Сайт за рецепти

Курсов проект

Изготвили:
Петър Николаев 19118,
Цветина Томова 19126,
Теодор Пацов 19122,
Теодора Недялкова 19123

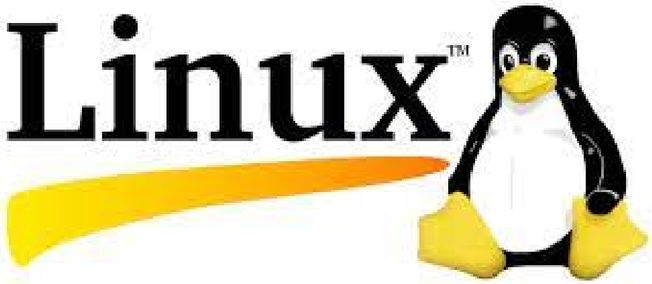
Идея на сайта:

Целта на този сайт е хората лесно да намират и споделят своите рецепти с останалите потребители. В него може да добавяте ваши рецепти и да използвате тези на другите. Той е направен със сигурност за потребителя и данните му остават лични. Сайта за улеснение не е само на български, а има вариант и на английски език.

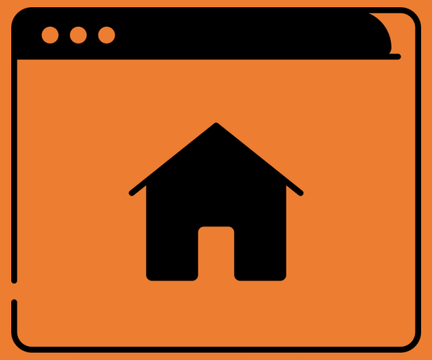
Технологии на сайта:

За него сме използвали:

- PHP
- MYSQL
- HTML5
- CSS3
- JAVASCRIPT
- BOOTSTRAP
- APACHE
- LINUX



Начална страница:

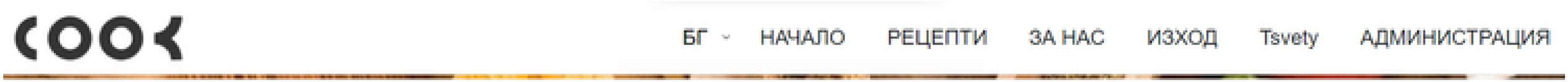


Страницата е първото нещо, което виждате в сайта. Отгоре ще забележите навигационна лента. В центъра на страницата ще намерите бутон, който ви отвежда към категориите на рецептите, разделени за Ваше удобство. Най-отдолу се намира информация за свързване със създателите или за съобщаване на грешки.

Навигационна лента:

На лентата има 6 бутона:

- Първият е за избор на език. За подразбиране е български
- Вторият е за отвеждане към началната страница
- Третият е същински, защото те отвежда към рецептите
- Четвъртият те отвежда към информация за създателите
- Петият и шестият са Login, Register form на сайта, ако в момента потребителя не е в акаунта си. В другия случай имаме бутон за изход, бутон за профила на потребителя и ако той е админ имаме администрация,



Вход

Имейл?

Потребителско име

Парола

Вход

Все още не сте член? Регистрация?

Назад към Начало

Регистрация

Потребителско име

Имейл (незадължително)

Парола

Потвърдете парола

Впиши

Вече сте член? Вход

Назад към Начало

Login Form:

Изисквана информация за потребителя:

Чрез бутон потребителя има право да избере дали да влез чрез имейл или потребителско име

Второто поле е за парола изискваща минимум 8 символа

Под паролата се намира бутона за влизане

Също под тези бутони се намира линк отвеждащ към формата за регистрация, ако нямаш акаунт

Следващият линк те отвежда към началната страница

Register Form

Изисквана информация за потребителя:

- Потребителско име

- Имейл, който не е задължителен

- Парола поне 8 символа

- Потвърждение на паролата

- Бутон за създаване на профил, който те препраща към Login Form

- Също под тези бутони се намира линк отвеждащ към формата за вписване, ако имаш акаунт

- Следващият линк те отвежда към началната страница

Профил:

- Пази се потребителско име
- Имейл - по избор
- Парола
- Роля
- Всичко това се пази в базата данни
- Има страница за редактиране на профила

Recipe

Съдържа информация за рецептите:

1. Рейтинг на рецептата
2. Създател
3. Дата на качване
4. Инструкции за приготвяне
5. Необходими продукти

Авторът може да премахне рецептата.
Има също формуляр за качване на рецепти, в който се пишат данните за рецептата.

