Spotifood

Software design document

By

Mr. Kittinut Saengsri 592115003

Mr. Sirisak Wongwal 592115031

Department of Software Engineering

College of Arts, Media, and Technology

Chiang Mai University

Project Advisor

Dr. Passakorn Phannachitta

Document History

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Document name | Version | Status | Date | Viewable | Editable | Responsible |
| Spotifood\_SoftwareDesign\_v0.1 | - Add Chapter one  - Add chapter two  - Add chapter three  - Add chapter four  - Add chapter five | Draft | 07/10/2019 | SW, KS, PP | SW, KS | SW, KS |
| Spotifood\_SoftwareDesign\_v1.0 | - Add Chapter one  - Add chapter two  - Add chapter three  - Add chapter four  - Add chapter five | Draft | 07/29/2019 | SW, KS, PP | SW, KS | SW, KS |

\*SW = Sirisak Wongwal

\*KS = Kittinut Saengsri

\*PP = Dr.Passakorn Phannachitta

Table of Contents

[Chapter 1 | Introduction 4](#_Toc15336248)

[1.1 Identification 4](#_Toc15336249)

[1.2 Project Overview 4](#_Toc15336250)

[1.2.1 Purpose 4](#_Toc15336251)

[1.2.2 Scope 4](#_Toc15336252)

[1.3 Work Product to be developed 5](#_Toc15336253)

[1.3.1 Deliverable 5](#_Toc15336254)

[1.4 Acronyms and Definitions 5](#_Toc15336255)

[1.4.1 Acronyms 5](#_Toc15336256)

[CHAPTER TWO | SOFTWARE ARCHITECTURE 6](#_Toc15336257)

[CHAPTER THREE | DETAIL DESIGN 7](#_Toc15336258)

[3.1 CLASS DIAGRAM 7](#_Toc15336259)

[3.2 CLASS DIAGRAM DESCRIPTION 8](#_Toc15336260)

[3.2.1 Spotifood Application Class Description 8](#_Toc15336261)

[Chapter Four| Sequence Diagram 21](#_Toc15336262)

[Chapter Five| User Interface 28](#_Toc15336263)

# Chapter 1 | Introduction

## 1.1 Identification

This Project Management Plan is the document for planning, scheduling activities and evaluating overall of the project so that it could complete successfully despite all the risks. It provides detailed scheduled, assigned task and identified risks of the project.

## 1.2 Project Overview

The Spotifood application will guide people find their new dish and help people who are interested in cooking and people who have cooking skill by suggesting the dishes from a word that people search. The AI system will suggest a dish from the personality of people. For example, The people regularly search steamed fish with herbs dish. The AI system will analyze the ingredient tag of dish that is related to the healthy tag. Then, the AI will suggest the people in the type of healthy dishes.

The suggest dish of AI will help those people found new dishes and challenging in cooking every day that makes people not disconnected and bored to cooking. Moreover, we will be a middleman that brings trader of ingredient to people who can’t find the ingredients in any food. The Spotifood help people to buy an ingredient. And the Spotifood will suggest a dish that is similar to personality or suggests from the behavior of people.

### 1.2.1 Purpose

The main purpose of developing the Spotifood is to help the people who are interested in cooking and who already have skills not bored with cooking or disconnected to prepare the food.

### 1.2.2 Scope

There are six major parts in Spotifood.

Feature #1: Authentication

Feature #2: Content CRUD

Feature #3: User management

Feature #4: Similar dishes identification

Feature #5: Personalization

Feature #6: ADS ingredient

## 1.3 Work Product to be developed

### 1.3.1 Deliverable

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Deliverable/Release | Media | Copies | Date |
| 1 | Project proposal  - Spotifood\_proposal3 | Hard copy | 3 | 19/06/2019 |
| 2 | Progress report I  - Project management plan version 1.0  - Software requirement specification version 1.0  - Software design document version 1.0  - Test plan version 1.0  - Traceability record version 1.0  - Software version 0.1 | Hard copy | 3 | 27/07/2019 |

