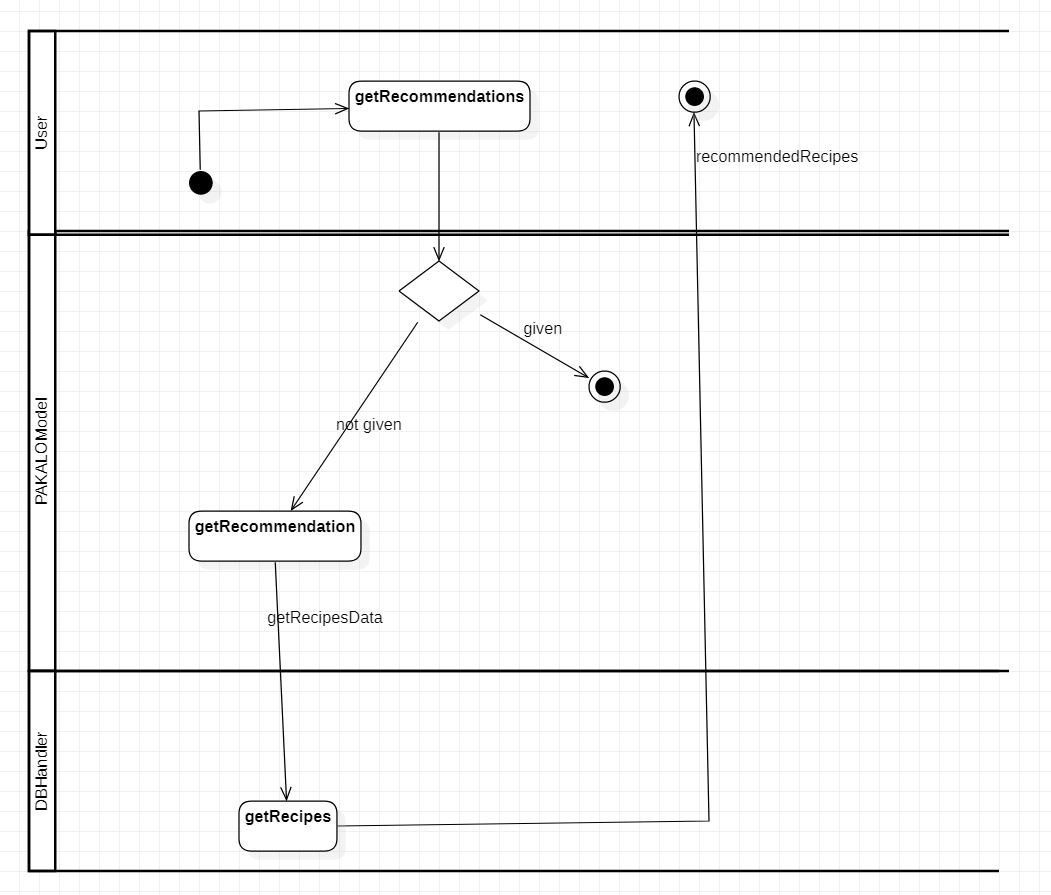
**Technology and Data variation:** Data entering is through typing and/or through speech.

**Frequency of Occurrence:** Near continuous.

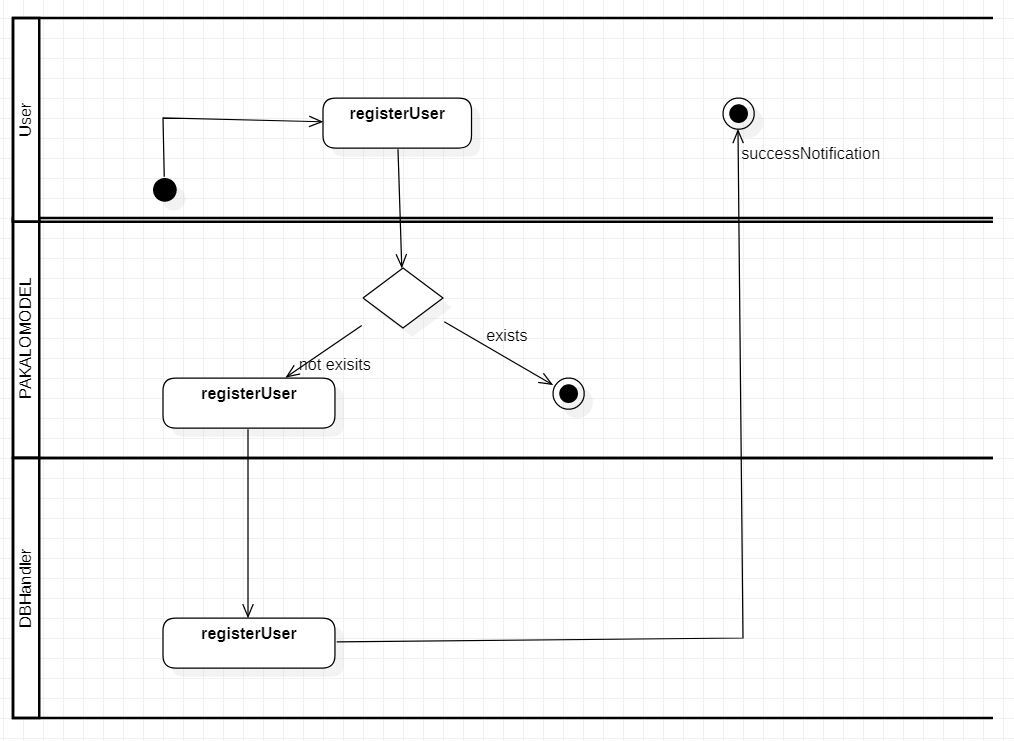
**Miscellaneous:** Require Internet Connection

