My CookBook – popis implementace

# Úvod

V tomto dokumentu jsou popsány hlavní rysy webové aplikace My CookBook z hlediska implementace.

Serverová část této aplikace je implementována v jazyku PHP a klientská část v jazycích HTML, CSS a Javascript. Pro práci s daty byla použita databáze MySQL, díky čemuž je tato aplikace snadno přenositelná a snadno editovatelná. Ke správě dat byl použit nástroj **Adminer**.

Všechny použité soubory je možné nalézt na <https://github.com/OBU5/MyCookBook/>

Veškeré použité .php soubory jsou přímo v adresáři „***MyCookBook***“. Soubor *„styles.css*“ je v adresáři „**MyCookBook/Styles**“ a skripty v javascriptu jsou v souboru „**MyCookBook/Scripts**“. Použité příkazy ke generování databáze je pak možné nalézt v adresáři „**MyCookBook/Database“**.

# Popis webové aplikace "

## Popis použitých souborů

**Soubory php**

* php soubory nám na serveru umožňují generování HTML kódu podle dat v databázi a podle parametrů zadaných uživatelem.

**Soubory CSS**

* Soubory CSS upravují styl vygenerovaného HTML kódu. Pro snadnou úpravu a rozšiřitelnost je v této webové aplikaci pouze jeden soubor CSS ve složce **Styles** s názvem „styles.css“.

**Soubory JS**

* Soubory JS umožňují zejména kontrolu formulářových polí, což uživateli rychle napoví, co udělal špatně. Kontrola v této webové aplikaci probíhá jednak při opuštění vstupního pole (metoda „blur“), ale také při odeslání formuláře, kdy zamezí odeslání a vypíše uživateli chybovou hlášku. To jednak zmenší datový tok mezi klientem a serverem, ale také zpohodlní práci klientovi.
* Dále byl javascript použit k přidání HTML kódu, konkrétně k možností dynamického rozšíření tabulky ingrediencí.

## Správa uživatelů

* **editLogin.php**
  + generuje HTML kód, který umožňuje změnu uživatelských údajů
  + Probíhá zde
    - kontrola velikosti textového řetězce pro všechna vstupní pole
    - ověření unikátního uživatelského jména
    - Hashování hesla s osolením (pokud je nové)
    - Uložení zadaných údajů do databáze
* **login.php**
  + generuje HTML kód, který umožňuje registraci případně přihlášení uživatele (podle GET parametru)
  + Při registraci dochází ke
    - kontrole velikosti textového řetězce pro všechna vstupní pole
    - Hashování hesla s osolením
    - Uložení zadaných údajů
  + Při přihlášení dochází k
    - Zjištění správnosti údajů
    - Pokud jsou údaje správně, dojde k přihlášení
      * To probíhá přiřazením hodnoty do proměnné „$\_SESSION["username"]“.
  + V případě, že jsou údaje chybné, je formulář zaslán uživateli stále vyplněný (kromě hesla)
* **logout.php**
  + Odhlášení uživatele vynulováním hodnoty „$\_SESSION["username"]“
  + Presměrování na hlavní stránku
* **userInfo.php**
  + generuje HTML kód, který umožňuje zobrazení informací o přihlášeném uživateli
  + lze se odtud dostat (stiskem tlačítka) na stránku editLogin.php a upravit informace o přihlášeném uživateli.
* **viewAllUserInfo.php**
  + generuje HTML kód, který umožňuje zobrazení informací o všech uživatelích. Tuto stránku mohou zobrazit pouze uživatelé s rolí „Admin“
  + Obsah je rozdělen na stránky, kde číslo stránky se předává pomocí metody GET
* **deleteLogin.php**
  + Pokud na to má přihlášený uživatel práva, dojde ke smazání uživatele s daným ID.
* **Scripts/checkLoginForm.js**
  + Kontroluje, že jsou všechny prvky formuláře správně vyplněny posloucháním eventu „blur“ a „klik“ na odesílací tlačítko
  + V případě chybného vyplnění napíše do elementu s id „errorMessage“ chybovou hlášku