## 1.4 Acronyms and Definitions

### 1.4.1 Acronyms

UC Use case

URS User Requirement Specification

SRS Software Requirement Specification

CRUD Create-Read-Update-Delete

AI Artificial Intelligence

ADS Advertising

ID Identification Data

# CHAPTER TWO | SOFTWARE ARCHITECTURE

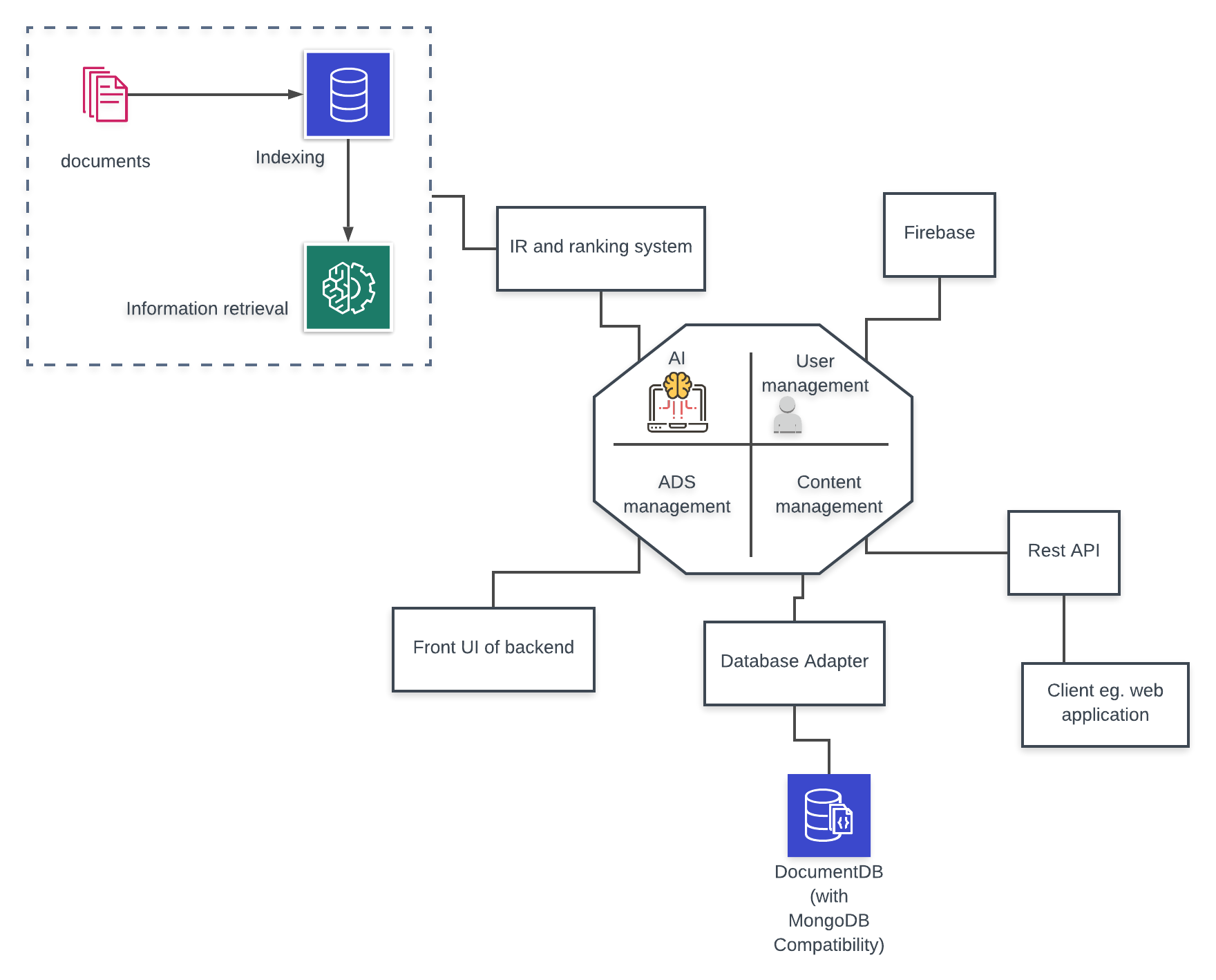


Figure 02: Spotifood System Architecture Diagram

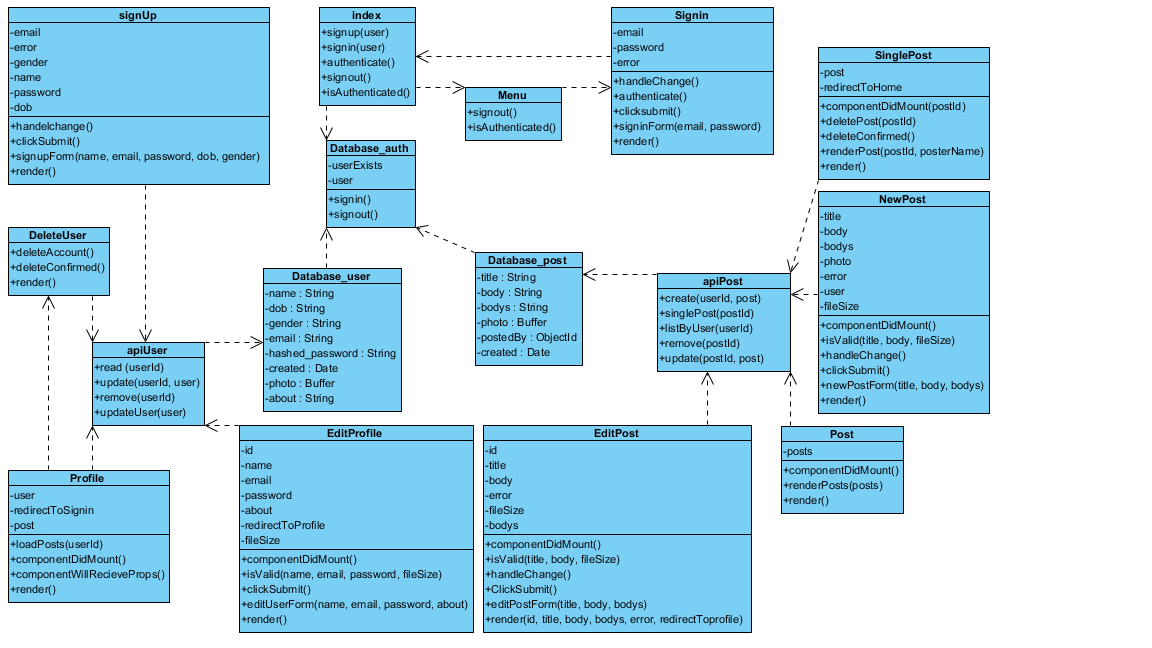
There are four subsystem of system architecture

1. AI (Artificial intelligence)
2. User management
3. ADS management (Advertisement management)
4. Content management

Firstly content management , The user interact with the system through web application developed using React, Node.js, And backend service of spotifood system using MongoDB. Secondly user management, Is the system backend for user authentication is handled with Firebase services because We use to collect only username and password. Thirdly ADS management , Is the font UI for admin to interact with the system through web application developed using React, Node.js. Finally AI or artificial intelligence , This part is the search for items that are stored in the form that is most relevant to the person from their search using IR ( information retrieval) to incidence matrix and inverted index

# CHAPTER THREE | DETAIL DESIGN

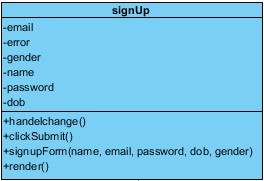
## 3.1 CLASS DIAGRAM



## 3.2 CLASS DIAGRAM DESCRIPTION

### 3.2.1 Spotifood Application Class Description

**CD-01 : signUp**



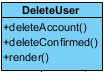
**Attributes**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | email | Email of the user. | String |
| 2 | error | Contain message of each error parameter. | String |
| 3 | gender | Gender of the user. | String |
| 4 | name | Name of the user. | String |
| 5 | password | Password of the user. | String |
| 6 | dob | Date of birth of the user. | String |

**Method**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | handleChange | Set state of error message and set event from user name value. | function(event) |
| 2 | clickSubmit | Send all parameter from signupForm to the database. | function(event) |
| 3 | signupForm | Display signup form to the user for handling input. | Return name , email , password , dob , gender |
| 4 | render | Alert error and success message to the user. | Return error message and success message |

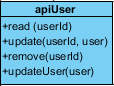
**CD-02 : DeleteUser**



**Method:**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | deleteAccount | Delete account the user. | function(event) |
| 2 | deleteConfirmed | Alert confirm box to the user before delete. | function(event) |
| 3 | render | Display the delete profile button. | Return Delete profile button |

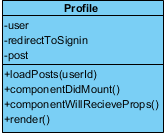
**CD-03 : apiUser**



**Method**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | read | Get the user json file from database by user id. | Param: userId  Return: - |
| 2 | update | Put or update the user json file to the database by user id. | Param: userId  Return: - |
| 3 | remove | Delete the user json file from the database by user id. | Param: userId  Return: - |
| 4 | updateUser | Put or update the user json file to the database and show old information by user id. |  |

**CD-04 : Profile**



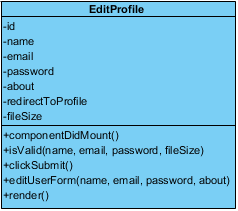
**Attributes**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | user | Data of the user from database. | String |
| 2 | redirectToSignin | Use to redirect to sign-in page. | boolean |
| 3 | post | Data of the post from user. | Array |

**Method**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | loadPosts | Display post from user. | Param: userId  Return: posts.data |
| 2 | componentDidMount | Receive post of the user from user id. | function(event) |
| 3 | componentWillReceiveProps | Receive all post of the user from user id. | function(event) |
| 4 | render | Display information of user and allow user to click create post button and edit profile button. | Return: user , posts. |

**CD-05 : EditProfile**



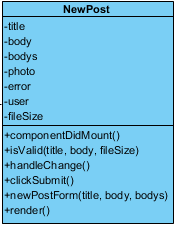
**Attributes**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | id | Unique identifier of the stadium. | String |
| 2 | name | Name of the user. | String |
| 3 | password | Password of the user | String |
| 4 | about | About user can handling in about field. | String |
| 5 | redirectToProfile | Redirect to profile page. | boolean |
| 6 | fileSize | File size of photo of the user. | int |

**Method**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | isValid | Verify all format of parameters and send the error message to render method. | Param: name, email , password, fileSize  Return: - |
| 2 | componentDidMount | Receive post of the user from user id. | function(event) |
| 3 | clickSubmit | Put the information of user update to the database. | function(event) |
| 4 | editUserForm | Display information of user and allow user to handling edit information. | Param: name, email,  Password, fileSize  Return: - |
| 5 | render | Display information of user and allow user to create post band edit profile. | Return: name, email , password , about |

**CD-06 : NewPost**



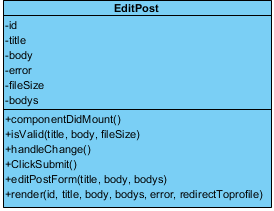
**Attributes**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | photo | Unique identifier of the stadium. | String |
| 2 | title | Name of the dish. | String |
| 3 | body | Ingredient of the dish. | String |
| 4 | bodys | Step of cook of the dish. | String |
| 5 | error | Contain message of each error parameter. | String |
| 6 | fileSize | File size of thumbnail photo of the post. | int |
| 7 | user | Information of user in database | Array |

