

# Aplikacja książka kucharska

Mikołaj Kubik

12 czerwca 2020

## 1 Wstęp

Celem projektu było stworzenie aplikacji webowej, będącej publiczną książką kucharską. Użytkownicy mogą dodawać nowe składniki oraz tworzyć z nich przepisy a także dodawać komentarze do istniejących przepisów. W bazie danych przechowywane są wartości odżywcze poszczególnych składników, dzięki czemu użytkownik może lepiej dopasować dany przepis do swoich preferencji żywieniowych.

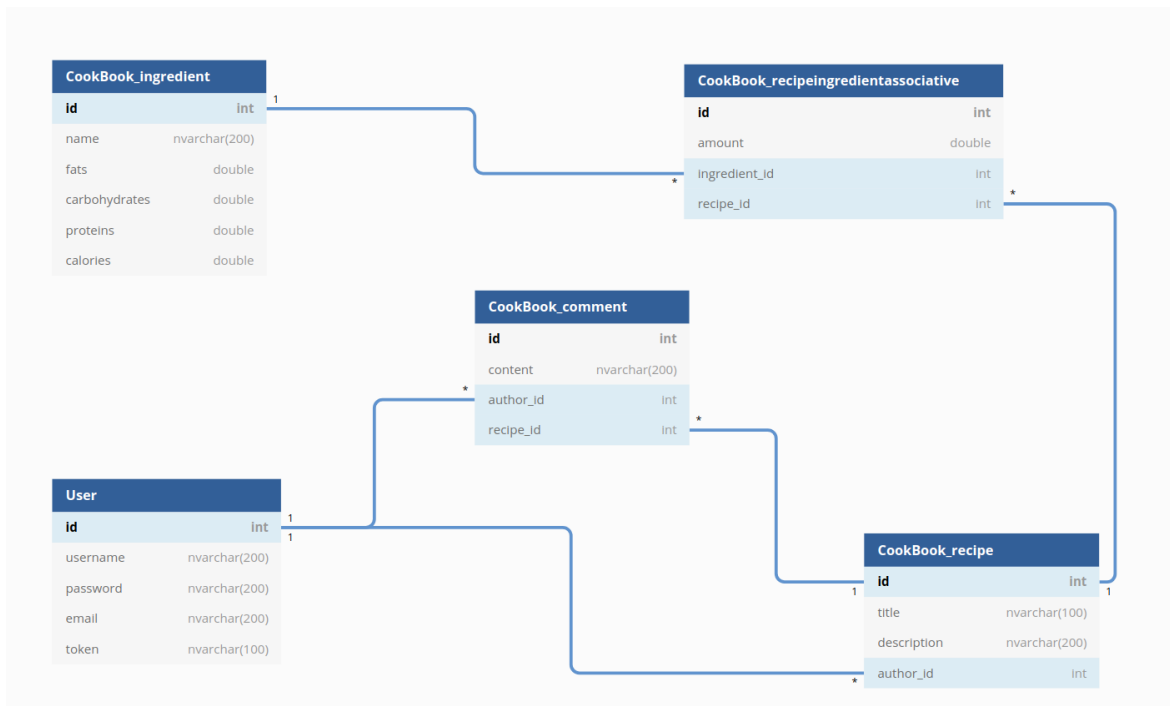
## 2 Architektura techniczna

### 2.1 Wykorzystane narzędzia

- Docker - utworzenie zamkniętego ekosystemu zawierającego bazę danych, API oraz aplikację frontendową. Docker compose budujący infrastrukturę aplikacji korzystając z obrazu bazy danych SQL server oraz zdefiniowanych na potrzeby projektu plików Dockerfile dla frontendu i backendu,
- Django
  - obsługa bazy danych Django ORM
  - Django Rest Framework - obsługa interfejsu API
  - pyodbc, django-mssql-backend - podłączenie bazy danych mssql do aplikacji django
- React - obsługa interfejsu użytkownika - obustronnej komunikacji z serwerem
- MS SQL - baza danych