* 1. **Activity Diagram**

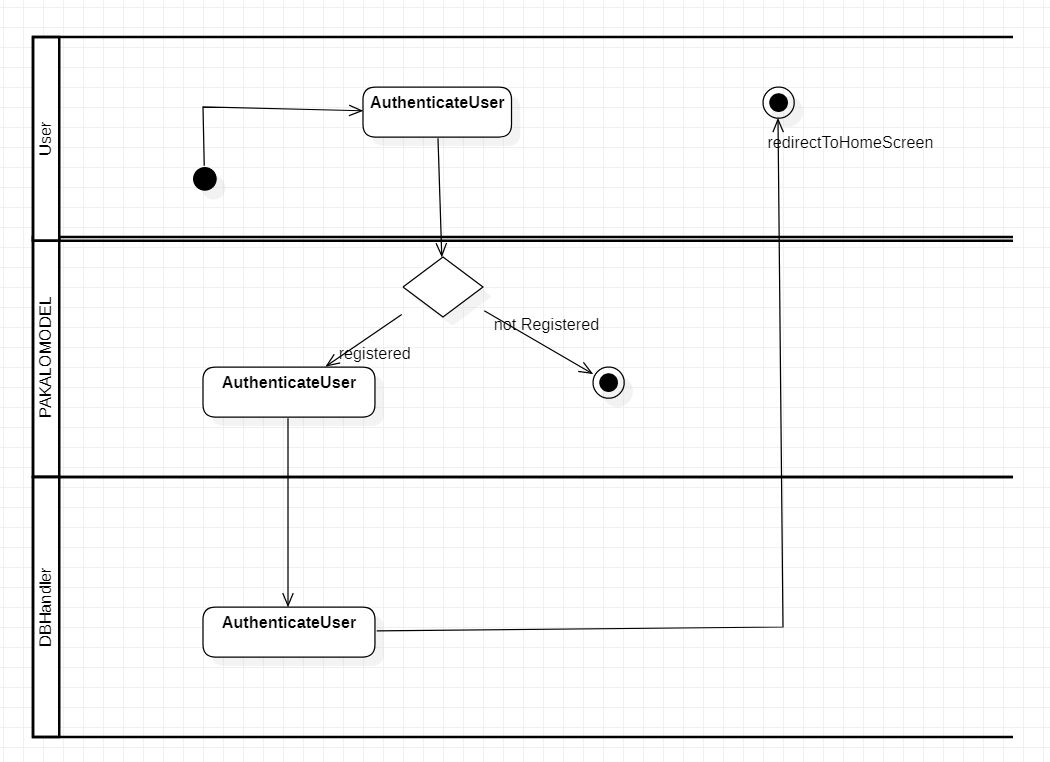
***UC-05: Recipe Recommendation based on history***