## Správa receptů

* **addRecipe.php**
  + generuje HTML kód, který umožňuje přidání nového receptu
  + Probíhá zde
    - Kontrola, zda je uživatel přihlášen
    - kontrola velikosti textového řetězce pro všechna vstupní pole
    - ověření unikátního jména receptu
    - Uložení zadaných údajů do databáze
  + V případě, že jsou údaje chybné, je formulář zaslán uživateli stále vyplněný (kromě hesla)
* **deleteRecipe.php**
  + Pokud na to má přihlášený uživatel práva, dojde ke smazání receptu s daným ID.
* **editRecipe.php**
  + generuje HTML kód, který umožňuje editaci receptu se zadaným id
  + Probíhá zde
    - Kontrola, zda je uživatel přihlášen
    - Zda má uživatel práva změnit recept
    - kontrola velikosti textového řetězce pro všechna vstupní pole
    - ověření unikátního jména receptu
    - Uložení zadaných nových údajů do databáze a odstranění starých údajů
  + V případě, že jsou údaje chybné, je formulář zaslán uživateli stále vyplněný (kromě hesla)
* **viewAllRecipes.php**
  + generuje HTML kód, který umožňuje zobrazení informací o všech receptech.
  + Obsah je rozdělen na stránky, kde číslo stránky se předává pomocí metody GET
* **viewRecipe.php**
  + generuje HTML kód, který umožňuje zobrazení detailních informací o receptu s ID předaným jako parametr metody GET
* **Scripts/checkRecipeForm.js**
  + Kontroluje, že jsou všechny prvky formuláře správně vyplněny posloucháním eventu „blur“ a „klik“ na odesílací tlačítko
  + V případě chybného vyplnění napíše do elementu s id „errorMessage“ chybovou hlášku
  + Při kliknutí na tlačítko „přidat další ingredienci“ přidá další řádek do tabulky ingrediencí a umožní tak použít dostatečný počet ingrediencí.

## Ostatní

* **about.php**
  + generuje HTML kód, který umožňuje zobrazení informací o této stránce, za jakým účelem vznikla, jaké jazyky byly použity.
* **changeStyle.php**
  + nastaví cookies „style“ na hodnotu odpovídající zvolenému stylu
* **index.php**
  + generuje HTML kód, který umožňuje zobrazení informací s popisem, co je možné na stránce dělat.

# Popis správy dat

## Popis použitých souborů

Pro vytvoření databáze byl vytvořen skript „**CreateTable.sql**“. Databáze pak byla zpočátku plněna pomocí skriptu „**SimpleInsert.sql**“, nicméně po editaci dat a řádné úpravě byla databáze exportována a pro vygenerování defaultní databáze je možné použít kód „**runThis – create default table.sql**“.

## Tabulky

V aplikaci jsou ukládány 2 „struktury“, **uživatelé** a **recepty**. Uživatelé jsou uloženy v jedné samostatné tabulce a struktura receptů je uloženy v 6 různých tabulkách.

* Tabulky
  + **Users**
    - Tabulka uživatelů
  + **Recipes**
    - Tabulka receptů
  + **MealCategory** 
    - Tabulka kategorií jídla
  + **Recipe\_MealCategory**
    - Tabulka, ve které jsou uloženy vztahy mezi receptem a kategorií jídla
    - Slouží k tomu, aby bylo možné k jednomu receptu přiřadit více kategorií jídla
  + **OriginCountry**
    - Tabulka zemí původu
  + **Ingredients**
    - Tabulka ingrediencí
  + **Recipe\_Ingredients**
    - Tabulka, ve které jsou uloženy vztahy mezi receptem a ingrediencí
    - Slouží k tomu, aby bylo možné k jednomu receptu přidat více ingrediencí

### Příkazy pro vygenerování tabulek

Před vytvořením je nejprve potřeba, v případě že již tabulky existují, tyto tabulky odstranit.

DROP TABLE IF EXISTS Recipe\_Ingredients;

DROP TABLE IF EXISTS Ingredients;

DROP TABLE IF EXISTS OriginCountry;

DROP TABLE IF EXISTS Recipe\_MealCategory;

DROP TABLE IF EXISTS MealCategory;

DROP TABLE IF EXISTS Recipes;

DROP TABLE IF EXISTS Users;

### Tabulka uživatelů

* obsahuje:
  + **ID** – primární klíč, který jednoznačně identifikuje daného uživatele a automaticky zvyšuje svou hodnotu. Při přidání nového uživatele je tedy hodnota ID vždy o 1 větší než ID posledního uživatele
  + **Jméno**
  + **Příjmení**
  + **uživatelské jméno**
  + **email**
  + **heslo**

