# Linnaeus Course Project

# Cookbook

**CookBook** is an application that is used to store and browse cooking recipes.

## Technological Requirements

* **The application must be implemented using ASP.NET Core and MSSQL Server.**
* **The application must have at least 20 recipes**
* **The application must have at least 30 ingredients.**

The Technological Requirements are **ABSOLUTE**. If you do not follow them, you will **NOT** be scored for other Requirements.

Now that you know the **Technological Requirements**, let us see what the **Functional Requirements** are.

## Database Requirements

The **entities** of **CookBook**:

### User

* **Id** **– GUID (PK)**
* **Username – String [3, 30] (required, unique)**
* **Email – Email** **(required)**
* **Password – Hashed** **string [5, 30] (required)**
* **Picture – Byte[] (optional)**
* **Favorites** – **Collection<Recipe> (required)**
* **Cart** – a **Dictionary<Ingredient, Double> (required)**
* **Messages** – a **Collection<Message> (required)**

### Ingredient

* **Id** **– GUID (PK)**
* **Name** **- String [3, 30] (required)**
* **Unit - String (required)**

### Recipe

* **Id – GUID (PK)**
* **Name –** **String** **[6, 50] (required)**
* **Picture – Byte[] (required)**
* **Description – String [10, 300] (required)**
* **Instructions –** **String [20, 1000] (required)**
* **Author – GUID (FK, required)**
* **Tags – Collection<RecipeTag> (required)**
* **Ingredients – Dictionary<Ingredient, Double> (required)**
* **ServingSize – byte [2, 20] (required)**
* **Comments – Collection<Comment> (required)**
* **Date - DateTime (required)**

### Tag

* **Id – GUID (PK)**
* **Name – String** **[2, 30] (required)**
* **User – GUID (FK, optional)**

### RecipeTag

* **RecipeId – GUID (PK)**
* **TagId – GUID (PK)**

### Comment

* **Id – GUID (PK)**
* **User – GUID (FK, required)**
* **Text – String [5, 300] (required)**
* **Recipe – GUID (required)**
* **Date – DateTime (required)**

### Message

* **Id – GUID (PK)**
* **Sender – GUID (FK, required)**
* **Receiver –** **GUID (FK, required)**
* **Text -** **String** **[5, 300] (required)**
* **IsRead – Boolean (required)**
* **Recipe – GUID (FK, optional)**
* **Date – DateTime (required)**

Implement the entities with the **correct data types** and their **relations**.

## Page Requirements

### Login Page (logged-out user)

* User can either log-in or register.

### Register Page (logged-out user)

* User can either log-in or register.

### Home Page (logged-in user)

* This is where we store all user recipes.
* Users can create new recipes by clicking the plus button in the bottom right corner.
* User can search for a recipe by name.
* User can search for a recipe by filters.
  + Tags: Vegetarian, Vegan, Pesto, Lactose-free, Gluten-free, Starter, Main course, Dessert, Sweets, etc.
  + Ingredients: Egg, pasta, tomato, chicken, beef, pork, lettuce, potato, cheese, onion, garlic, milk, etc.
* Each page stores 6 recipes, and user can navigate pages.
* User can hover on recipes to see a short description and click on them to go to their own page.
* User can hover on recipes and add them to favorites by clicking on the empty heart, or if it is colored, click on it to remove them from favorites.

### Home Page Filter(logged-in user)

* User can filter by some given filters or search for more.
* All selected filters should be shown to the user in a plaintext boxt that the user can edit.
* Filters are applied when user clicks on “Search”.

### Favorites Page (logged-in user)

* This is where we store the recipes a user has liked.
* User adds recipes to favorites and they appear here.
* If a user removes a recipe from favorites, the recipe should no longer show up here.
* User can navigate the recipes using the search or filter functions.
* There can only be 6 recipes per page.

### Add Recipe Page (logged-in user)

* This is where a new recipe is added.
* All users can create a recipe.
* The recipe must have a name between 6 and 50 characters long.
* The recipe may or may not have a PictureURL.
* Users can list ingredients and tags. If they add a non-existent tag, it needs to be created.
* Users must list instructions (cooking steps) separated by a comma.
* The recipe must have a description, between 10 and 300 characters long.

### Edit Recipe Page (logged-in user)

* Edit recipe is the same as the add recipe page but with pre-filled fields.
* Only the admin and the author of the recipe can edit it.

### Messages Page (logged-in user)

* All user messages should be stored here.
* Users can delete messages.
* Unread messages should be bold.

### Settings Page (logged-in user)

* User can edit their username, email, password.
* User can delete account.

### Shopping Cart Page (logged-in user)

* User can see products and quantities
* User can modify quantities
* User can add new products
* User can remove existing products
* User can remove all products at once

## Functionality

The functionality of the **Cookbook** is very simple.

### Guests

**Guests** can **Register** and **Login.**

### Users

**Users** can:

* **see recipes**
* **add recipes to favorites**
* **add new recipes**
* **edit their own recipes**
* **add ingredients to cart**
* **send messages**
* **receive messages**
* **add tags**

### Admin

**Admin can:**

* **add users**
* **remove users**
* **edit any recipe**

### Redirections

Guest:

* **Index** **Page** should redirect to **Login page**.

Customer:

* Upon successful **Registration** of a **User**, you should be redirected to the **Login** **Page**.
* Upon successful **Login** of a **User**, you should be redirected to the **Home Page**.
* Upon successful **Update of settings**, you should be redirected to the **Settings Page** with a success message**.**
* Upon successful **Addition of a recipe**, users should be redirected to the **Recipe’s Page**.
* Upon successful **Modification of a recipe,** user should be redirected to the **Recipe’s Page.**
* Upon successful **Removal of a recipe,** user should be redirected to the **Home Page.**
* Upon successful **Starring of a recipe,** user should not be redirected.
* Upon successful **Unstarring of a recipe,** user should not be redirected.
* Upon successful **Addition of a product to cart,** user should not be redirected.
* Upon successful **Removal of a product to cart,** user should not be redirected.
* Upon successful **Sending of a message,** user should be redirected to **Message Page.**
* Upon successful **Removal of a message,** user should be redirected to **Messages Page.**
* Upon successful **Logout** of a **User**, you should be redirected to the **Login Page**.

Admin:

* Upon successful **Addition** of a **User**, you should be redirected to the **Home Page**.
* Upon successful **Removal** of a **User**, you should be redirected to the **Home Page**.

## Security

The **Security** section mainly describes access requirements. Configurations about which users can access specific functionalities and pages:

* **Guest** (not logged in) users cannot access **Users-and Admin-only** pages.
* **Guest** (not logged in) users can access the **Login** page.
* **Guest** (not logged in) users can access the **Register** page.
* **Users** (logged in) cannot access **Users- and Admins-only** pages.
* **Users** (logged in) can access the **Home** page.
* **Users** (logged in) can access the **Favorites** page.
* **Users** (logged in) can access the **Recipe** page.
* **Users** (logged in) can access the **Cart** page.
* **Users** (logged in) can access the **AddRecipe** page.
* **Users** (logged in) can access the **ModifyRecipe** page.
* **Users** (logged in) can access the **Messages** page.
* **Users** (logged in) can access the **User** page.
* **Users** (logged in) can access the **Settings** page.
* **Admin** (logged in) can access **all Users’** pages.

## Limitations

* **Users** can only **modify or remove recipes** they have created.
* **Admin** can **modify/remove all recipes.**
* There can only be one **Admin** (nickname **“Admin”**).