**Method**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | isValid | Verify all format of parameters and send the error message to render method. | Param: title, body, fileSize  Return: - |
| 2 | handleChange | Receive post of the user from user id. | function(event) |
| 3 | clickSubmit | Create the information of post to the database. | function(event) |
| 4 | newPostForm | Display information of post and allow user to handling create information. | Param: title, body, bodys  Return: - |
| 5 | render | Display information of create a new post. | Return: title, body |

**CD-07 : EditPost**



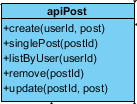
**Attributes**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | id | Unique identifier of the stadium. | String |
| 2 | title | Name of the dish. | String |
| 3 | body | Ingredient of the dish. | String |
| 4 | bodys | Step of cook of the dish. | String |
| 5 | error | Contain message of each error parameter. | String |
| 6 | fileSize | File size of thumbnail photo of the post. | int |

**Method**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | isValid | Verify all format of parameters and send the error message to render method. | Param: title, body, fileSize  Return: - |
| 2 | handleChange | Receive post of the user from post id. | function(event) |
| 3 | clickSubmit | Put the information of post update to the database. | function(event) |
| 4 | editPostForm | Display information of post and allow user to handling edit information. | Param: title, body, bodys  Return: - |
| 5 | render | Display information of post. | Return: title, body, bodys |

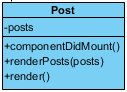
**CD-08 : apiPost**



**Method**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | create | Post or create the post of user to the database by userId. | Param: userId, post  Return: - |
| 2 | singlePost | Get or receive post information from the database by postId. | Param: postId  Return: - |
| 3 | listByUser | Get or receive all post information of post update to the database by userId. | Param: userId  Return: - |
| 4 | remove | Delete a post from the database by postId. | Param: postId  Return: - |
| 5 | update | Put or update the post of user to the database by postId. | Param: postId , post  Return: - |

**CD-09 : apiPost**



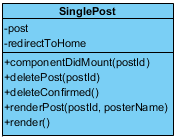
**Attributes**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | posts | All of the posts in the database | Array |

**Method**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | componentDidMount | Receive information of all posts from the database. | function(event) |
| 2 | renderPosts | Receive all of the posts to display and send to render function. | Param: posts  Return: - |
| 3 | render | Display all of the posts. | Return: posts |

**CD-10 : SinglePost**



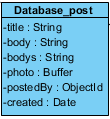
**Attributes**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | posts | All of the posts in the database | Array |
| 2 | redirectToHome | Redirect to home page. | boolean |

**Method**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | componentDidMount | Receive information of post from the database by postId. | Param: postId  Return: - |
| 2 | deletePost | Remove the post from the database by postId and send to deleteConfirmed | Param: postId  Return: - |
| 3 | deleteConfirmed | Alert delete confirm box to the user. | function(event) |
| 4 | renderPost | Receive all of the posts to display , provide Edit and Delete post button send to render function. | Param: postId, postName  Return: - |
| 5 | render | Display single page of the post. | Return: post |

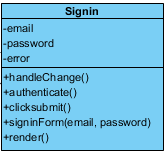
**CD-11 : Database\_post**



**Attributes**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | photo | Thumbnail photo of the post. | Buffer |
| 2 | title | Name of the dish. | String |
| 3 | body | Ingredient of the dish. | String |
| 4 | bodys | Step of cook of the dish. | String |
| 5 | postedBy | Name of user from object id by user. | ObjectId |
| 6 | created | Date of user created the post. | Date |

**CD-12 : Signin**



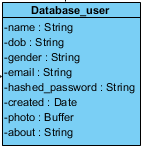
**Attributes**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | email | Email of the user | String |
| 2 | password | Password of the user | String |
| 3 | error | Contain message of each error parameter. | String |

**Method**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | handleChange | Set state of error message and set event from user name value. | function(event) |
| 2 | authenticate | Authenticate the user to sign in to the web application. | function(event) |
| 3 | clicksubmit | Send email and password to the database. | function(event) |
| 4 | signinForm | Display the sign in form to the user for handling sign in. | Param: email, password  Return: - |
| 5 | render | Display error message and redirect to home page. | Return: email, password |

**CD-13 : Database\_user**



**Attributes**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | name | Thumbnail photo of the post. | Buffer |
| 2 | dob | Name of the dish. | String |
| 3 | gender | Ingredient of the dish. | String |
| 4 | email | Step of cook of the dish. | String |
| 5 | hashed\_password | Name of user from object id by user. | ObjectId |
| 6 | created | Date of user created the post. | Date |
| 7 | photo | Photo profile of the user. | Buffer |
| 8 | about | Bio about the user. | String |

**CD-14 : Database\_auth**



**Attributes**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | userExists | The user that has in the database. | String |
| 2 | user | The new user in the database. | String |

**Method**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | signin | Sign in to the spotifood application and check email , password in the database then access to the spotifood application. | function(event) |
| 2 | signout | Sign out the user to the spotifood application. | function(event) |

**CD-15 : Database\_Menu**



**Method**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | isAuthenticated | Authenticated the user who can access the function in spotifood application. | function(event) |
| 2 | signout | Sign out the user to the spotifood application. | function(event) |

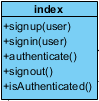
**CD-16 : Database\_Menu**



**Method**:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | isAuthenticated | Authenticated the user who can access the function in spotifood application. | function(event) |
| 2 | signout | Sign out the user to the spotifood application. | function(event) |