******

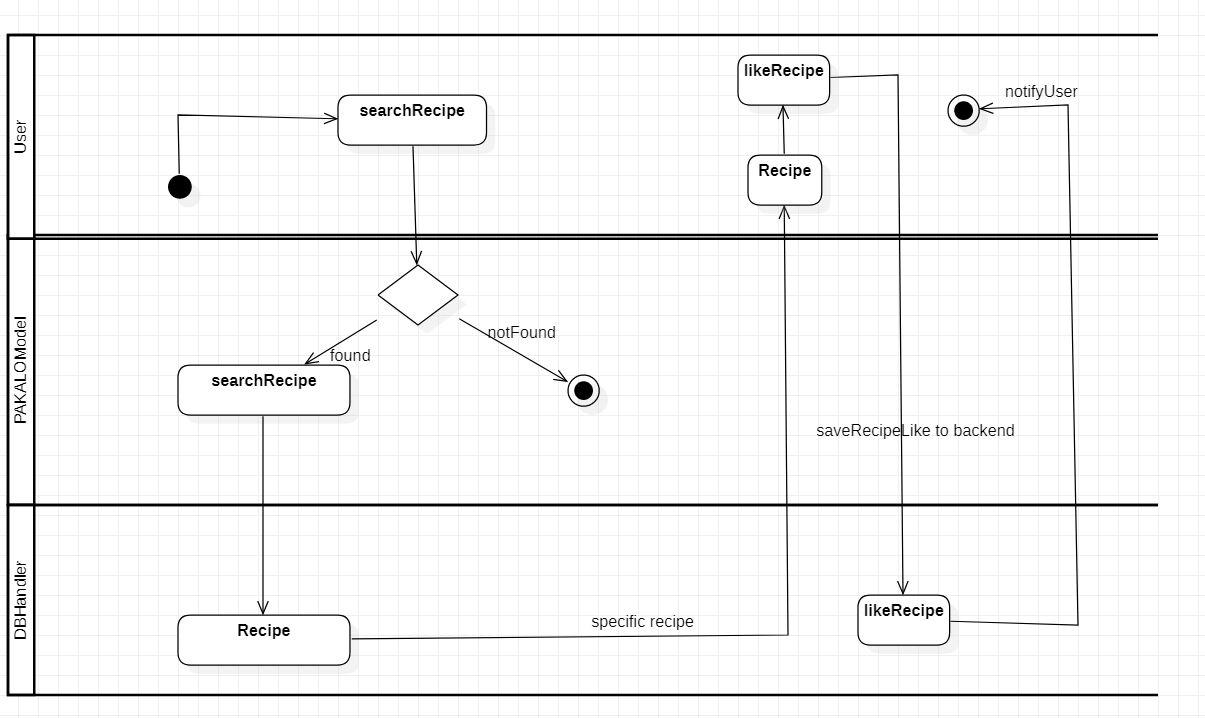
***UC-08: Register User***

******

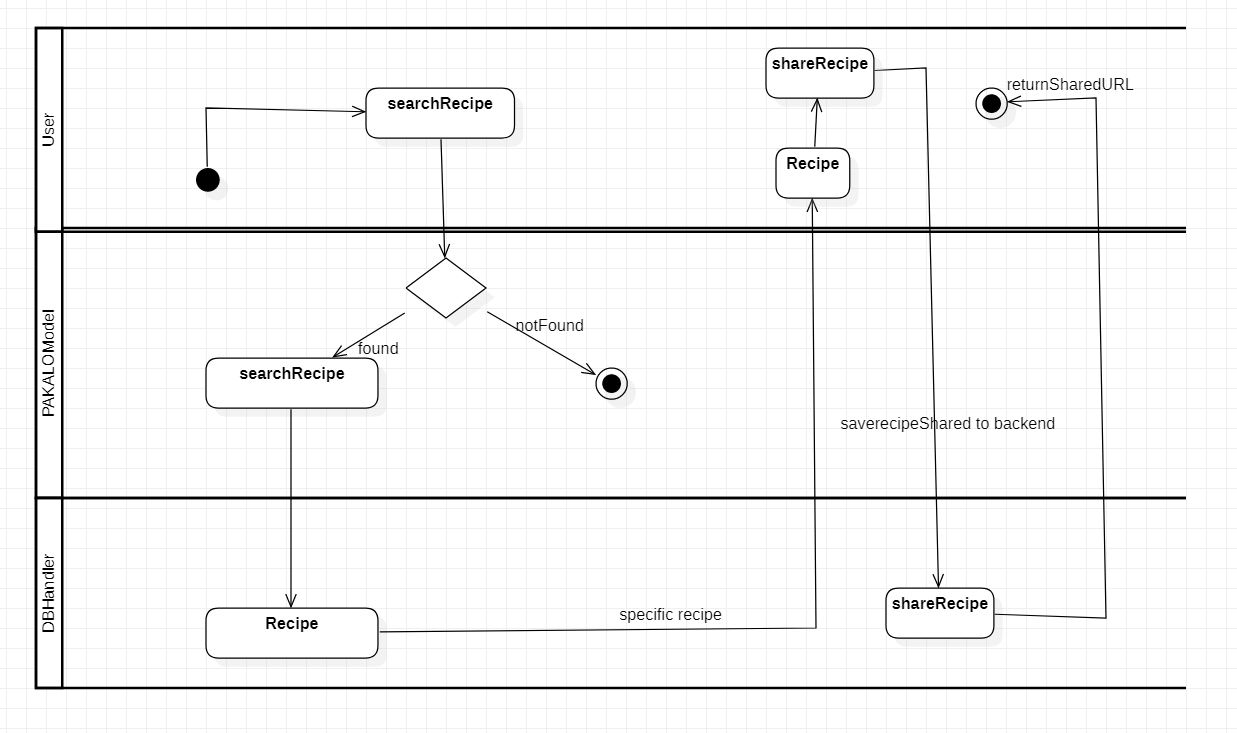
***UC-09: Login User***



***UC-11: Like Recipe***

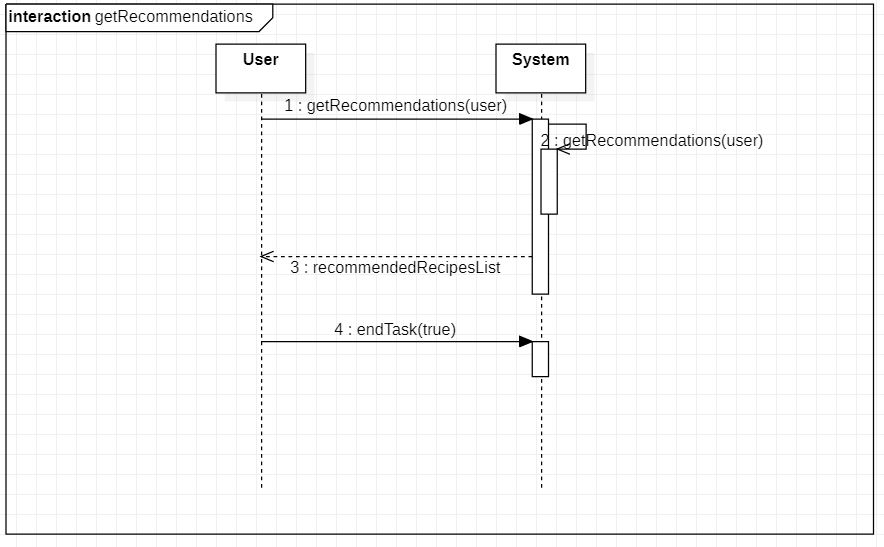


***UC-13: Share Recipe***

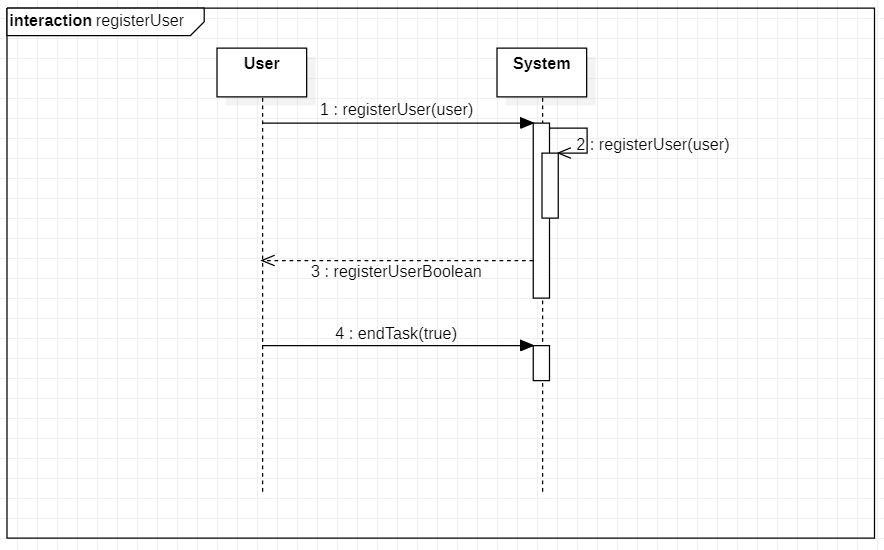
******

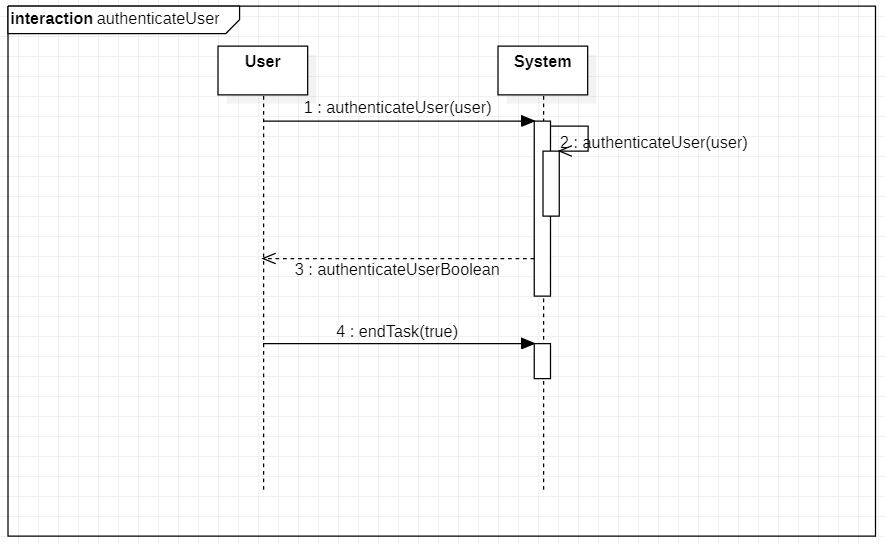
* 1. **System Sequence Diagram**

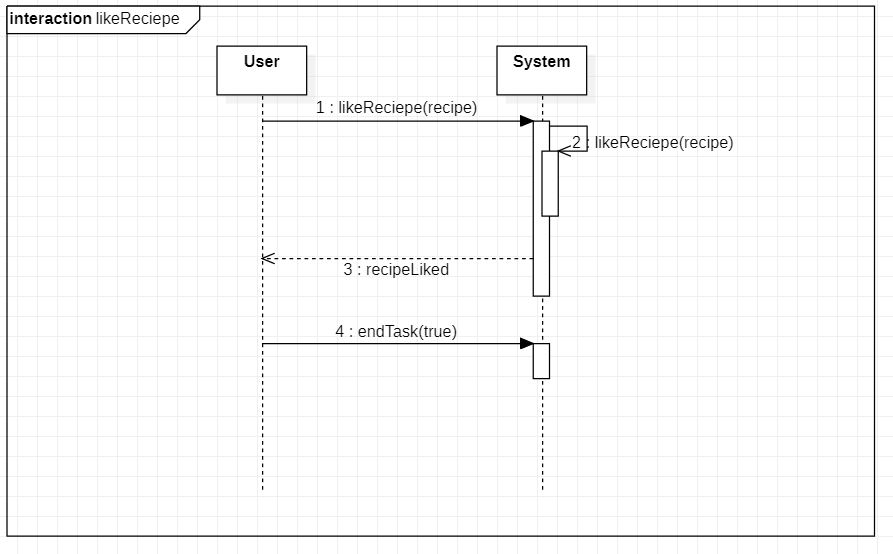
***UC-05: Recipe Recommendation based on history***

******

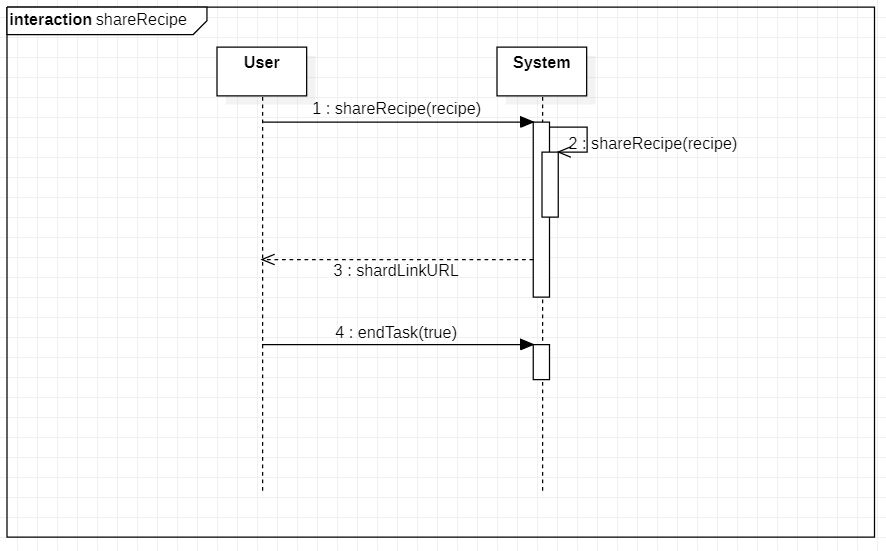
***UC-08: Register User***

**

***UC-09: Login User*****

***UC-11: Like Recipe*****

***UC-13: Share Recipe***

******

* 1. **Operation Contracts**

***Contract CO6: GetRecommendations***

**Operation:** getRecommendations(user)

**Responsibility:** Find the lists of recipes recommended for specific user and return them.

**Cross References:** Recipe Recommendation base on history

**Preconditions:** User must be logged in.

**Post conditions:**

1. The recipes instance r is created.
2. Model m was associated with PAKALOModel.
3. R is derived from m.
4. Attributes of r were initialized.
5. r.recipe became recommended recipes.
6. Every recommended recipe instance r is returned in a list of recommended recipes to User u.

***Contract CO7: RegisterUser***

**Operation:** registerUser(UserInfo)

**Responsibility:** Create a registration account for user on the basis of userInfo and returns.

**Cross References:** Register User

**Preconditions:** None.

**Post conditions:**

1. User instance u was created
2. Model m was associated with PAKALOModel.
3. Attributes of u were initialized by userInfo.
4. m saves the userInfo in DB.

***Contract CO8: AuthenticateUser***

**Operation:** authenticateUser(user)

**Responsibility:** Validate the user credentials and allow user to log into the system.

**Cross References:** Login User

**Preconditions:** User must have an account or registered by PAKALO.

**Post conditions:**

1. User instance u was created.
2. u attributes were initialized by authenticateUser parameters
3. Model m was associated with PAKALOModel.
4. m validates/check the user credentials with DB credentials