Admin

Ако си админ се показава бутон в навигационната лента. В него има таблица, която съдържа информация за всички потребители. Админа има право да изтрие профилите на потребителите, които не са админи

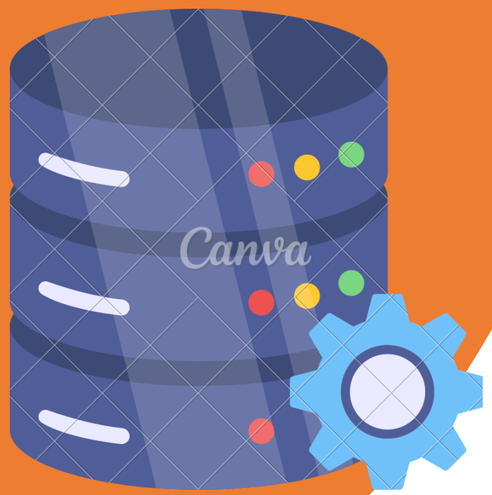
AboutUs

- Съдържанието на страницата е разделено на основна секция (main), която съдържа информация за екипа на проекта.
- Информацията за екипа е представена в четири колони, като за всяка колона се показва снимка, име на член от екипа и икони за свързване със социални медии (Facebook, GitHub, Instagram).

Utils

Класът "Utils" предоставя няколко статични метода за работа със сесии, база данни и други операции. Тези методи включват взимане на потребител, добавяне на потребител, задаване на потребител, взимане на текущ потребител, извеждане на грешка и взимане на рецепта. Класът съдържа и метод "session", който стартира сесия, ако такава не е стартирана.
Класът "Recipe" представя модел на рецепта, който съдържа различни свойства като идентификатор, рейтинг, име, автор, дата и час на създаване, съставки, стъпки и категория.

dbconn



В началото се извличат информацията за базата данни от JSON файл, наречен "info.json", намиращ се в папката "../models/".
Съдържанието на файла се чете и декодира като JSON, а след това се достъпват параметрите на базата данни, като хост, име на базата данни, потребителско име и парола. Тези параметри се използват за създаване на стринг за свързване с базата данни (DSN - Data Source Name).
След това се създава нов обект от класа PDO, като се подават DSN, потребителско име и парола на конструктора. Това създава връзка с базата данни, която може да се използва за изпълнение на заявки към нея.

lang

Този код има следните функционалности:

1. Четене на съдържанието на JSON файл с преводи на различни езици от папката "../models/languages.json".
2. Декодиране на JSON формата в асоциативен масив чрез функцията **json_decode()**.
3. Взимане на избрания език от заявката **$_GET["lang"]** и използване на него като ключ за достъп до съответния превод от асоциативния масив.
4. Ако преводът за избрания език не е намерен (променливата **$dict** е **null**), се използва българският език като резервен вариант за превод.
5. Резултатът от превода се съхранява в променливата **$dict**, която може да се използва в по-нататъшния код за изведение на преведен текст на уеб страницата.

Този код осигурява функционалност за превод на текстови елементи на уеб страницата на различни езици, като използва JSON файл за съхранение на преводите и параметър от заявката **$_GET** за избор на езика.

add_recipe_controller

PHP кодът обработва формуляр за добавяне на рецепта. Той извършва следните действия:

1. Проверява дали потребителят е влезнал в системата, като проверява наличието на сесионна променлива "id".
2. Проверява дали всички полета на формуляра (име на рецептата, съставки, инструкции и категория) са попълнени.
3. Ако някое от полетата липсва или е празно, извежда грешка.
4. Ако всички полета са попълнени, вмъква данните за рецептата в базата данни.
5. Създава JSON файл, който съдържа съставките и инструкциите за рецептата.
6. Пренасочва потребителя към страницата за преглед на рецепти.
7. Ако потребителят не е влезнал в системата, се извежда съобщение за забранено действие.

delete_account_controller

Кодът включва файловете "dbconn.php", "utils.php" и "lang.php".
"lang.php" съдържа информация за езика на сайта.
Следва извикване на статичния метод "session()" от класа "Utils". Това служи за стартиране или управление на сесията.
Кодът проверява дали са зададени полетата "password" и "id" в $_POST и $_SESSION съответно. Това означава, че формата за изтриване на профила е изпратена и потребителят е в сесия.
Изпълнението на заявката връща резултат като многомерен масив $res.
Ако резултатът не съдържа точно един ред, се извежда грешка.