## 2.2 Baza danych

### 2.2.1 Diagram bazy danych



### 2.2.2 Skrypt tworzący bazę danych

```
1  --
2  -- Create model Ingredient
3  --
4  CREATE TABLE [User]
5  (
6      [id] int IDENTITY (1, 1) NOT NULL PRIMARY KEY,
7      [username] nvarchar(200) NOT NULL UNIQUE,
8      [password] nvarchar(200) NOT NULL,
9      [email] nvarchar(200) NOT NULL UNIQUE,
10     [token] nvarchar(100) NOT NULL UNIQUE,
11 );
12 --
13 -- Create model Ingredient
14 --
15 CREATE TABLE [CookBook_ingredient]
16 (
17     [id] int IDENTITY (1, 1) NOT NULL PRIMARY KEY,
18     [name] nvarchar(200) NOT NULL UNIQUE,
19     [fats] double precision NOT NULL,
20     [carbohydrates] double precision NOT NULL,
21     [proteins] double precision NOT NULL,
22     [calories] double precision NOT NULL
23 );
24 --
25 -- Create model Recipe
26 --
27 CREATE TABLE [CookBook_recipe]
28 (
29     [id] int IDENTITY (1, 1) NOT NULL PRIMARY KEY,
30     [title] nvarchar(100) NOT NULL UNIQUE,
```

```

31     [description] nvarchar(200) NOT NULL,
32     [author_id] int NOT NULL
33 );
34 --
35 -- Create model RecipeIngredientAssociative
36 --
37 CREATE TABLE [CookBook_recipeingredientassociative]
38 (
39     [id] int IDENTITY (1, 1) NOT NULL PRIMARY KEY,
40     [amount] double precision NOT NULL,
41     [ingredient_id] int NOT NULL,
42     [recipe_id] int NOT NULL
43 );
44 --
45 -- Create model Comment
46 --
47 CREATE TABLE [CookBook_comment]
48 (
49     [id] int IDENTITY (1, 1) NOT NULL PRIMARY KEY,
50     [content] nvarchar(200) NOT NULL,
51     [author_id] int NOT NULL,
52     [recipe_id] int NOT NULL
53 );
54
55 ALTER TABLE [CookBook_recipe] ADD CONSTRAINT [
56     CookBook_recipe_author_id_fk_User_id] FOREIGN KEY ([author_id]) REFERENCES [
57     User] ([id]);
58 ALTER TABLE [CookBook_comment] ADD CONSTRAINT [
59     CookBook_comment_author_id_fk_User_id] FOREIGN KEY ([author_id]) REFERENCES
60     [User] ([id]);
61 ALTER TABLE [CookBook_recipeingredientassociative] ADD CONSTRAINT [
62     CookBook_recipe_id_5219806b_fk_CookBook_recipe_id] FOREIGN KEY ([recipe_id])
63     REFERENCES [CookBook_recipe] ([id]);
64 ALTER TABLE [CookBook_comment] ADD CONSTRAINT [
65     CookBook_comment_recipe_id_fk_CookBook_recipe_id] FOREIGN KEY ([recipe_id])
66     REFERENCES [CookBook_recipe] ([id]);
67 ALTER TABLE [CookBook_recipeingredientassociative] ADD CONSTRAINT [
68     CookBook_recipeingredientassociative_ingredient_id_fk_CookBook_ingredient_id
69     ] FOREIGN KEY ([ingredient_id]) REFERENCES [CookBook_ingredient] ([id]);

```

## 2.3 Opis tabel

Baza danych składa się z czterech głównych tabel oraz tabeli obsługującej relację wiele do wielu pomiędzy przepisami i składnikami.

- Cookbook\_recipe: tabela zawierająca przepisy
  - id
  - title: nazwa przepisu
  - description: opis przepisu
  - author\_id: id autora przepisu
- Cookbook\_ingredient: tabela reprezentująca składnik
  - id
  - name: nazwa składnika
  - fats: zawartość tłuszczu w 100g składnika
  - carbohydrates: zawartość węglowodanów w 100g składnika

- proteins: zawartość białka w 100g składnika
- calories: ilość kalorii w 100g składnika
- Cookbook\_recipeingredientassociative: tabela łącząca wiele do wielu składniki i przepisy
  - id
  - amount: ilość składnika w przepisie w gramach
  - ingredient\_id: id składnika
  - recipe\_id: id przepisu
- User: tabela zawierająca dane użytkownika
  - id
  - login: login
  - password: hasło
  - token: token, któremu musi odpowiadać cookie w żądaniach dodawania nowych przepisów i komentarzy
- Cookbook\_comment: tabela zawierająca komentarze do przepisów
  - id
  - content: treść komentarza
  - author\_id: id autora komentarza
  - recipe\_id: id komentowanego przepisu

