# 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 JDK16, JavaFX, MySQL.**
* **The application must have at least 20 recipes**
* **The application must have at least 30 ingridients**
* **Each recipe must have at least 3 tags**

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 **Database** of **HousingAgency**:

### Users

* Has an Id – a **GUID, Primary Key**
* Has a UserName – a string with **min length** **3** and **max length 20** (**required**)
* Has a DisplayName – a string with **min length** **3** and **max length 20** (**required**)
* Has an Email – an email (**required**)
* Has a Password – a hashed string with **min length** **5** and **max length 20 (before hashed)** - no max length required for a hashed password in the database (**required**)

### Ingridients

* Has a **Id** – a **UUID, Primary Key**
* Has a IngridientName – a String with min length 3 and max length 20 (required)
* Has a **Unit** – a **UnitEnum (required)**

### ShoppingList

* Has an **User**Id – a **UUID, Foreign key, Primary key**
* Has a **ProductId** – a **UUID, Foreign key, Primary key**
* Has a **Quantity –** a **Float (required)**

### Recipes

* Has an **Id** – a **GUID, Primary Key**
* Has a **RecipeName –** a **String** with **min length 6** and **max length 50 (required)**
* Has a **PictureUrl** – a **String**
* Has a **Description –** a **String** with **min length 10** and **max length 300 (required)**
* Has a **AuthorId –** a **UUID (required)**

### RecipesTags

* Has a **RecipeId –** a **UUID (required)**
* Has a **TagId – a UUID (required)**

### Tags

* Has a **Id –** a **UUID (required)**
* Has a **TagName –** a **String (required)**

### RecipeIngridients

* Has a **RecipeId –** a **UUID, Primary Key**
* Has a **IngridientId** – a **UUID, Primary Key**
* Has a **Quantity –** a **float (required)**

### RecipeComment

* Has an **Id** – a **GUID, Primary Key**
* Has a **RecipeId –** a **UUID (required)**
* Has a **UserId** – a **UUID (required)**
* Has a **Comment –** a **String** with **max length 200 (required)**

### UsersFavorites

* Has a **RecipeId –** a **UUID, Primary Key**
* Has a **UserId** – a **UUID, Primary Key**

### UsersMessages

* Has an **Id** – a **GUID, Primary Key**
* Has a **SenderId –** a **UUID (required)**
* Has a **ReceiverId** – a **UUID (required)**
* Has a **RecipeId –** a **UUID**
* Has a **Message –** a **String** with **max length 200 (required)**

### UsersLists

* Has an **Id** – a **GUID, Primary Key**
* Has a **SenderId –** a **UUID (required)**
* Has a **ReceiverId** – a **UUID (required)**
* Has a **RecipeId –** a **UUID**
* Has a **Message –** a **String** with **max length 200 (required)**

Implement the entities with the **correct datatypes** and their **relations**.

## Page Requirements

### Login Page (logged-out user)

Graphical user interface, text, application, chat or text message

Description automatically generated

* User can either log-in or register.

### Register Page (logged-out user)

Graphical user interface, text, application, chat or text message

Description automatically generated

* User can either log-in or register.

### Home Page (logged-in user)

A picture containing text, food, dish

Description automatically generated

* 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 Search (logged-in user)

Graphical user interface, application

Description automatically generated

* 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)

Graphical user interface, application, website

Description automatically generated

* 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.

### Weekly Plan Page (logged-in user)

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

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

### Messages Page (logged-in user)

Graphical user interface, application

Description automatically generated

* All user messages should be stored here

### Settings Page (logged-in user)

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

### Account Page (logged-in user)

**NOTE**: The templates should look **EXACTLY** as shown above.

**NOTE**: The templates do **NOT** **require** **additional** **CSS** for you to write. Only **bootstrap** and the **given CSS** are enough.

## Functionality

The functionality of the **Football Manager** Platform is very simple.

### Users

Guests can Register, Login and visit all properties.

Customers can store profile information, accumulate points, apply for properties and own a property.

**Administrators** can **add a property, remove a tenant.**

### Customers

Customers can:

* store profile information
* accumulate and buy points (free coupon for testing)
* apply for properties

### Redirections

Guest:

* Index Page should redirect to /Properties/All.

Customer:

* Upon successful Registration of a Customer, you should be redirected to the **/Customer/Settings** Page.
* Upon successful update of settings, you should be redirected to the **/Customer/Settings Page** with a success message**.**
* Upon successful Login of a User, you should be redirected to the Index Page.
* Upon successful Application to a Property, you should be redirected to the /Properties/Details?id={propertyId} Page.
* Upon successful Purchase of points to the User’s collection, should be redirected to the /Customer/BuyPoints Page.
* Upon successful Logout of a User, you should be redirected to the Index Page.
* If any of the **validations** in the POST forms **don’t pass**, **redirect** to the **same page** (**reload/refresh** it).

Tenant:

* Upon successful Login of a Tenant, you should be redirected to the **/Tenant/**Property Page.
* Upon successful **Termination of contract**, you should be redirected to the **/Tenant/Property Page**.
* Upon successful **Payment of bill**, you should be redirected to the **/Tenant/Property Page**.

Admin:

* Upon successful **Addition of a Property**, you should be redirected to the **/Properties?id={propertyId} Page**.
* Upon successful **Removal of Tenant**, you should be redirected to the **/Properties?id={propertyId} 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-, Tenants- and Admins-only pages.
* Guest (not logged in) users can access the Index page.
* Guest (not logged in) users can access the Login page.
* Guest (not logged in) users can access the Register page.
* Customers (logged in) cannot access Guest, Tenant and Admin-only pages.
* Customers (logged in) can access the /Customers/Profile/ page.
* Customers (logged in) can access the **/**Properties/All page.
* Customers (logged in) can access the **/Properties/Apply?id={propertyId}** page.
* Customers (logged in) can access the **/Customers/**BuyPoints page.
* Customers (logged in) can access the **/**Logout page.
* Tenants (logged in) cannot access Users- and Admins-only pages.
* Tenants (logged in) can access the /Tenants/Property page.
* Tenants (logged in) can access the /Tenants/Property/Terminate page.
* Tenants (logged in) can access the /Tenants/Bills page.
* **Admins** (logged in) can access the **/Properties/Add** page.
* **Admins** (logged in) can access the **/Properties/RemoveTenant?id={propertyId}** page.
* **Admins** (logged in) can access the **/Customers/Profile?id={customerId}**

## Limitations

* **Customers** can only **apply for a property** if they have provided a **personal number** and **phone number.**