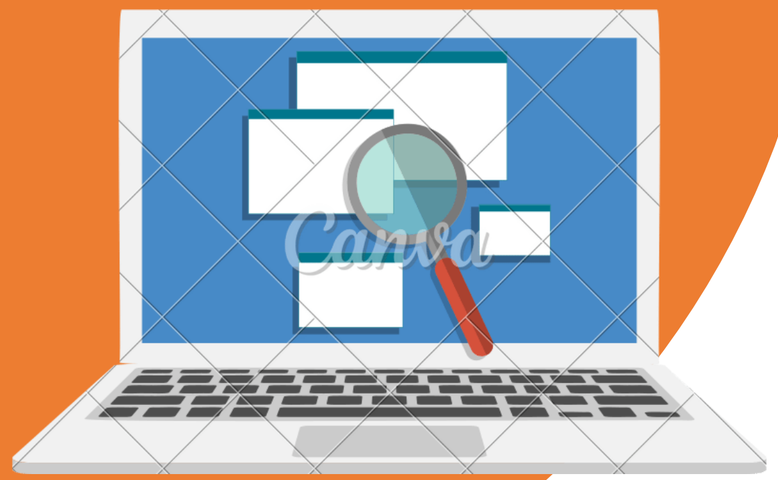
delete_recipe

Включва необходимите файлове, включително "dbconn.php", за да се осъществи връзка с базата данни.
Проверява дали е предоставено полето "id" в $_POST, което вероятно се получава от формуляр за изтриване на рецепта.
Съставя SQL заявка за изтриване на рецептата от таблицата "recipes" в базата данни, използвайки стойността на $_POST["id"].
Изпълнява SQL заявката за изтриване на рецептата чрез $pdo->query($sql).
Опитва се да изтрие свързаният JSON файл, използвайки функцията unlink() и пътя до файла.
Накрая, потребителят се пренасочва към страницата за преглед на рецепти чрез header("Location: ../view/recipes.php").

delete_user

Кодът включва файлът "dbconn.php", който създава връзка с базата данни.
Проверява се дали е предоставено полето "id" в $_POST. Това вероятно означава, че е изпратена форма за изтриване на потребителски профил.
Ако полето "id" е предоставено, се съставя SQL заявка за изтриване на профила от таблицата "profiles" в базата данни, използвайки стойността на $_POST["id"].
Изпълнява се SQL заявката за изтриване на профила чрез $pdo->query($sql).
Накрая, потребителят се пренасочва към страницата за администриране (admin.php) чрез header("Location: ../view/admin.php").

display_recipe



1. Включва необходимите файлове, включително "dbconn.php", "utils.php" и "lang.php", за да се осъществи връзка с базата данни и да се извлечат помощни функции и езикови преводи.
2. Изпълнява заявка към базата данни, за да вземе рецептата със съответното идентификационно число ($_GET["id"]).
3. Ако възникне резултат, т.е. има само една рецепта с това идентификационно число, извлича необходимата информация за рецептата.
4. Увеличава рейтинга на рецептата с 1 и актуализира базата данни.
5. Генерира HTML страница, която показва изображението, името, подробности и инструкциите на рецептата.
6. Ако сесията е активна и текущият потребител е автор на рецептата, се показва форма за изтриване на рецептата. Ако не е той създателя, не се появява бутон за изтриване.