**CD-17 : Database\_Menu**

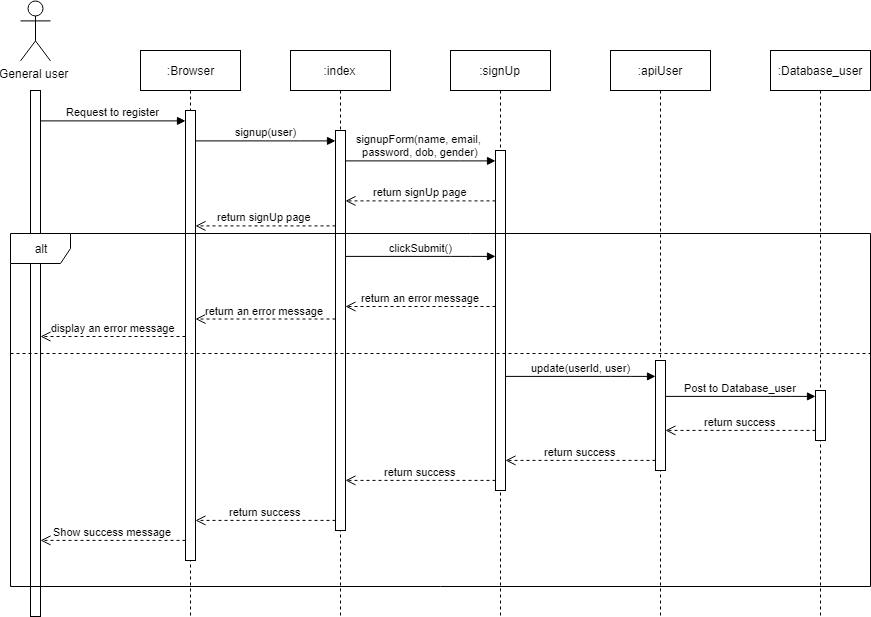


**Method**:

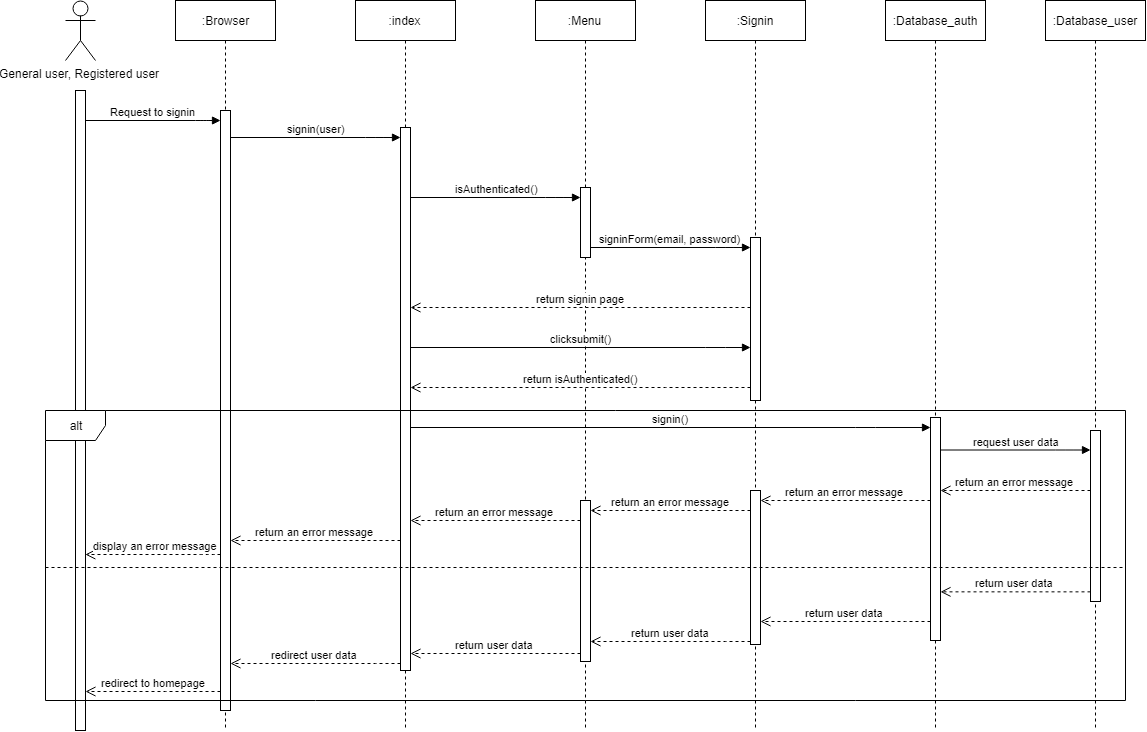
|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | Description | Type |
| 1 | signup | Authenticated the user who can access the function in spotifood application and send to authenticate function by post jwt token to json file in database. | Param: user  Return: - |
| 2 | signout | Sign out the user to the spotifood application. | function(event) |
| 3 | authenticate | Set jwt token to json file in database. | function(event) |
| 4 | signin | Post the user into the database of spotifood application by jwt token to json file in database. | Param: user  Return: - |
| 5 | isAuthenticated | Authenticated the user who can access the function in spotifood application by get jwt token to json file. | function(event) |