## 3 Strona serwerowa

### 3.1 Definicja bazy danych w pythonie

```

1 from django.db import models
2 from django.contrib.auth.models import User
3
4 class Ingredient(models.Model):
5     name = models.CharField(max_length=200, unique=True)
6     fats = models.FloatField(default=0)
7     carbohydrates = models.FloatField(default=0)
8     proteins = models.FloatField(default=0)
9     calories = models.FloatField(default=0)
10
11     def __str__(self):
12         return self.name
13
14 class Recipe(models.Model):
15     title = models.CharField(max_length=100, unique=True)
16     description = models.CharField(max_length=200)
17     author = models.ForeignKey(User, on_delete=models.CASCADE)
18     ingredients = models.ManyToManyField(Ingredient, through='
RecipeIngredientAssociative')
19
20     def __str__(self):
21         return self.title
22
23 class RecipeIngredientAssociative(models.Model):
24     recipe = models.ForeignKey(Recipe, on_delete=models.CASCADE)

```

```

25     ingredient = models.ForeignKey(Ingredient, on_delete=models.CASCADE)
26     amount = models.FloatField()
27
28     class Comment(models.Model):
29         content = models.CharField(max_length=200)
30         author = models.ForeignKey(User, on_delete=models.CASCADE)
31         recipe = models.ForeignKey(Recipe, on_delete=models.CASCADE)
32
33     def __str__(self):
34         return f"{self.author}: {self.content}"

```

## 3.2 Serializery

Dane w większości przechodziły jedynie podstawową walidację - dotyczącą jedynie zadeklarowanego typu jednak w przypadkach tworzenia nowego przepisu lub dodawania komentarza w miejsce autora wstawiany był obecnie uwierzytelniony użytkownik.

```

1 from django.contrib.auth.models import User
2 from .models import Ingredient, Recipe, Comment, RecipeIngredientAssociative
3 from rest_framework import serializers
4
5 class UserSerializer(serializers.HyperlinkedModelSerializer):
6     class Meta:
7         model = User
8         fields = ['url', 'username', 'email', 'groups']
9
10
11 class RecipeIngredientAssociativeSerializer(serializers.ModelSerializer):
12     class Meta:
13         model = RecipeIngredientAssociative
14         fields = ['id', 'recipe', 'ingredient', 'amount']
15
16 class IngredientSerializer(serializers.ModelSerializer):
17     class Meta:
18         model = Ingredient
19         fields = ['id', 'name', 'fats', 'carbohydrates', 'proteins', 'calories']
20 ]
21
22 class RecipeSerializer(serializers.ModelSerializer):
23     class Meta:
24         model = Recipe
25         fields = ['id', 'title', 'description', 'author']
26
27     def create(self, validated_data):
28         validated_data['author'] = self.context['request'].user
29         return super().create(validated_data)
30
31 class CommentSerializer(serializers.ModelSerializer):
32     class Meta:
33         model = Comment
34         fields = ['id', 'content', 'author', 'recipe']
35
36     def create(self, validated_data):
37         validated_data['author'] = self.context['request'].user
38         return super().create(validated_data)