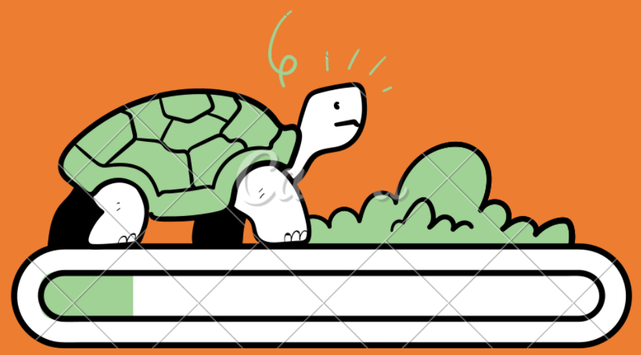
edit_controller

1. Включва необходимите файлове, включително "dbconn.php" и "utils.php", за да се осъществи връзка с базата данни и да се използват помощни функции.
2. Изпълнява заявка към базата данни, за да провери дали съществува потребител със съответното идентификационно число ($_SESSION["id"]).
3. Ако резултатът е различен от 1, се извежда грешка.
4. В противен случай, се проверяват и обновяват въведените данни за потребителското име, имейл адрес и парола, ако те са били предоставени.
5. Проверките включват валидация на входните данни, като се използват регулярни изрази и вградени функции за валидация.
6. При успешна актуализация на данните, се извежда съобщение за успех.

get_recipes

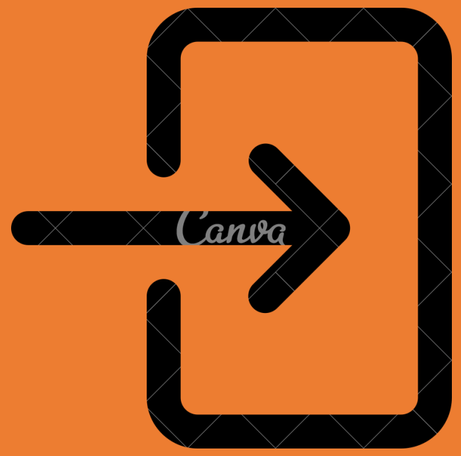
Включва необходимите файлове, включително "utils.php" и "lang.php", за да се използват помощни функции и езикови настройки.
Проверява наличието на параметър "lang" в URL адреса и генерира съответния код за връщане на същия URL адрес с добавен или актуализиран параметър за езика.
Опитва се да получи информация за рецепт със зададеното идентификационно число, използвайки функцията Utils::get_recipe.
В зависимост от категорията на рецептата, се избира подходящо изображение за показване.
Генерира HTML код, който включва заглавие на рецептата, изображение и резултат за оценка.
Ако възникне грешка по време на изпълнението на функцията, се извиква функцията Utils::error_message с код 15 за обработка на грешката.

load_recipes



Включва необходимите файлове, включително "get_recipes.php", "dbconn.php" и "utils.php", които осигуряват връзка с базата данни и помощни функции.
Проверява наличието на параметър "cat" в URL адреса. Ако параметърът "cat" е зададен, кодът извършва избор на рецепти в зависимост от категорията.
Ако параметърът "cat" е "personal", се изпълнява подготвена заявка към базата данни, която избира всички уникални рецепти, сортирани на случаен принцип, които са създадени от текущия потребител (ако има активна сесия). Резултатът се съхранява в променливата $arr.
Ако параметърът "cat" не е "personal", се изпълнява подготвена заявка към базата данни, която избира всички уникални рецепти от определената категория, сортирани на случаен принцип и с ограничен брой (максимум 8). Резултатът се съхранява в променливата $arr.
Ако параметърът "cat" не е зададен, се изпълнява подготвена заявка към базата данни, която избира случайни 8 уникални рецепти от всички налични. Резултатът се съхранява в променливата $arr.
След това се извършва цикъл, който обхожда всеки елемент в $arr и извиква функцията generate_recipe, която генерира HTML код за показване на информацията за рецептата.

login_controller



Включва необходимите файлове за връзка с базата данни, за полезни функции и за езика.
Проверява дали е подадена парола чрез isset($_POST["password"]).
Проверява дали е подадено потребителско име и дали е непразно чрез isset($_POST["username"]) && strlen($_POST["username"]) != 0.
Изпълнява заявката към базата данни, за да провери дали съществува акаунт със зададеното потребителско име. Резултатът се съхранява в променливата $res.
Проверява дали има само един резултат в масива $res, което означава, че съществува акаунт със зададеното потребителско име. Ако не е изпълнено това условие, се извежда съобщение за грешка.
Проверява дали подадената парола отговаря на хешираната парола в базата данни, използвайки функцията password_verify(). Ако паролите съвпадат, се изпълняват следните действия:
Се стартира сесия с помощта на функцията Utils::session().
Се задават стойности на променливите на сесията, като $_SESSION["id"], $_SESSION["username"], $_SESSION["role"] и $_SESSION["email"].
Ако паролите не съвпадат, се извежда съобщение за грешка.
Процеса е същия и за имейла