# Chapter Four| Sequence Diagram

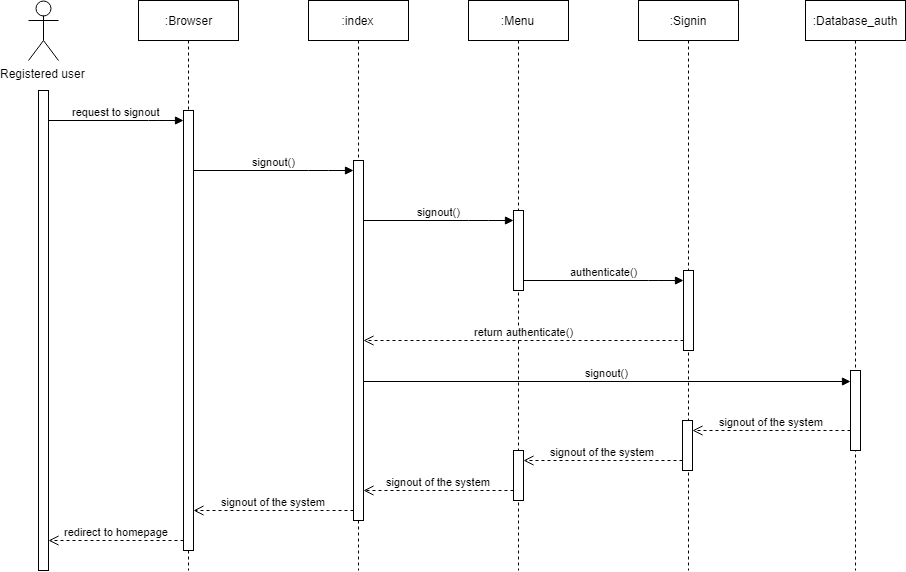
SD-01 Register



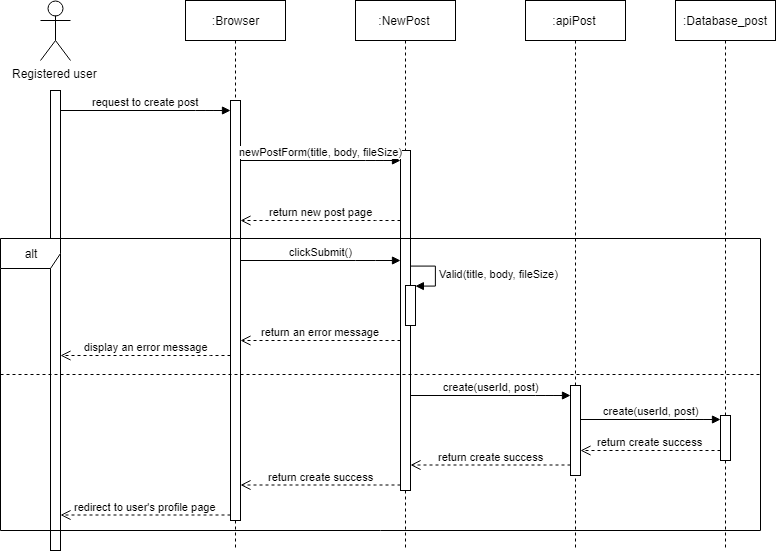
SD-02 Sign in



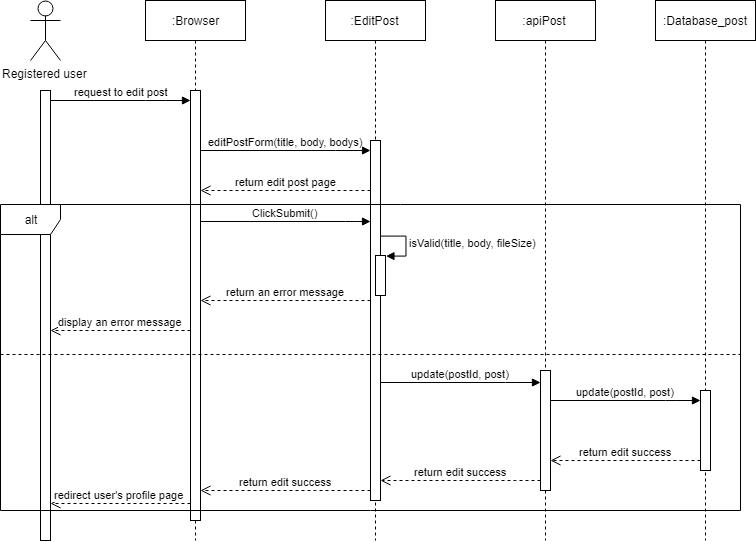
SD-03 Sign out



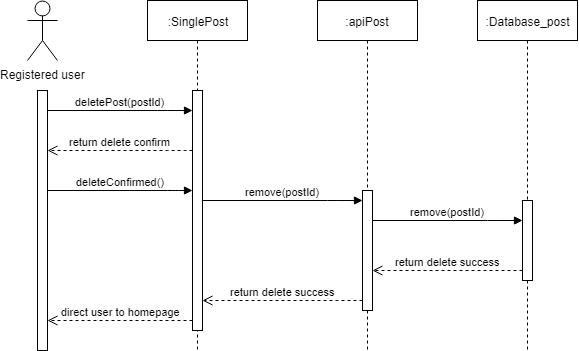