```

## 4 Dokumentacja aplikacji

### 4.1 Dodawanie składników



Sign up Log out

**Add ingredient**

Name

Fats

Proteins

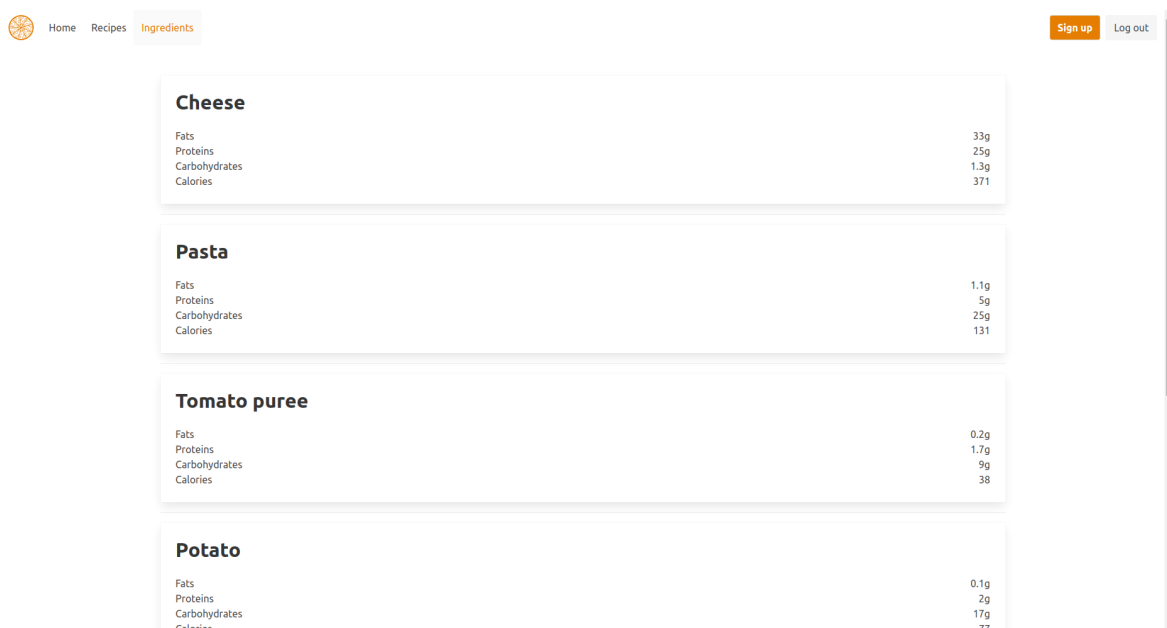
Carbohydrates

Calories

Submit

Uwierzytelniony użytkownik może dodać nowy składnik i jego wartości odżywcze

### 4.2 Wyświetlanie listy składników



Sign up Log out

**Cheese**

Fats	33g
Proteins	25g
Carbohydrates	1.3g
Calories	371

**Pasta**

Fats	1.1g
Proteins	5g
Carbohydrates	25g
Calories	131

**Tomato puree**

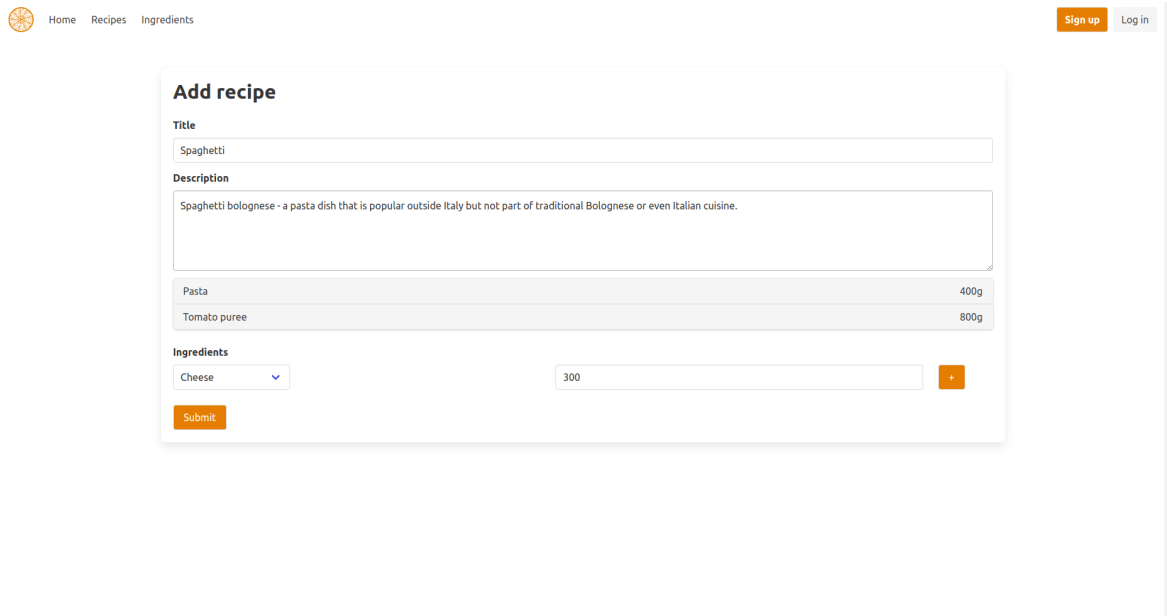
Fats	0.2g
Proteins	1.7g
Carbohydrates	9g
Calories	38