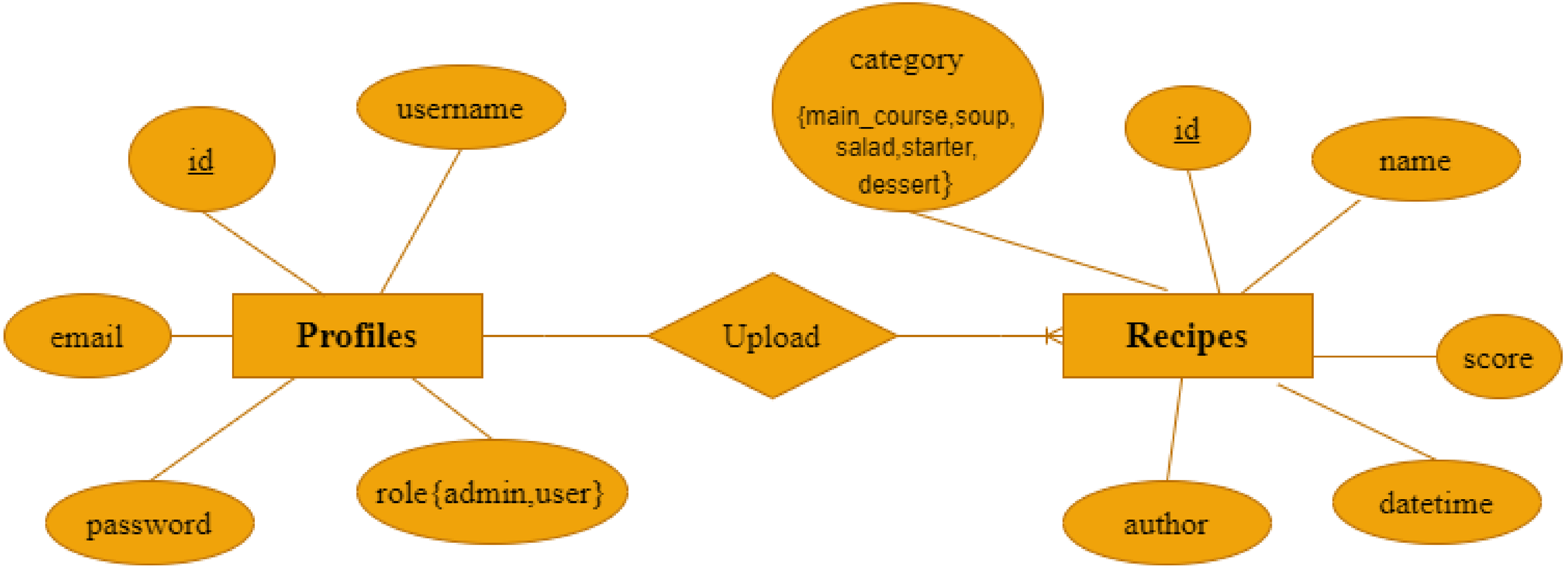
register_controller

Проверява дали са подадени стойности за потребителско име, електронна поща, парола и потвърждение на паролата, използвайки isset($_POST["username"]) && isset($_POST["email"]) && isset($_POST["password"]) && isset($_POST["confirm"]).
Изпълнява заявката към базата данни, за да провери дали вече съществува потребител със същото потребителско име или електронна поща. Резултатът се съхранява в променливата $res.
Проверява различни условия, за да валидира въведените данни:
Проверява дали са попълнени всички полета, използвайки strlen($_POST["username"]) > 0 && strlen($_POST["password"]) > 0 && strlen($_POST["confirm"]) > 0.
Проверява дали вече съществува потребител със същото потребителско име или електронна поща, използвайки count($res) != 0.
Проверява валидността на потребителското име и електронната поща, използвайки регулярни изрази и функцията filter_var().
Проверява дали паролата отговаря на потвърждението на паролата, използвайки $_POST["password"] != $_POST["confirm"].
Ако всички проверки преминат успешно, изпълнява заявката към базата данни, за да се създаде нов профил. Паролата се хешира преди да бъде записана в базата данни.
Стартира сесия с помощта на функцията Utils::session().
Извършва пренасочване към страницата "login.php" с включен езиков код $lang_code.

signout

Включва необходимите файлове за полезни функции и за езика.
Стартира сесия с помощта на Utils::session(), която инициализира сесията, ако все още не е стартирана.
Изтрива сесията с помощта на session_destroy(), която премахва всички данни, свързани с текущата сесия.
Извършва пренасочване към началната страница, указана с "Location: ..", което ще пренасочи потребителя към главната страница или кореновата директория на уеб приложението.