SD-04 Create post



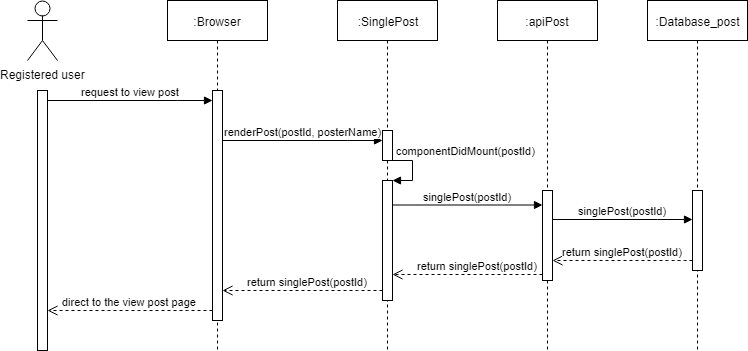
SD-05 Edit post



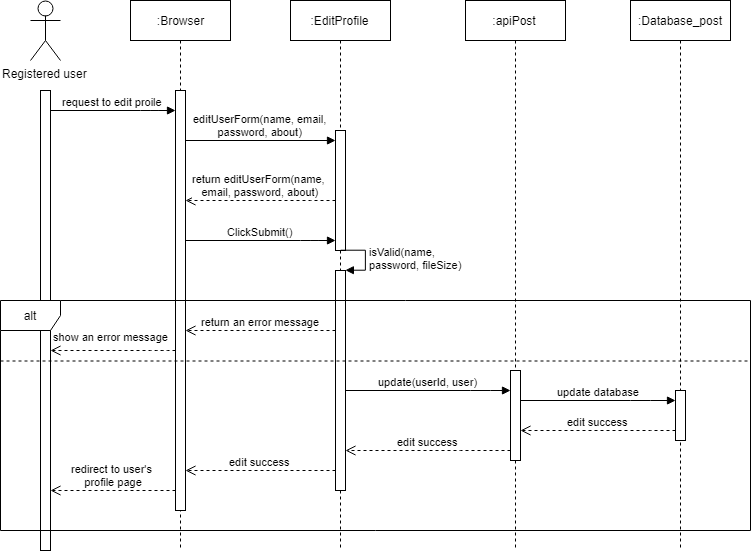
SD-06 Delete post



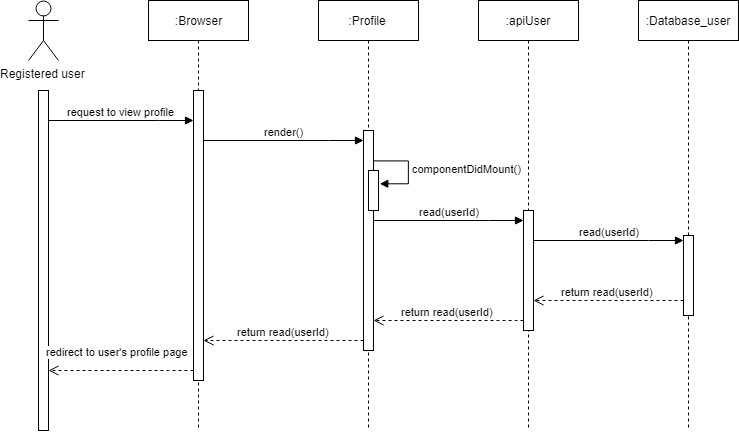
SD-07 View post



SD-08 Edit profile

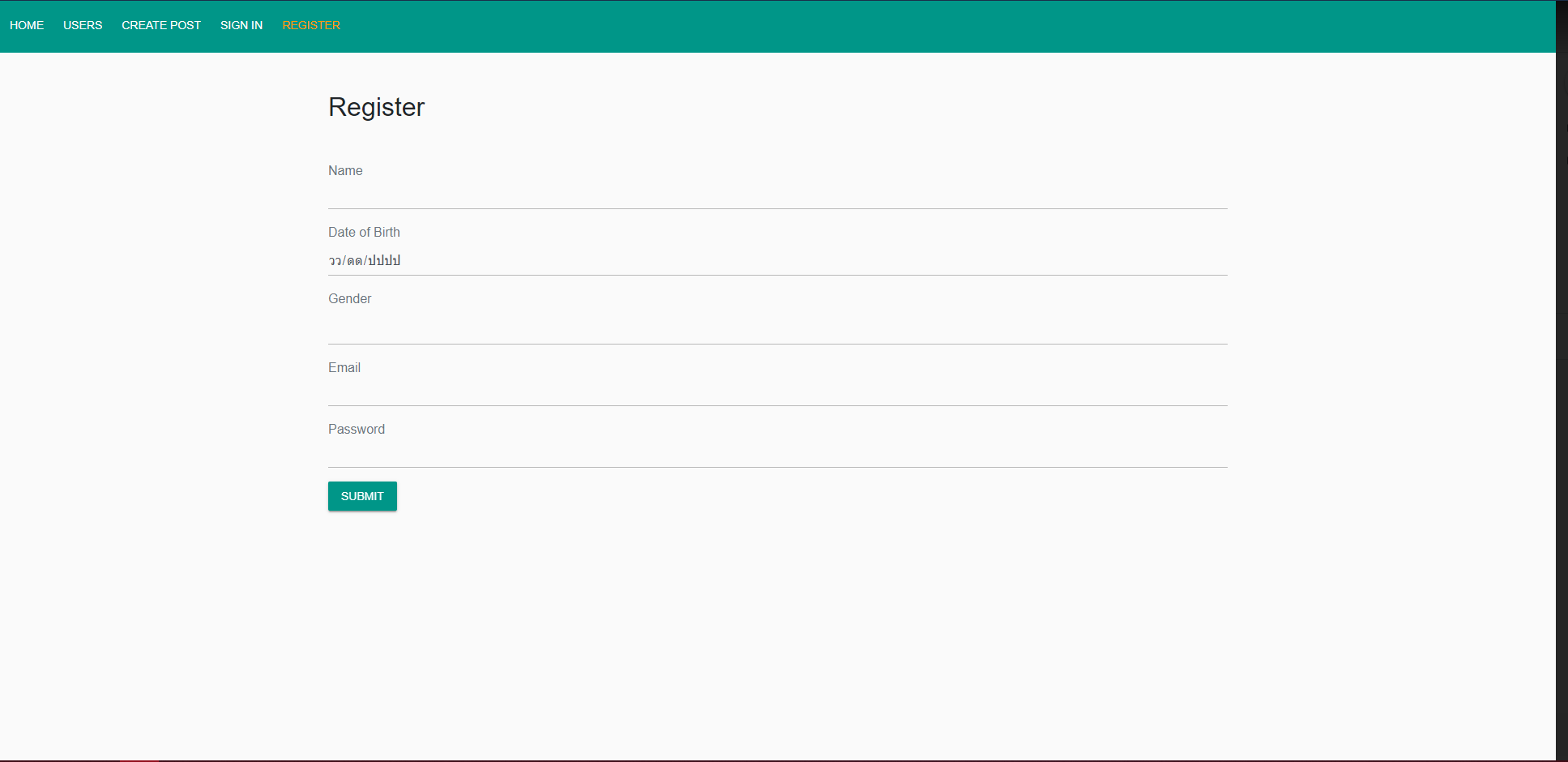


