

## Struktura zdrojového kódu

Aplikace se skládá z několika hlavních modulů a knihoven. Každý modul má jasně definovanou odpovědnost.

## Hlavní moduly a knihovny

### Knihovny

#### 1. data\_lib.php

- Knihovna pro správu poznámek.
- Obsahuje funkce pro CRUD operace nad poznámkami, ukládání poznámek do souborů a jejich načítání.
- **Klíčové funkce:**
  - loadNotes() - Načítá poznámky ze souboru.
  - saveNotes(\$notes) - Ukládá poznámky do souboru.
  - addNoteToUser(\$username, \$title, \$content) - Vytváří novou poznámku.
  - deleteNote() - Maže poznámku uživatele.
  - getPaginatedNotes() - Stránkování poznámek.

#### 2. users\_lib.php

- Knihovna pro správu uživatelů.
- Obsahuje funkce pro CRUD operace nad uživateli, autentizaci a validaci dat.
- **Klíčové funkce:**
  - loadUsers() - Načítá seznam uživatelů.
  - saveUsers(\$users) - Ukládá seznam uživatelů.
  - addUser(\$username, \$password, \$profile\_picture) - Přidává nového uživatele.
  - promoteToAdmin(\$username) - Proměňuje uživatele na administrátora.
  - deleteUserAcc(\$username) - Odstraňuje účet uživatele.

### Frontendové soubory

#### 1. index.php

- Domovská stránka aplikace.
- Zobrazuje seznam poznámek uživatele (nebo z cookies u nepřihlášených).

#### 2. login.php

- Přihlašovací stránka.
- Obsahuje formulář pro zadání uživatelského jména a hesla.

#### 3. register.php

- Registrační stránka.
- Umožňuje uživatelům vytvořit nový účet včetně nahrání profilové fotky.

#### 4. view\_note.php

- Detailní zobrazení konkrétní poznámky.

## 5. update\_note.php

- Rozhraní pro úpravu existující poznámky.

## 6. redact\_profile\_page.php

- Rozhraní pro úpravu uživatelských údajů (uživatelské jméno, avatar, heslo).

## 7. Admin rozhraní:

- admin\_panel.php - Přehled všech uživatelů a jejich poznámek.
- admin\_view\_user.php - Zobrazení poznámek konkrétního uživatele pro administrátora.

# Klíčové algoritmy a jejich implementace

## 1. CRUD operace nad poznámkami

Operace pro vytváření, čtení, úpravu a mazání poznámek jsou definovány v knihovně data\_lib.php.

**Načtení poznámek:**

```
/**
 * Načte poznámky konkrétního uživatele a seřadí je podle data vytvoření.
 *
 * @param string $username Uživatelské jméno.
 * @return array Pole poznámek.
 */
function getNotesForUsername(string $username): array {
    $notes = loadNotes();
    if (!isset($notes[$username])) {
        return [];
    }

    uasort($notes[$username], function($a, $b) {
        $timeA = isset($a['created_at']) ? strtotime($a['created_at']) : 0;
        $timeB = isset($b['created_at']) ? strtotime($b['created_at']) : 0;
        return $timeB - $timeA;
    });

    return $notes[$username];
}
```

**Popis:** Funkce vrací seznam poznámek uživatele seřazený podle data vytvoření (nejnovější jako první).

- Vytvoření poznámky:

```
/**
 * Přidá poznámku konkrétnímu uživateli.
 *
 * @param string $username Uživatelské jméno.
 * @param string $title Název poznámky.
 * @param string $content Obsah poznámky.
 * @return string Unikátní identifikátor přidané poznámky.
 */
function addNoteToUser(string $username, string $title, string $content): string {
    $notes = loadNotes();

    if (!isset($notes[$username])) {
        $notes[$username] = [];
    }

    $noteID = uniqid();
    $notes[$username][$noteID] = [
        'title' => $title,
        'content' => $content,
        'created_at' => date('Y-m-d H:i:s'),
    ];

    saveNotes($notes);

    return $noteID;
}
```