**Potato**

Fats	0.1g
Proteins	2g
Carbohydrates	17g
Calories	77

Każdy może wyświetlić wszystkie dostępne składniki i ich parametry

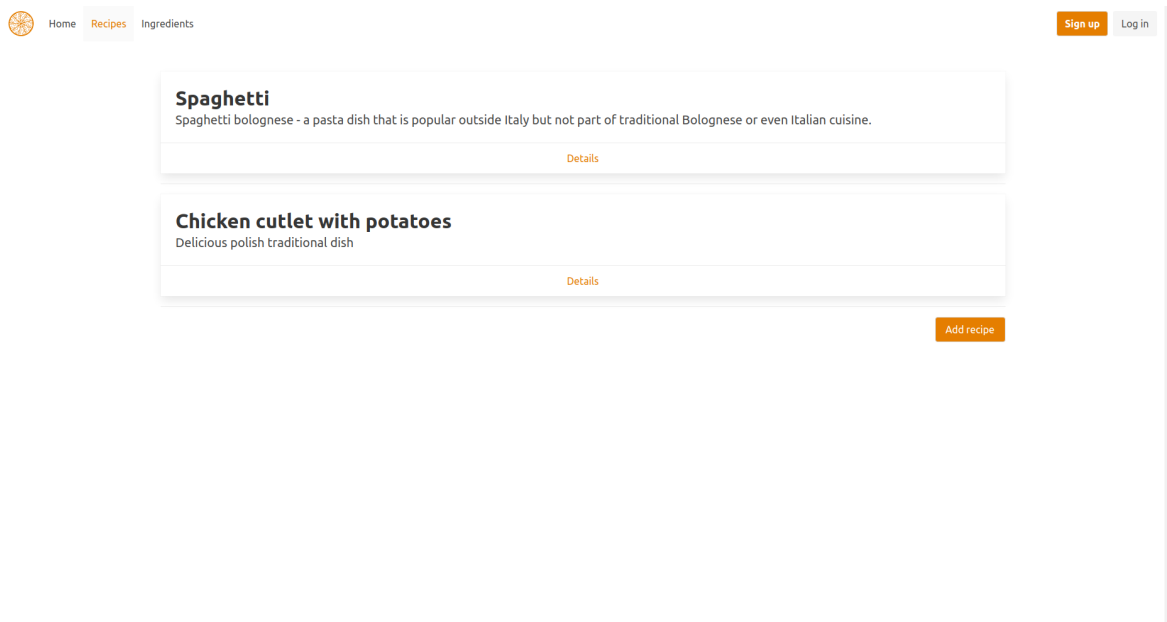
## 4.3 Dodawanie przepisu



The screenshot shows a web application interface for adding a recipe. At the top, there is a navigation bar with a logo (a stylized orange circle with a cross) and links for 'Home', 'Recipes', and 'Ingredients'. On the right side of the navigation bar, there are two buttons: 'Sign up' (orange) and 'Log in' (grey). The main content area is titled 'Add recipe'. It contains several input fields: a 'Title' field with the text 'Spaghetti', a 'Description' field with the text 'Spaghetti bolognese - a pasta dish that is popular outside Italy but not part of traditional Bolognese or even Italian cuisine.', a table for ingredients with two rows: 'Pasta' with a quantity of '400g' and 'Tomato puree' with a quantity of '800g', and an 'Ingredients' section with a dropdown menu showing 'Cheese' and a quantity input field with the value '300'. There is a small orange button with a plus sign next to the quantity input. At the bottom of the form, there is a 'Submit' button (orange).