SD-09 View profile

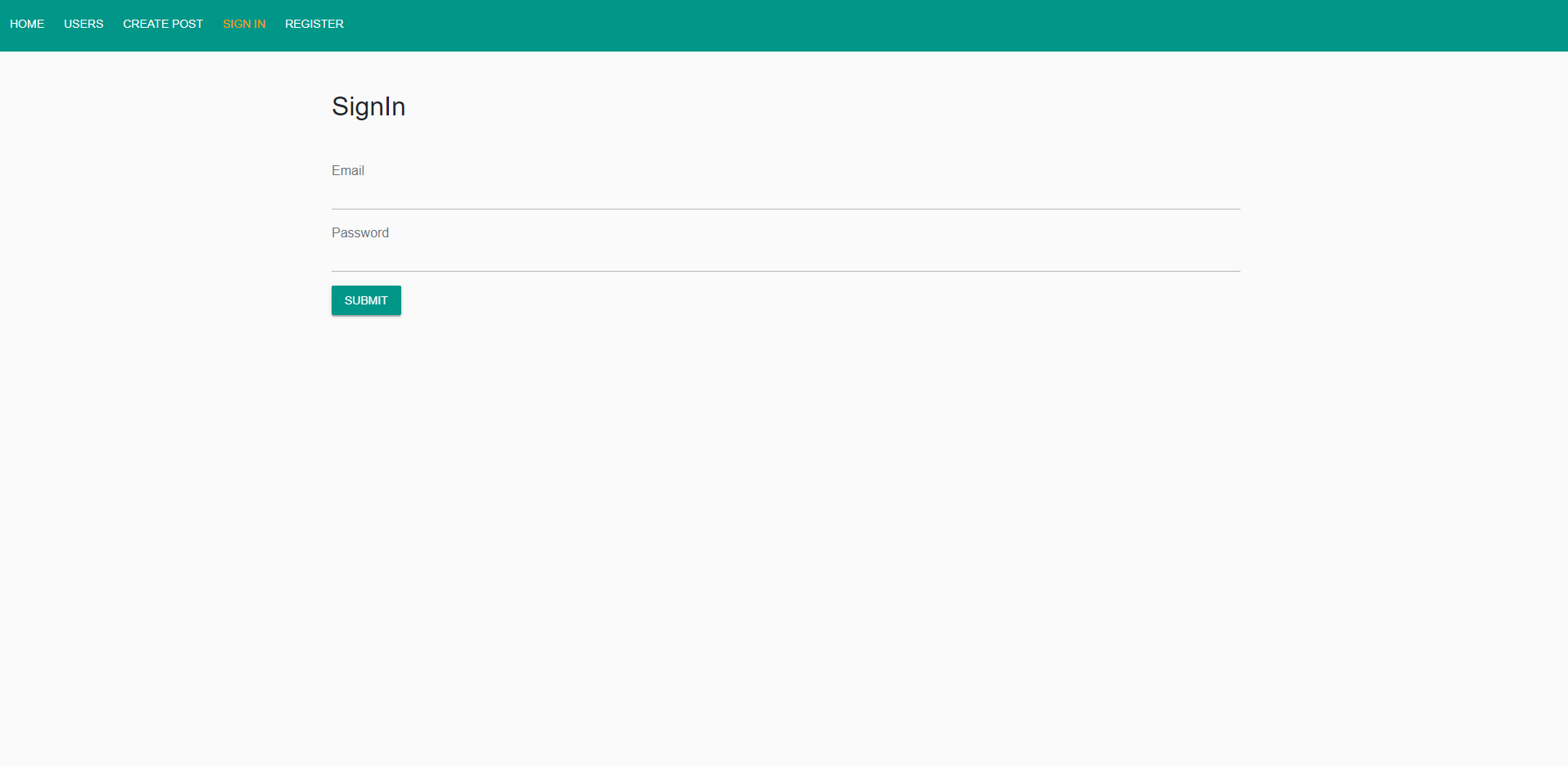


# Chapter Five| User Interface

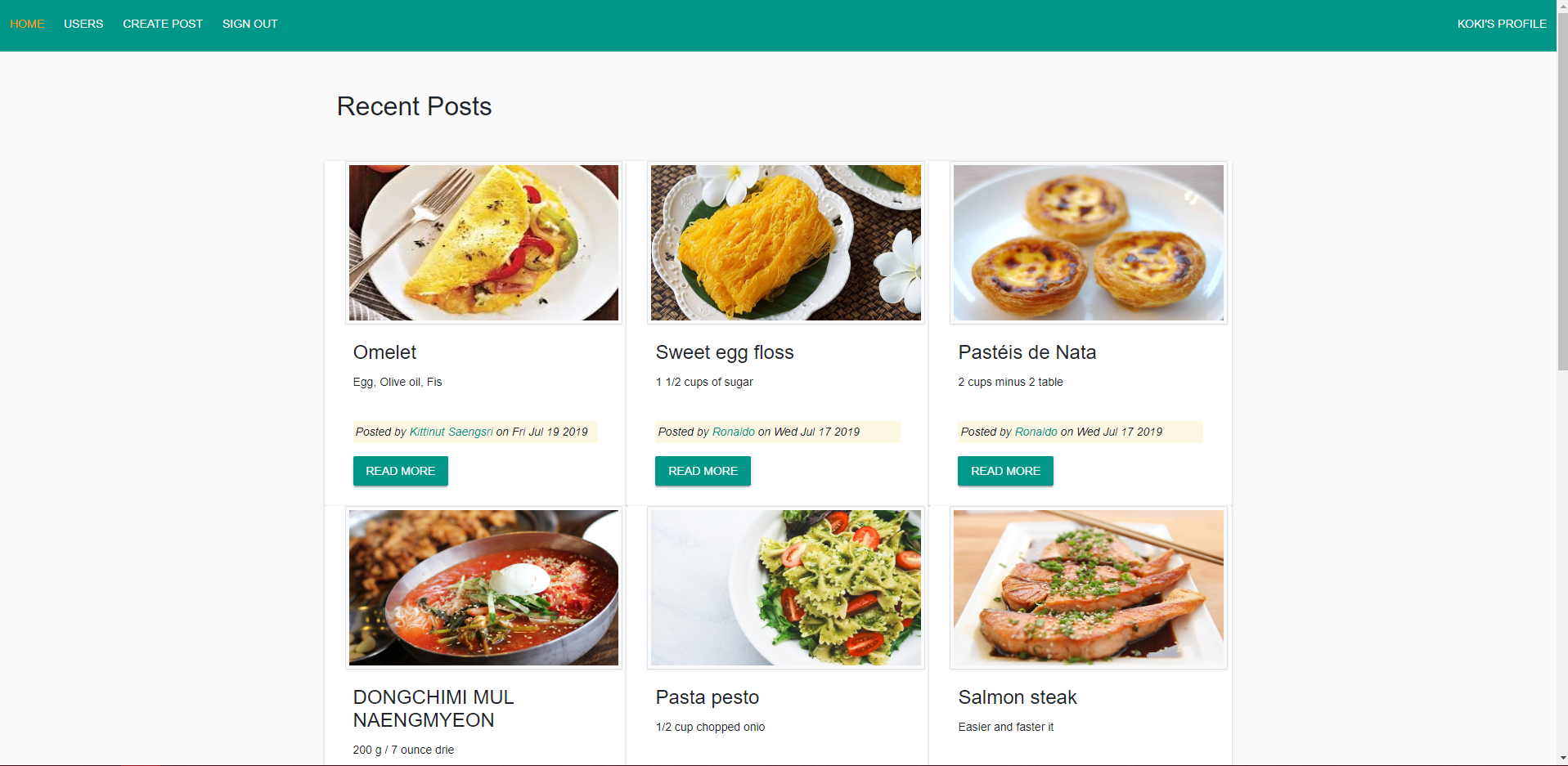
UI-01: Register page



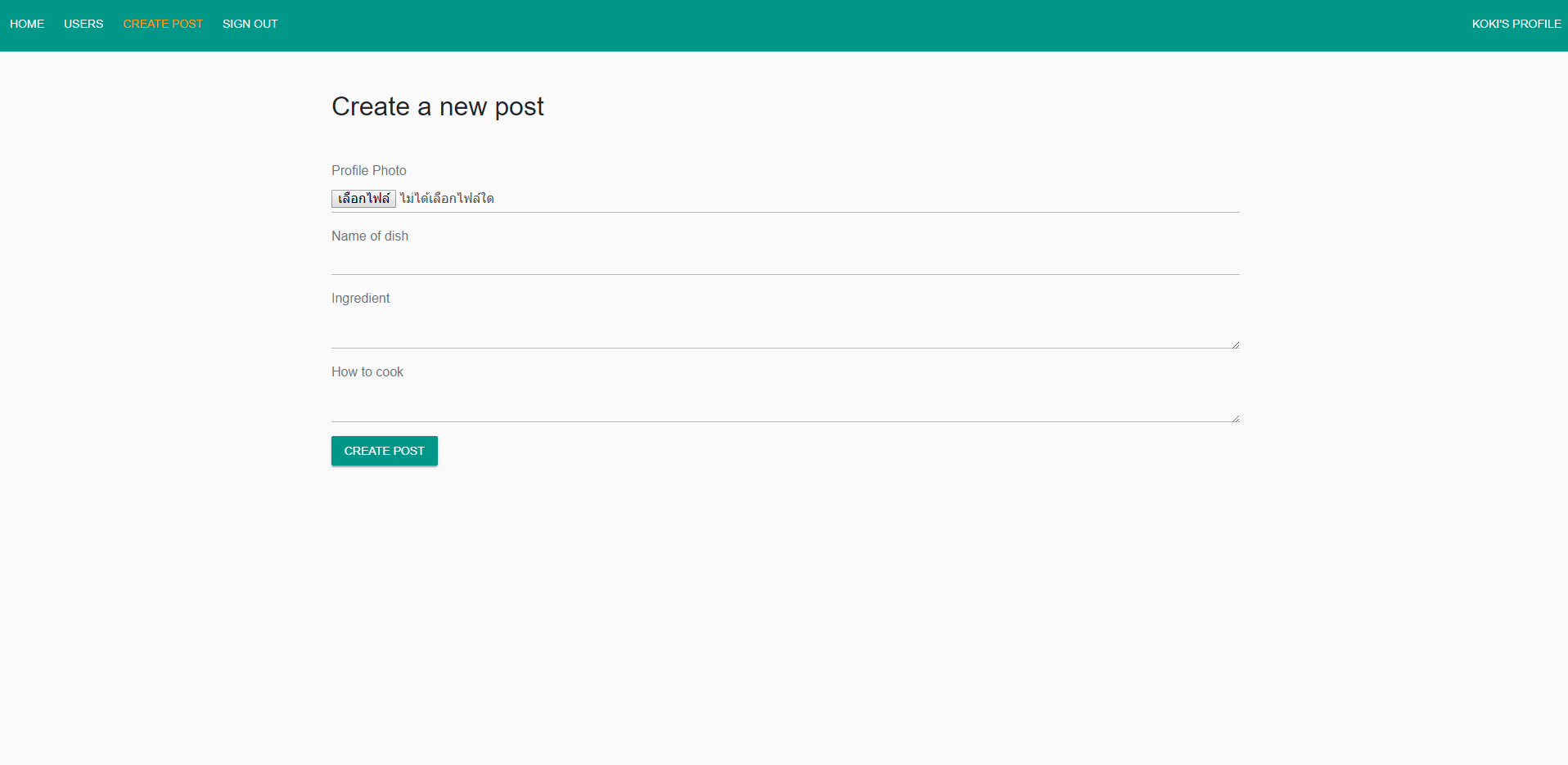
UI-02: Sign in page



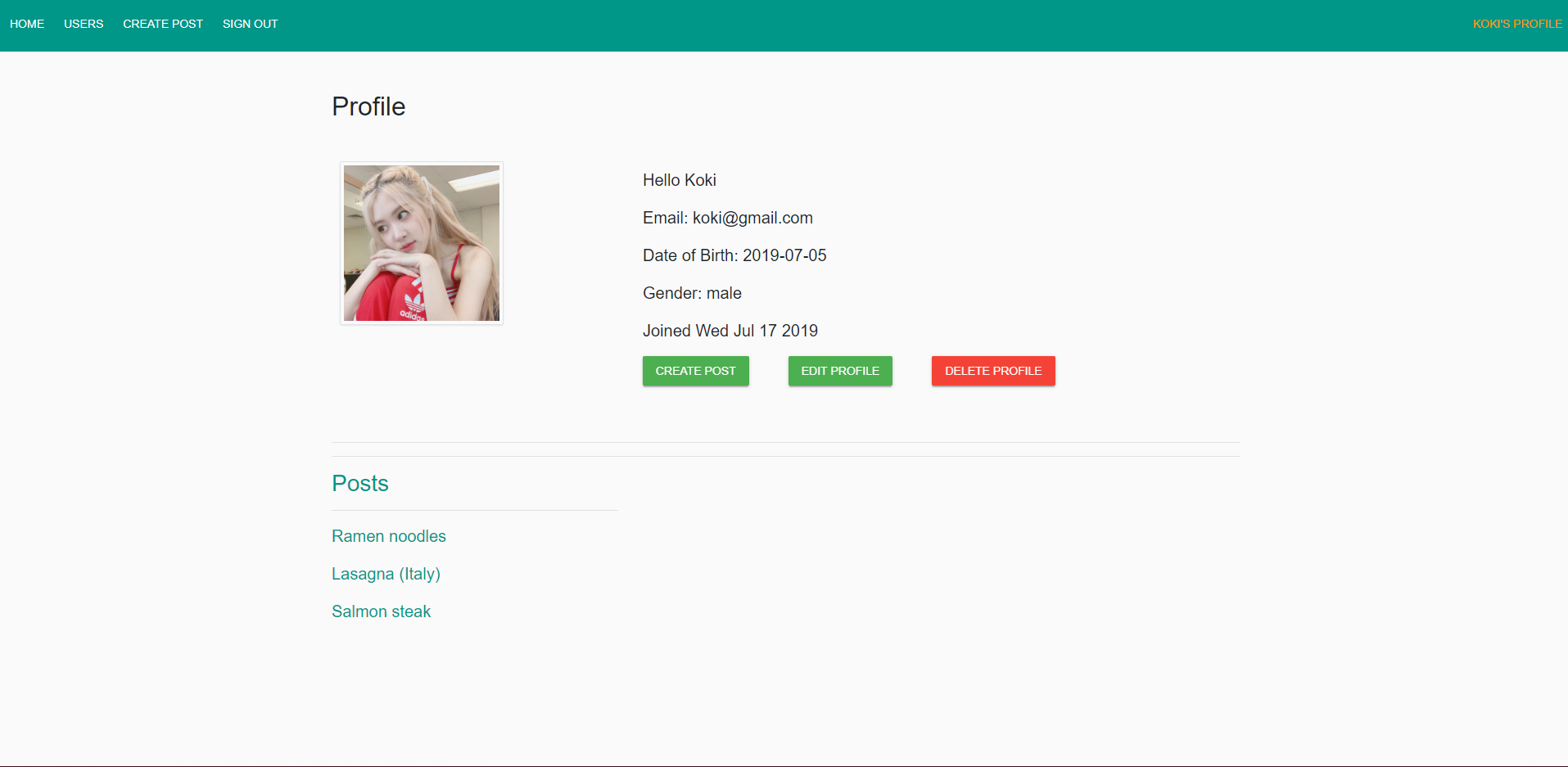