***Contract CO9: LikeRecipe***

**Operation:** likeRecipe(recipe)

**Responsibility:** mark the specified recipe as liked/rated by the user.

**Cross References:** Like Recipe

**Preconditions:** User must be logged in.

**Post conditions:**

1. Recipe r instance was associated with recipe.
2. Attributes of r were initialzed.
3. r.like become recipe.like

***Contract CO10: ShareRecipe***

**Operation:** shareRecipe(recipe)

**Responsibility:** finds the specified recipe sharedable URL link and return it.

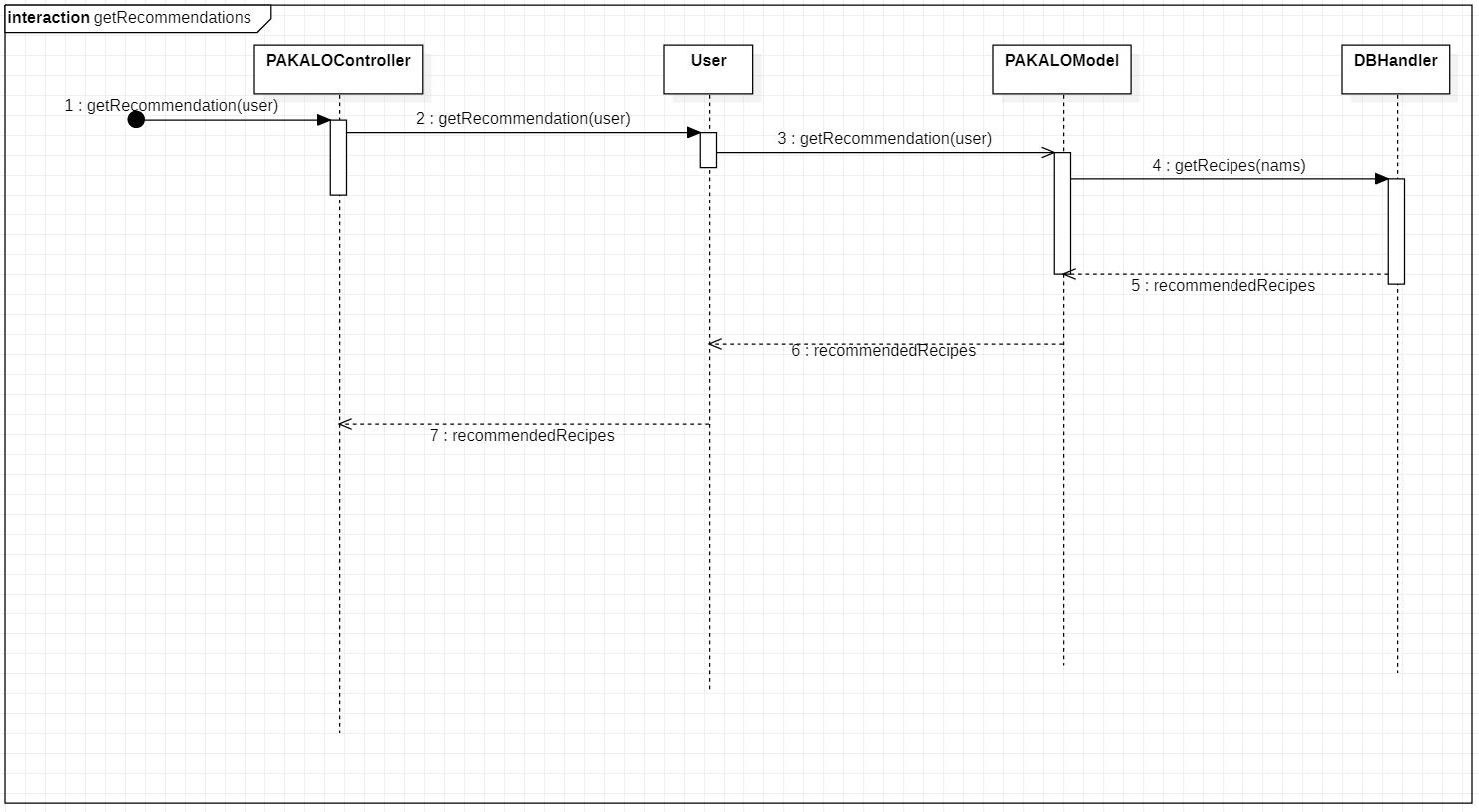
**Cross References:** Share Recipe

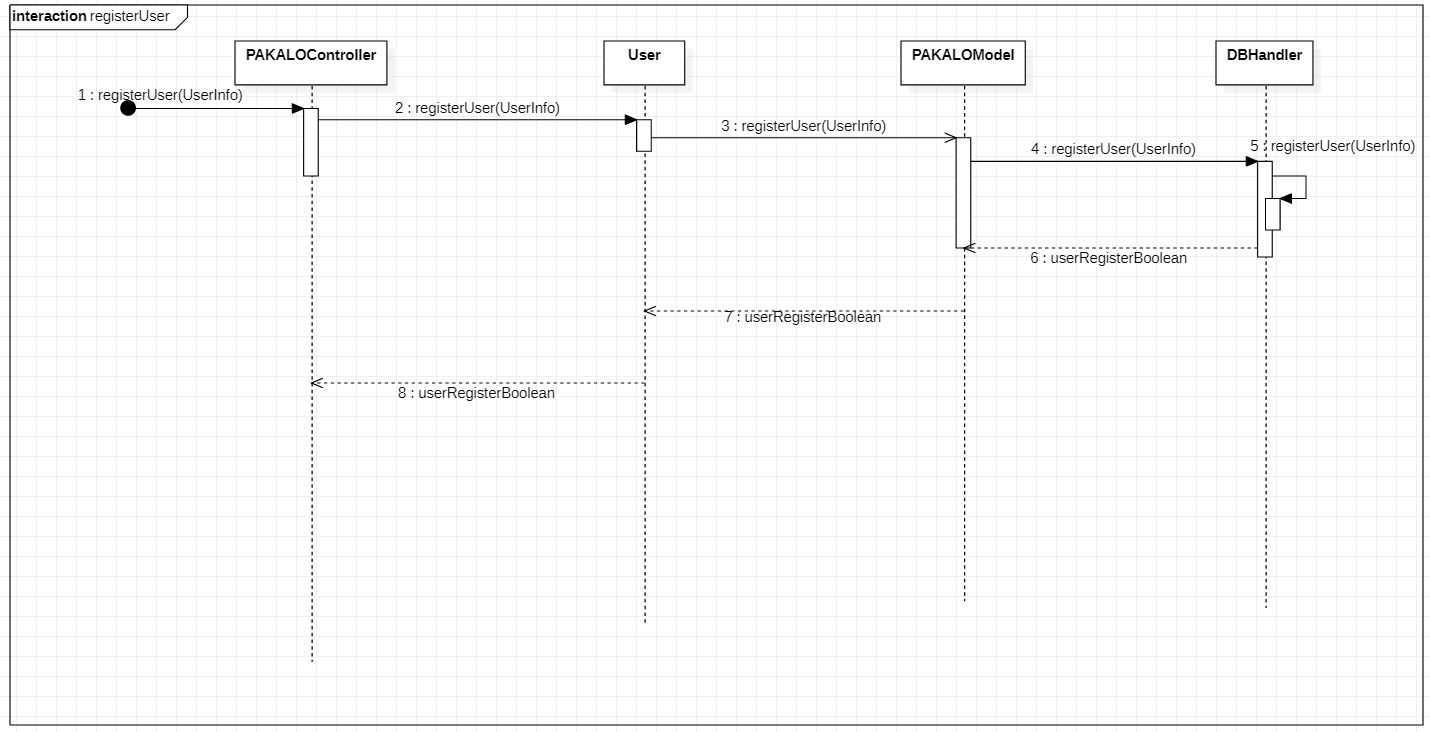
**Preconditions:** User must be logged in.