Uwierzytelniony użytkownik może dodać przepis podając jego tytuł, opis oraz wybrane z listy składniki wraz z zadeklarowanymi ilościami

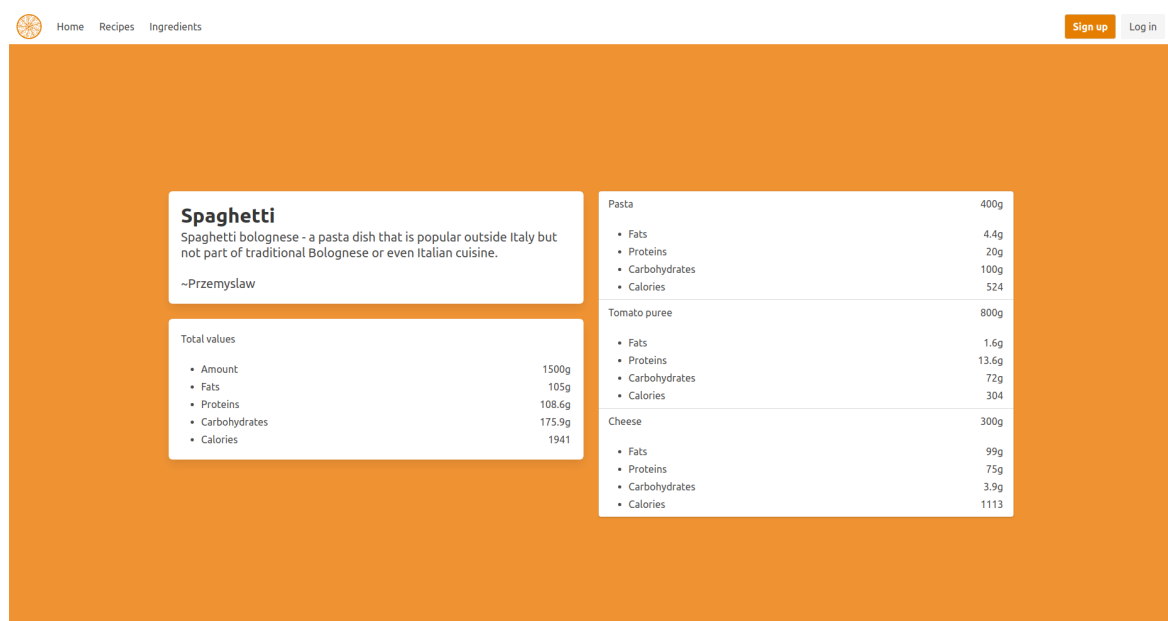
## 4.4 Wyświetlanie listy przepisów



The screenshot shows a web application interface displaying a list of recipes. At the top, there is a navigation bar with a logo (a stylized orange circle with a cross) and links for 'Home', 'Recipes', and 'Ingredients'. On the right side of the navigation bar, there are two buttons: 'Sign up' (orange) and 'Log in' (grey). The main content area displays a list of recipes. The first recipe is 'Spaghetti' with the description 'Spaghetti bolognese - a pasta dish that is popular outside Italy but not part of traditional Bolognese or even Italian cuisine.' Below the description is a 'Details' link (orange). The second recipe is 'Chicken cutlet with potatoes' with the description 'Delicious polish traditional dish'. Below the description is a 'Details' link (orange). At the bottom right of the list, there is an 'Add recipe' button (orange).

Każdy może wyświetlić listę przepisów - ich tytuły i opisy

## 4.5 Wyświetlanie szczegółów przepisu



**Spaghetti**  
Spaghetti bolognese - a pasta dish that is popular outside Italy but not part of traditional Bolognese or even Italian cuisine.  
~Przemyslaw

**Total values**

• Amount	1500g
• Fats	105g
• Proteins	108.6g
• Carbohydrates	175.9g
• Calories	1941