CREATE TABLE Users (

    ID INT NOT NULL AUTO\_INCREMENT,

    name VARCHAR(40) NOT NULL,

    lastname VARCHAR(40),

    email VARCHAR(40),

    username VARCHAR(40),

    password VARCHAR(100) NOT NULL,

    role VARCHAR(40),

    PRIMARY KEY(ID)

);

### Tabulka receptů

* obsahuje:
  + ID
  + jméno receptu
  + postup
  + datum přidání
  + id autora – tedy uživatele, který tento recept vytvořil
  + id země původu
  + odkaz na obrázek

CREATE TABLE Recipes (

    ID int AUTO\_INCREMENT NOT NULL ,

    name VARCHAR(40) NOT NULL,

    directions VARCHAR(1000) NOT NULL,

    date Date NOT NULL,

    author\_id int,

    originCountry\_id int,

    imgUrl VARCHAR(1000),

    PRIMARY KEY (ID),

    FOREIGN KEY (author\_id) REFERENCES Users(ID)

);

### Tabulka zemí původu

* obsahuje:
  + ID
  + Jméno země

CREATE TABLE OriginCountry (

    ID int AUTO\_INCREMENT NOT NULL ,

    name VARCHAR(50) NOT NULL,

    PRIMARY KEY (ID)

);

### Tabulka kategorií jídla

* obsahuje:
  + ID
  + jméno kategorie

CREATE TABLE MealCategory (

    ID int AUTO\_INCREMENT NOT NULL ,

    name VARCHAR(50) NOT NULL,

    PRIMARY KEY (ID)

);

### Tabulka ingrediencí

* obsahuje:
  + ID
  + jméno ingredience
  + množství ingredience
  + jednotky ingredience

CREATE TABLE Ingredients (

    ID int AUTO\_INCREMENT NOT NULL,

    name varchar(40) NOT NULL,

    quantity varchar(40) NOT NULL,

    unit varchar(40) NOT NULL,

    PRIMARY KEY (ID)

);

### Tabulka vztahu mezi kategorií jídla a receptem

* obsahuje:
  + ID
  + ID kategorie jídla, které patří k vybranému receptu
  + ID receptu, ke kterému patří vybraná kategorie jídla
* Slouží k uložení více kategorií jídla k danému receptu

CREATE TABLE Recipe\_MealCategory (

    ID int AUTO\_INCREMENT NOT NULL,

    mealCategory\_id int NOT NULL,

    recipe\_id int NOT NULL,

    PRIMARY KEY (ID),

    FOREIGN KEY (mealCategory\_id) REFERENCES MealCategory(ID),

    FOREIGN KEY (recipe\_id) REFERENCES Recipes(ID)

);

### Tabulka vztahu mezi ingrediencí a receptem

* obsahuje:
  + ID
  + ID ingredience, která patří k vybranému receptu
  + ID receptu, ke kterému patří vybraná ingredience
* Slouží k uložení více ingrediencí do receptu

CREATE TABLE Recipe\_Ingredients (

    ID int AUTO\_INCREMENT NOT NULL ,

    recipe\_id int NOT NULL,

    ingredient\_id int NOT NULL,

    PRIMARY KEY (ID),

    FOREIGN KEY (recipe\_id) REFERENCES Recipes(ID),

    FOREIGN KEY (ingredient\_id) REFERENCES Ingredients(ID)

);

## Práce s tabulkou

Pro práci s tabulkou byly využity příkazy

* **SELECT** pro výběr dat
* **INSERT** pro vkládání dat
* **UPDATE** pro změnu dat
* **DELETE** pro mazání dat

Dále pro specifikaci daných dat v tabulce byly použity příkazy

* **FROM** pro výběr tabulky
* **WHERE** pro specifikaci podmínek, za kterých data chceme
* **LIMIT** pro limitaci získaných prvků (použito pro výpis na stránku)

### Příklad vkládání dat do tabulky

Vložení kategorií jídla

INSERT INTO `mealcategory` (`ID`, `name`) VALUES

(1, 'Předkrm'),

(2, 'Polévka'),

(3, 'Hlavní chod'),

(4, 'Dezert'),

(5, 'Svačina');

### Příklad výběru dat dat

Výběr uživatelů, kde „username“ odpovídá hodnotě „JAN“

SELECT \* FROM users WHERE username = 'JAN'