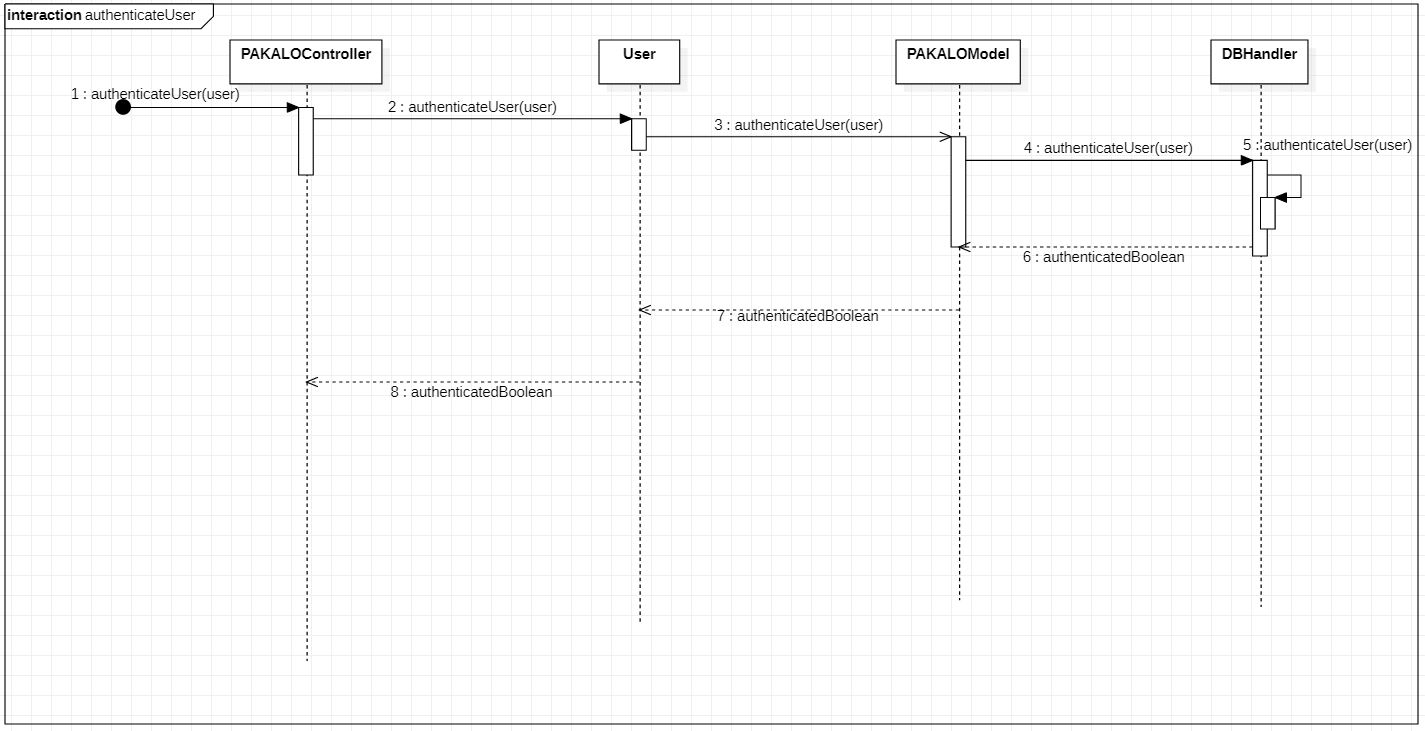
**Post conditions:**

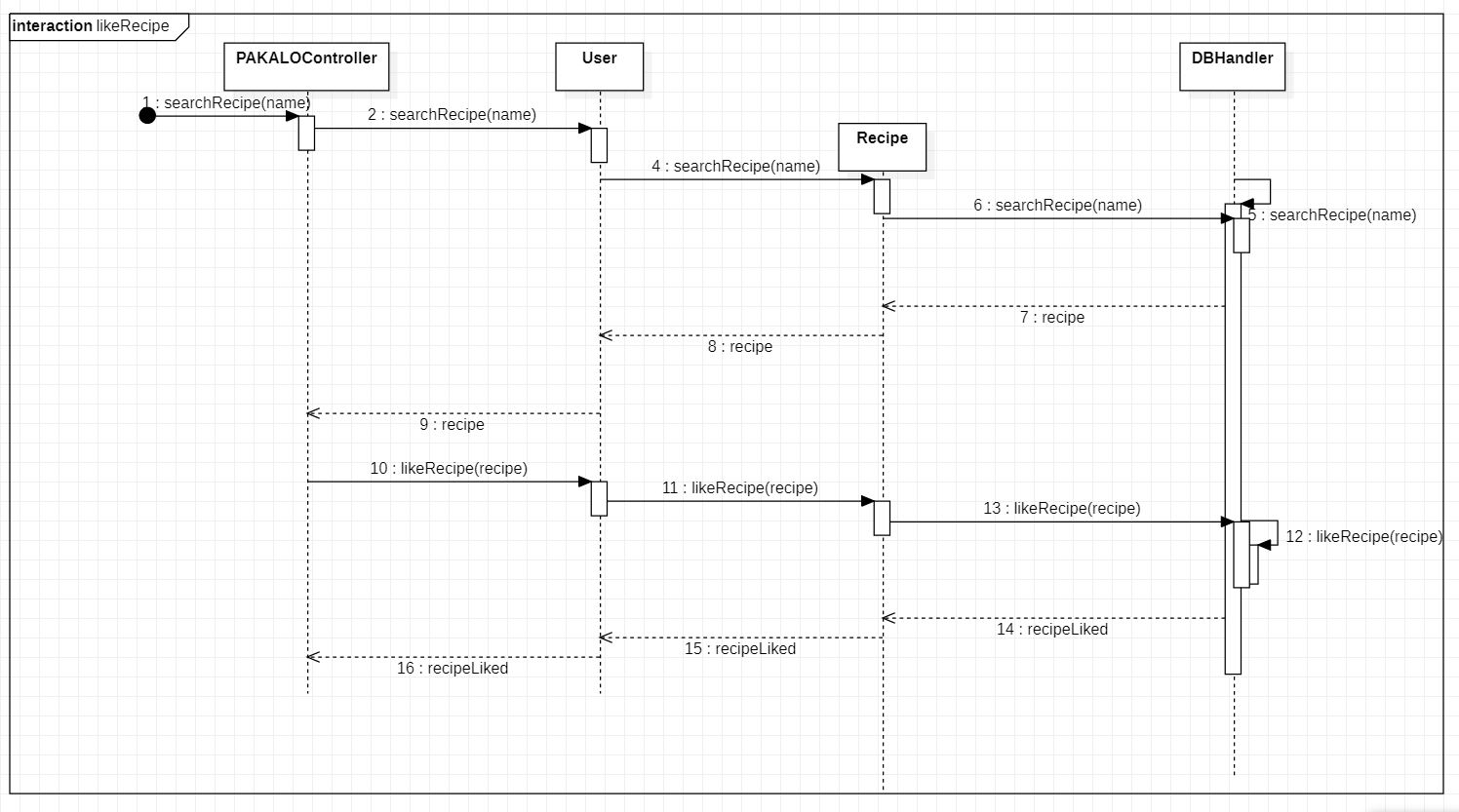
1. Recipe r instance was associated with recipe.
2. Model m was associated with PAKALOModel.
3. R attributes derived from m.
4. r.share become recipe.share
   1. **Sequence Diagrams**

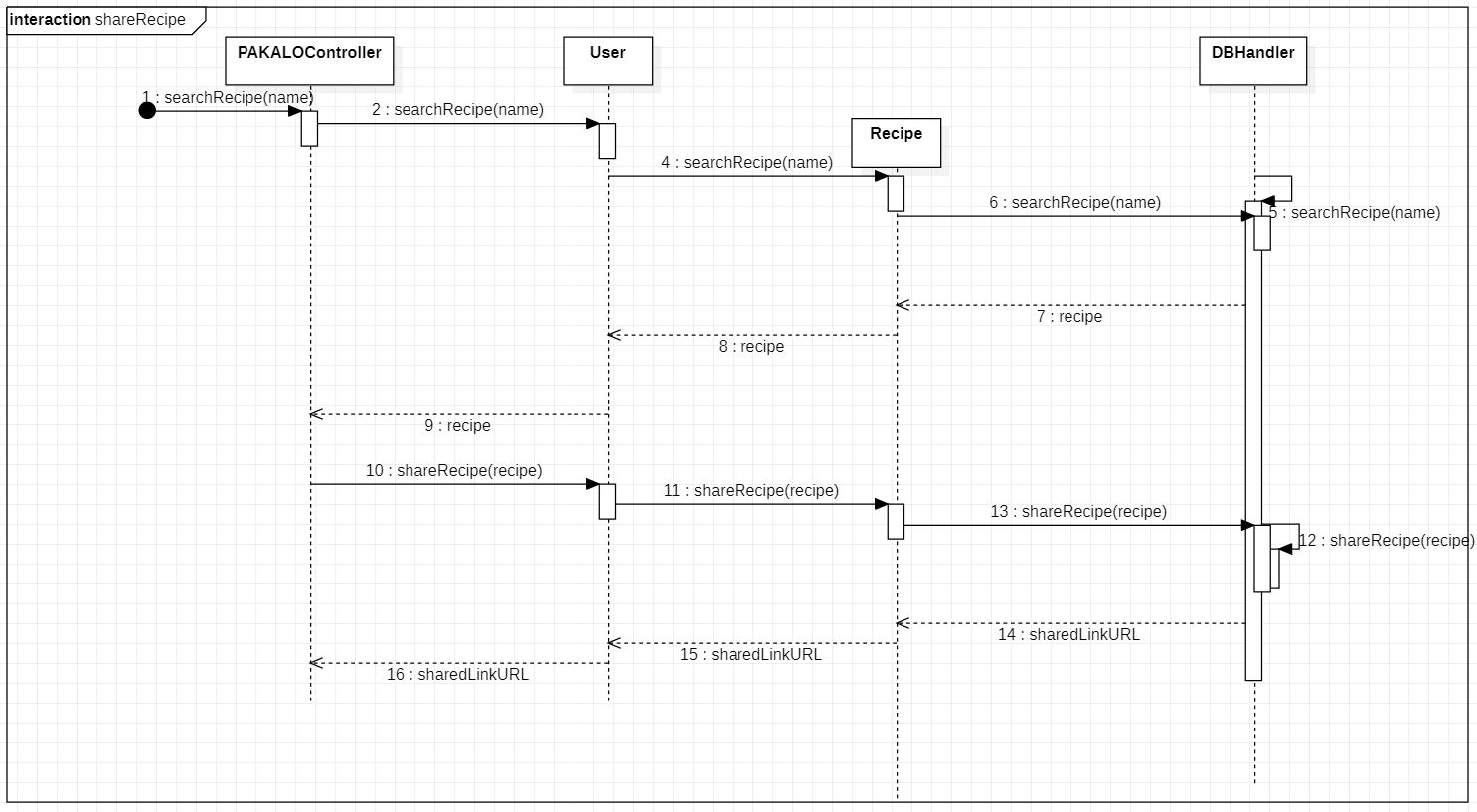
***CO-06: GetRecommendations***

******

***CO-07: RegisterUser* **

***CO-08: AuthenticateUser ***

***CO-09: LikeRecipe ***

***CO-10: ShareRecipe* **

1. **Implementation Details**

***Implementation Details for work done in Iteration 1***

* Data Gathering and Scraping
  + - * Gathered recipe data from several kaggle competitions
      * Recipe data was also scrapped from several websites.
* A Firestore database was made in Firebase.
* Preprocessing
  + - * Recipe data that was gathered was then processed and stored in a proper format.
      * The recipes were then uploaded to the noSQL Firestore.
      * The total amount of recipe data acquired is close to 1.1 million.
* Categorization
  + - * A food group categorization structure was found from the internet.
      * The ingredients within our recipe data was then categorized accordingly. For example, milk and cheese would be categorized as Dairy.