**Popis:** Funkce zajišťuje přidání poznámky konkrétnímu uživateli a generuje unikátní identifikátor.

- Redagování poznámek:

```
<div id="update-note-container">
    <form action="save_update.php" method="post">
        <input type="text" name="title" id="title" value="<?php echo htmlspecialchars($note['title']); ?>" required>

        <textarea name="content" id="content" required><?php echo htmlspecialchars($note['content']); ?></textarea>

        <input type="hidden" name="note_id" value="<?php echo htmlspecialchars($noteID); ?>">
        <input type="hidden" name="username" value="<?php echo htmlspecialchars($username === 'admin' ? $targetUsername : $username); ?>">

        <button type="submit">Save</button>
    </form>
    <a href="<?php echo htmlspecialchars($backUrl); ?>" id="back">Back</a>
</div>
```

**Popis:** Jedná se o rozhraní pro úpravu profilu uživatele (např. změna uživatelského jména, nahrání nového avatara).

- Obnovení poznámek:

```
/**
 * Upraví existující poznámku konkrétního uživatele.
 *
 * @param string $username Uživatelské jméno.
 * @param string $noteID Identifikátor poznámky.
 * @param array $updated_note Aktualizovaný obsah poznámky.
 * @return void
 */
function saveNoteForUser(string $username, string $noteID, array $updated_note): void {
    $notes = loadNotes();
    if (isset($notes[$username][$noteID])) {
        $existing_note = $notes[$username][$noteID];
        $updated_note['created_at'] = $existing_note['created_at'];
        $notes[$username][$noteID] = array_merge($existing_note, $updated_note);
        saveNotes($notes);
    } else {
        echo "Error: Note not found";
    }
}
```

**Popis:** Upraví existující poznámku konkrétního uživatele.

- Mazání poznámek:

```
/**
 * Odstraní poznámku konkrétního uživatele.
 *
 * @param string $username Uživatelské jméno.
 * @param string $noteID Identifikátor poznámky.
 * @return void
 */
function deleteNote(string $username, string $noteID): void {
    $notes = loadNotes();
    if (isset($notes[$username][$noteID])) {
        unset($notes[$username][$noteID]);
        saveNotes($notes);
    } else {
        echo "Error: Note not found";
    }
}
```

**Popis:** Odstraní poznámku konkrétního uživatele.

## 2. CRUD operace nad uživateli

Operace pro správu uživatelů jsou implementovány v knihovně users\_lib.php.

- **Přidání uživatele:**

```
/**
 * Přidá nového uživatele do seznamu.
 *
 * @param string $username Uživatelské jméno.
 * @param string $password Heslo uživatele.
 * @param string $profile_picture Cesta k profilovému obrázku.
 * @return void
 */
function addUser(string $username, string $password, string $profile_picture): void {
    $users = listUsers();
    $hashedPassword = password_hash($password, PASSWORD_BCRYPT);
    $users[] = [
        "username" => $username,
        "password" => $hashedPassword,
        "profile_picture" => $profile_picture,
        "id" => uniqid(),
        "admin" => false
    ];
    saveUsers($users);
}
```

**Popis:** Funkce přidává nového uživatele s hashovaným heslem a odkazem na profilovou fotku.

- **Proměnění na administrátora:**

```
/**
 * Propaguje uživatele na administrátora.
 *
 * @param string $username Uživatelské jméno.
 * @return void
 */
function promoteToAdmin(string $username): void {
    $users = listUsers();
    foreach ($users as &$user) {
        if ($user['username'] === $username) {
            $user['admin'] = true;
            break;
        }
    }
    saveUsers($users);
}
```

**Popis:** Funkce změní atribut admin uživatele na hodnotu true.

- Mazání uživatele

```
/**
 * Odstraní uživatelský účet.
 *
 * @param string $username Uživatelské jméno.
 * @return void
 */
function deleteUserAcc(string $username): void {
    $users = loadUsers();
    foreach ($users as $index => $user) {
        if ($user['username'] === $username) {
            unset($users[$index]);
            break;
        }
    }
    saveUsers(array_values($users));
}
```

**Popis:** Odstraňuje záznam o uživateli ze systému.

## Validace dat

**Validace uživatelského jména:**

- Délka uživatelského jména musí být mezi 3 a 50 znaky.
- Uživatelské jméno musí být unikátní (kontrola proti existujícím uživatelům).

```
// --- Validace uživatelského jména ---
if (strlen($username) < 3 || strlen($username) > 50) {
    $_SESSION['error'] = 'Username must be between 3 and 50 characters.';
    header('Location: register.php');
    exit;
}

// --- Kontrola existence uživatele ---
if (getUserByUsername($username)) {
    $_SESSION['error'] = 'User already exists.';
    header('Location: register.php');
    exit;
}
```