ER-диаграма


category
{main_course,soup,
salad,starter,
dessert}
id
name
username
id
email
Profiles
Upload
Recipes
score
password
role{admin,user}
author
datetime

Create Table заявки

```
-- Table structure for table `profiles`

DROP TABLE IF EXISTS `profiles`;
CREATE TABLE `profiles` (
  `id` int NOT NULL
AUTO_INCREMENT,
  `username` varchar(16) NOT NULL,
  `email` varchar(32) DEFAULT NULL,
  `password` varchar(64) NOT NULL,
  `role` enum('admin','user') NOT
NULL DEFAULT 'user',
  PRIMARY KEY (`id`),
  UNIQUE KEY `username_UNIQUE`
(`username`))
```

```
-- Table structure for table `recipes`

DROP TABLE IF EXISTS `recipes`;
CREATE TABLE `recipes` (
  `id` int unsigned NOT NULL
AUTO_INCREMENT,
  `name` varchar(64) NOT NULL,
  `author` int NOT NULL,
  `score` int NOT NULL DEFAULT '0',
  `datetime` datetime DEFAULT NULL,
  `category`
enum('main_course','soup','salad','starter','des
sert') DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `author_idx` (`author`),
  CONSTRAINT `author` FOREIGN KEY
(`author`) REFERENCES `profiles` (`id`) )
```

CRUD операции

- $sql = "DELETE FROM profiles WHERE id LIKE " . $_POST["id"] . ";";
- $pdo->prepare("INSERT INTO profiles (username, email, password) VALUES (?,?,?);")
- $pdo->prepare("UPDATE profiles SET username = ? WHERE id = ?;")
- $pdo->prepare("UPDATE profiles SET email = ? WHERE id = ?;")
- $pdo->prepare("UPDATE profiles SET password = ? WHERE id = ?;")

- $pdo->query("UPDATE recipes SET score = $score WHERE id = $id;")
- $sql = "DELETE FROM profiles WHERE id LIKE " . $_POST["id"] . ";"
- $sql = "DELETE FROM recipes WHERE id LIKE " . $_POST["id"] . ";"
- $sql = "DELETE FROM profiles WHERE id LIKE " . $_SESSION["id"] . ";"
- $pdo->prepare("INSERT INTO recipes (name, author, datetime, category) VALUES (?,?,?,?)")
-

SELECT заявки

- $pdo->query("SELECT id, username, email, recipes_count, password, role FROM profiles LEFT JOIN (SELECT COUNT(id) as recipes_count, author FROM recipes GROUP BY author HAVING COUNT(id) > 0) AS t ON t.author LIKE profiles.id;")
- $pdo->query("SELECT * FROM recipes WHERE id LIKE $id;")
- $pdo->query("SELECT username FROM profiles WHERE id LIKE $author_id;")
- $pdo->prepare("SELECT id, username, role FROM profiles WHERE username LIKE ? OR email LIKE ? AND email NOT LIKE ";")
- $pdo->prepare("SELECT id, password, role, email FROM profiles WHERE username LIKE ?;")
- $pdo->prepare("SELECT id, username, password, role FROM profiles WHERE email LIKE ?;")
- $pdo->prepare("SELECT DISTINCT * FROM recipes WHERE author LIKE ? ORDER BY RAND();")
- $pdo->prepare("SELECT DISTINCT * FROM recipes WHERE category LIKE ? ORDER BY RAND() LIMIT 8;")
- $pdo->query("SELECT DISTINCT * FROM recipes ORDER BY RAND() LIMIT 8;")

SELECT заявки

- $pdo->prepare("SELECT id FROM profiles WHERE id LIKE ?;")
- $pdo->prepare("SELECT * FROM recipes WHERE id LIKE ?;")
- $pdo->prepare("SELECT password FROM profiles WHERE id LIKE ?;")
- $pdo->query("SELECT id FROM recipes WHERE author LIKE $user ORDER BY id DESC LIMIT 1;")


#Thank's For
Watching!