***Implementation Details done for work done in Iteration 2***

* As Firestore does not allow native indexing or searching for text fields in documents within the Firestore database, Algolia (a third party app) was integrated in cloud Firestore to allow searching.
* Firebase cloud functions were written which will automatically manipulate data in Algolia when Firestore is updated.
* Model Formation and Model Training
  + - * Custom training data used in training model for the chatbot was made, specific to our domain.
      * Training data was formatted according to the appropriate requirements i.e. JSON and MD format.
      * Keras was used along with LSTM and embedding layers like Word2vec were used to make a chatbot model. The architecture can be summarized as follows:
        + Input data encoder
        + Word embedding (Word2vec from the Keras library)
        + 2 layers of LSTM
        + Output decoder
      * An additional model was formed using RASA NLU framework.
      * The pretrained English Spacy model was utilized for RASA.
      * spaCy’s supervised embeddings pipeline was used for the RASA chatbot
      * Training data, such as stories and intent examples were continuously added and removed, as experimentation was done.
* Model Result Ranking
  + - * Chatbot models were compared based on heuristics such as accuracy of intent recognition, ability to accurately recognize entities, perform appropriate actions for intents, along with a qualitative analysis of the chatbots flow in conversation.
      * Model formed in RASA was chosen as it allowed a more accurate model, due to the pipeline offered by spaCy’s English model.
      * The RASA model also allowed simpler control of slots, and actions, that were required for the chatbot.
* Integrated Speech to Text and Text to Speech in the chatbot.

***Development Challenges***

* Cloud Firestore does not support native indexing or search for text fields in documents. Additionally, downloading an entire collection to search for fields client-side is not practical.
* Data on Algolia had to be updated every time Firebase is updated.
* There is no question and answering training data available for training the chatbot specific to our domain.
* RASA framework was overhauled massively rendering much of the documentation online obsolete.

***Implementation Details done for work done in Iteration 3 and Iteration 4***

* We built an android app with following functionalities
  + Signup
  + Login
  + Searching recipes
  + Viewing recipes
  + Liking recipes
  + Integrating Rasa chatbot in android through api
  + Integrating speech to text and text to speech in android for communication with chatbot
* We improved the rasa chatbot model by implementing more intents and stories so that the chatbot can identify more intents of user.

***Development Challenges***

* As firestore is a relatively new database so it does not have many data manipulation functions. For every manipulation we had to search half hour just to find the best way to access or manipulate data in firestore.
* It was really hard to integrate rasa and android through api because RASA is a relatively new chatbot framework. It has helpful libraries but documentation on those libraries is very limited and Rasa community is really small so not much support is found online.
* Android speech to text was integrated in the last iteration but it did not understand the user well and our user would more probably in noisy places like kitchen so it would be hard to communication to chatbot. So we used Google speech to text api. But there is no library that simplifies the conversion of speech to text so implementing google speech to text was a real challenge.
* Speech to text and text to speech are both run on threads. And on top of that android life cycle makes the thread more difficult to manage so it was really difficult to manage speech to text and text to speech threads. (This problem was also faced by some people online and they didn’t solve this).
* In the previous iteration we used AIMYBOX which is an android app in which the app was able to communicate with RASA online. We tried really hard and spent a lot of time on that app but the code was really bad and we ended up wasting our time. There was no documentation through which we could figure out how the communication was taking place. So in the end we spent a lot of time figuring this out that how was communication between RASA and android was possible.

**References**

1. S. Bird, E. Klein, and E. Loper, Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit. O’Reilly Media, Inc. 2009.
2. Daniel Jurafsky and James H. Martin, Speech and Language Processing, Pearson Education India, 2000.
3. Shawar, Bayan Abu, and Eric Atwell. Chatbots: are they really useful? Ldv forum. Vol. 22. No. 1. 2007.
4. “Recipe1M : A Dataset for Learning Cross-Modal Embeddings for Cooking Recipes and Food Images,” MIT. [Online]. Available: <http://pic2recipe.csail.mit.edu/>
5. Swayam Mittal, Deep Learning Techniques for Text Classification, medium.com, August 17, 2019, [Online]. Available: <https://medium.com/datadriveninvestor/deep-learning-techniques-for-text-classification-9392ca9492c7>
6. Firebase Documentation [Online]. Available: <https://firebase.google.com/docs>
7. Kayla Matthews, The 10 Best Cooking and Dining Chatbots for Foodies, [Online]. Available: <https://chatbotsmagazine.com/the-10-best-cooking-and-dining-chatbots-for-foodies-da1af9bfec0a>
8. “Searching Firebase databases,” Algolia. [Online]. Available: <https://www.algolia.com/doc/guides/sending-and-managing-data/send-and-update-your-data/tutorials/firebase-algolia/>
9. “Tutorials,” Rasa Blog, 24-Jan-2019. [Online]. Available: <https://blog.rasa.com/tag/tutorials/>.
10. J. Petraityt, “RASA Masterclass,” YouTube. [Online]. Available: <https://www.youtube.com/watch?v=rlAQWbhwqLA&list=PL75e0qA87dlHQny7z43NduZHPo6qd-cRc>.

**Appendices**

* Natural Language Programming is the process of understanding context and meaning from language, mostly in the form of text. In its essence, NLP is used by a computer to understand language.
* Algolia is hosted search API for the Firestore that allows for us to get real time search results from the Firebase database.
* Cloud Firestore is a noSQL database service provided by Firebase and Google.
* Rasa NLU is an open-source natural language processing tool used in classification of intents, retrieval of responses and identification of entities for chatbots.
* spaCy is a free open-source library for natural language processing.
* Word2vec is a group of models that are used to produce word embeddings.
* LSTM stands for Long Short Term Memory which is a recurrent neural network used in deep learning.