## Validace hesla:

- Heslo musí mít minimálně 8 znaků a obsahovat alespoň jedno písmeno a jedno číslo.

```
// --- Validace hesla ---
if (strlen($password) < 8 || !preg_match('/\d/', $password) || !preg_match('/[a-zA-Z]/', $password)) {
    $_SESSION['error'] = 'Password must be at least 8 characters long and include at least one letter and one number.';
    header('Location: register.php');
    exit;
}
```

```
// --- Kontrola shody hesel ---
if ($password !== $confirm_password) {
    $_SESSION['error'] = 'Passwords do not match.';
    header('Location: register.php');
    exit;
}
```

## Struktura databázových souborů

### 1. users.json:

- Obsahuje informace o uživateli.
- Struktura:

```
stralka > Task Manager > {} users.json > ...
[{"username": "admin", "password": "$2y$10$9wq1260ykMR46NRRiDqt8.AgGwCu.kKktaX51cX",
{"username": "tester",
"password": "$2y$10$NGxTmIyqzeZLI6ujTKFAv.\5gAgShUXmWQ3KViDIxKiESinw62Pz.",
"profile_picture": "uploads/converted_677b75dd60d59-test.png",
"id": "677b75dd7bd56", "admin": false}
]
```

### 2. data.json:

- Obsahuje poznámky uživatelů.
- Struktura:

```
stralka > Task Manager > {} data.json > ...
{"testikula": {"677b6b558dffd": {"title": "1234", "content": "qwewrt12345",
"tester":
{"677b8c8fd26f0": {"title": "ahoj!!", "content": ":"}),
"created_at": "2025-01-06 07:55:59"}}}
```

## Zpracování a zmenšení obrázků

```
/**
 * Zpracování a zmenšení obrázku.
 * @var string $convertedImagePath Cesta pro zmenšený obrázek.
 */
$convertedImagePath = $uploadDir . 'converted_' . $newFileName;

list($originalWidth, $originalHeight) = getimagesize($fileTmpPath);
$newWidth = 150;
$newHeight = 150;

$sourceImage = null;
if ($fileType === 'image/jpeg') {
    $sourceImage = imagecreatefromjpeg($fileTmpPath);
} elseif ($fileType === 'image/png') {
    $sourceImage = imagecreatefrompng($fileTmpPath);
} elseif ($fileType === 'image/gif') {
    $sourceImage = imagecreatefromgif($fileTmpPath);
}

if ($sourceImage) {
    $resizedImage = imagecreatetruecolor($newWidth, $newHeight);
    imagecopyresampled(
        $resizedImage,
        $sourceImage,
        0, 0, 0, 0,
        $newWidth, $newHeight,
        $originalWidth, $originalHeight
    );
}
```



```

if ($sourceImage) {
    $resizedImage = imagecreatetruecolor($newWidth, $newHeight);
    imagecopyresampled(
        $resizedImage,
        $sourceImage,
        0, 0, 0, 0,
        $newWidth, $newHeight,
        $originalWidth, $originalHeight
    );

    if ($fileType === 'image/jpeg') {
        imagejpeg($resizedImage, $convertedImagePath, 90);
    } elseif ($fileType === 'image/png') {
        imagepng($resizedImage, $convertedImagePath, 9);
    } elseif ($fileType === 'image/gif') {
        imagegif($resizedImage, $convertedImagePath);
    }

    imagedestroy($sourceImage);
    imagedestroy($resizedImage);

    $destPath = $convertedImagePath;
} else {
    $_SESSION['error'] = 'There was an error processing the uploaded file.';
    header('Location: redact_profile_page.php');
    exit;
}

```

## Soubory s podporou JavaScriptu a CSS

- **css/styles.css:** Obsahuje styly pro frontend aplikace.
- **js/script.js:** Obsahuje funkce pro zpracování interakcí uživatele, jako je AJAX načítání poznámek nebo zobrazení/skrytí formulářů.
- **Js/print.js:** Tato funkce umožňuje uživateli tisknout konkrétní poznámku přímo ze stránky.
- **js/confirm.js:** Skript pro ověření nebezpečných akcí, jako je smazání účtu nebo poznámky.
- **js/reg\_validation.js:** Skript pro validaci registračního formuláře