**Pasta** 400g

- Fats 4.4g
- Proteins 20g
- Carbohydrates 100g
- Calories 524

**Tomato puree** 800g

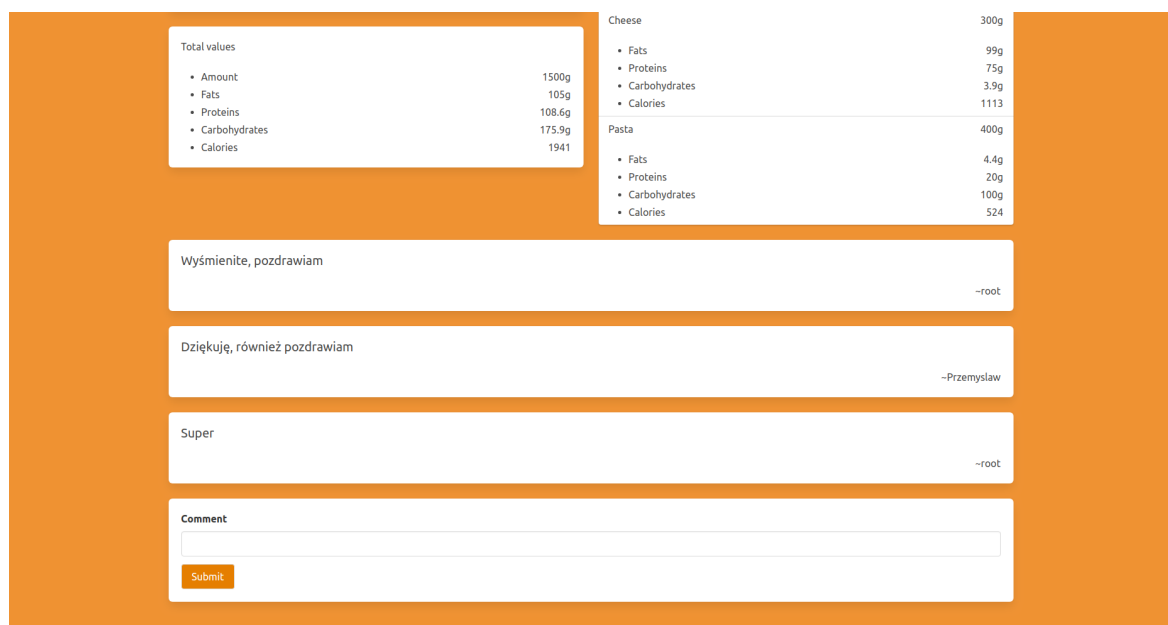
- Fats 1.6g
- Proteins 13.6g
- Carbohydrates 72g
- Calories 304

**Cheese** 300g

- Fats 99g
- Proteins 75g
- Carbohydrates 3.9g
- Calories 1113

Każdy może wyświetlić szczegóły przepisu - zawartość wartości odżywczych w danej ilości konkretnego składnika oraz w całym posiłku

## 4.6 Wyświetlanie i dodawanie komentarzy do przepisu



**Total values**

• Amount	1500g
• Fats	105g
• Proteins	108.6g
• Carbohydrates	175.9g
• Calories	1941

**Cheese** 300g

- Fats 99g
- Proteins 75g
- Carbohydrates 3.9g
- Calories 1113

**Pasta** 400g

- Fats 4.4g
- Proteins 20g
- Carbohydrates 100g
- Calories 524

Wyśmienite, pozdrawiam  
~root

Dziękuję, również pozdrawiam  
~Przemyslaw

Super  
~root

**Comment**

Komentarze mogą zostać wyświetlone przez każdego jednak tylko zalogowany użytkownik może dodać nowy komentarz

## 5 Wnioski

- Dostępne obecnie narzędzia znacznie ułatwiają pracę z bazami danych, nie oznacza to jednak że można całkowicie obejść się bez znajomości relacyjnych baz danych. Pomimo interfejsu dostosowanego do języka programowania, którego używamy, nadal najistotniejszym elementem aplikacji bazodanowej jest przemyślana architektura bazy danych.



- Oddzielenie warstwy widoku zapewnia dużą swobodę implementacji oraz łatwiejszą skalowalność aplikacji internetowej.
- Aby przekształcić bazę danych w pełnoprawną i przydatną aplikację wystarczy rozszerzyć ją o kontrolę dostępu (system uprawnień) i przystępny protokół do komunikacji z użytkownikiem (w tym wypadku GUI)

## 6 Możliwości rozwoju

Jako, że aplikacja posiada rozdzielone bazę danych, serwer i aplikację po stronie klienta łatwo jest połączyć ją z innymi aplikacjami w podobnej tematyce oraz dodać nowe funkcje.

- Wykorzystanie api w aplikacji pomagającej w trzymaniu diety, liczeniu kalorii
- Rozszerzenie funkcjonalności aplikacji o dopasowywanie przepisów do dietetycznych wymagań użytkownika
- Dodanie integracji z mediami społecznościowymi np.: udostępnianie przepisów