UI-03: Home page



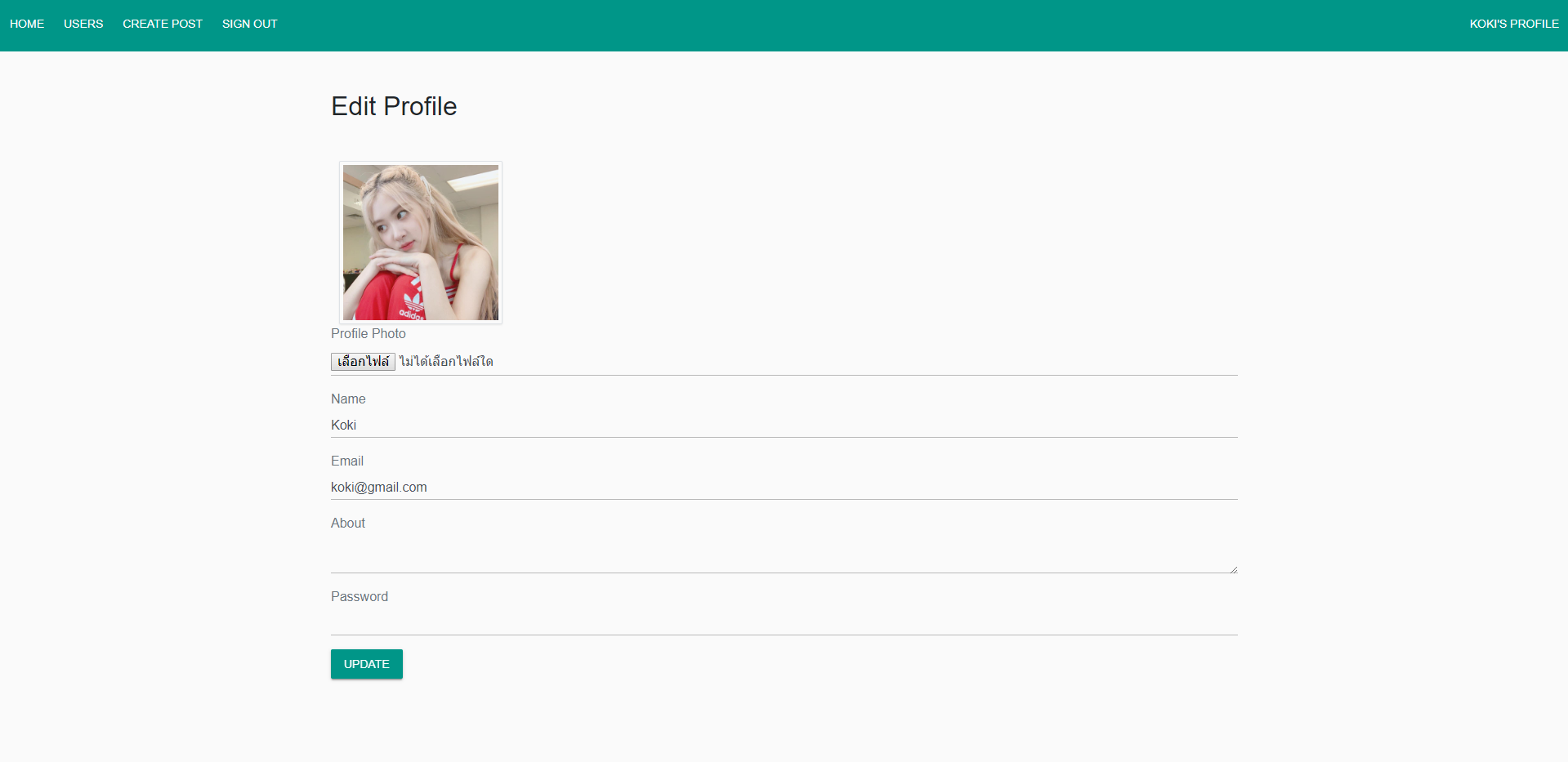
UI-04: Create new post page



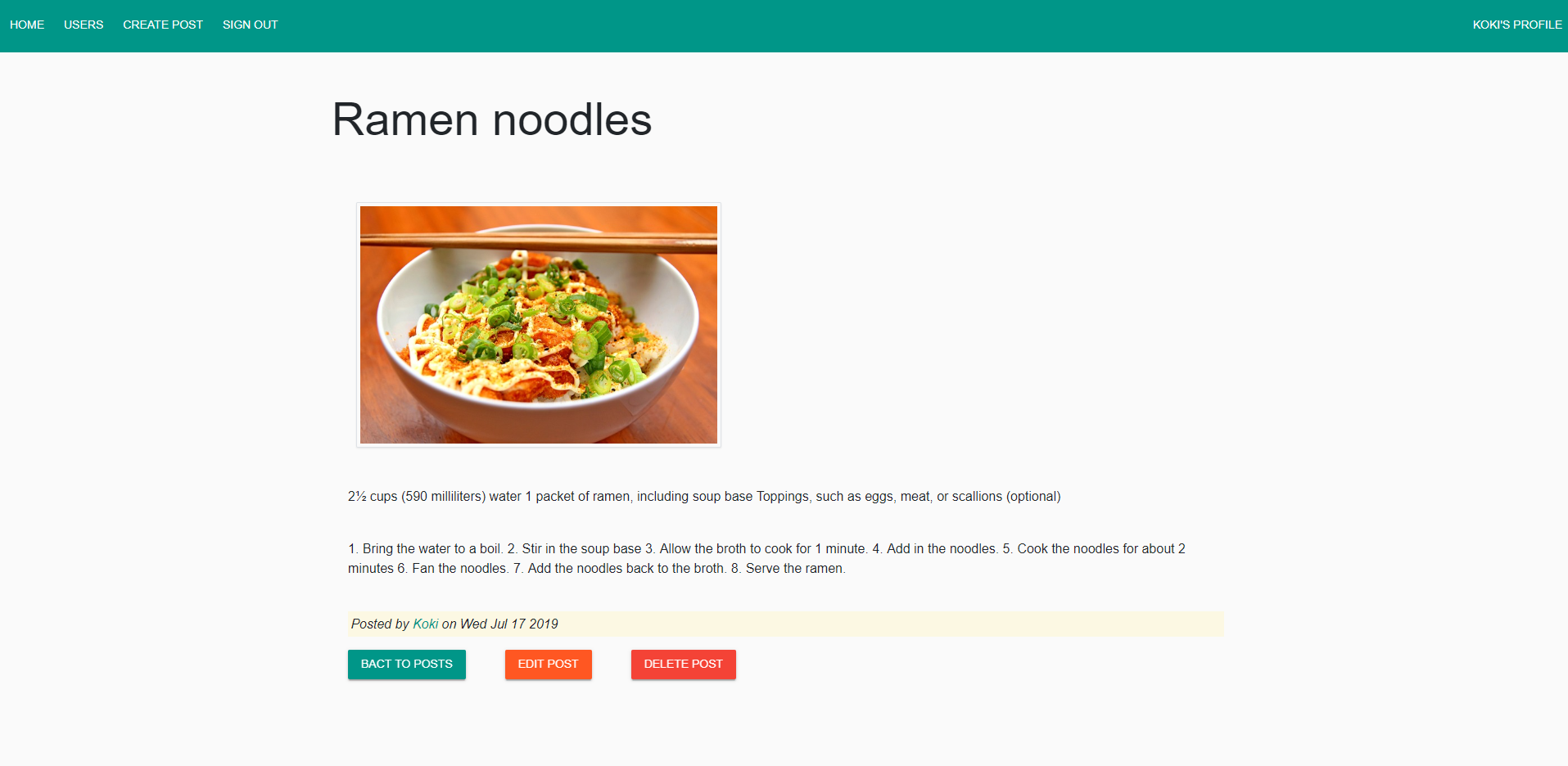
UI-05: Profile page



UI-06: Edit profile page



UI-07: The single dish post page



UI-08: Edit dish page

