

PROGRAMÁTORSKÁ DOKUMENTACE - EDULOG

Aplikace: EduLog - Studijní plánovač

Verze: 1.0

Technologie: PHP 8.1, HTML5, CSS3, JavaScript (ES6), JSON data storage

1. ÚVOD A ARCHITEKTURA

Aplikace je navržena jako "Single-User Multi-Tenant" systém, kde data všech uživatelů sídlí na jednom serveru, ale jsou logicky oddělena. Architektura je postavena na modulárním procedurálním PHP bez použití frameworků.

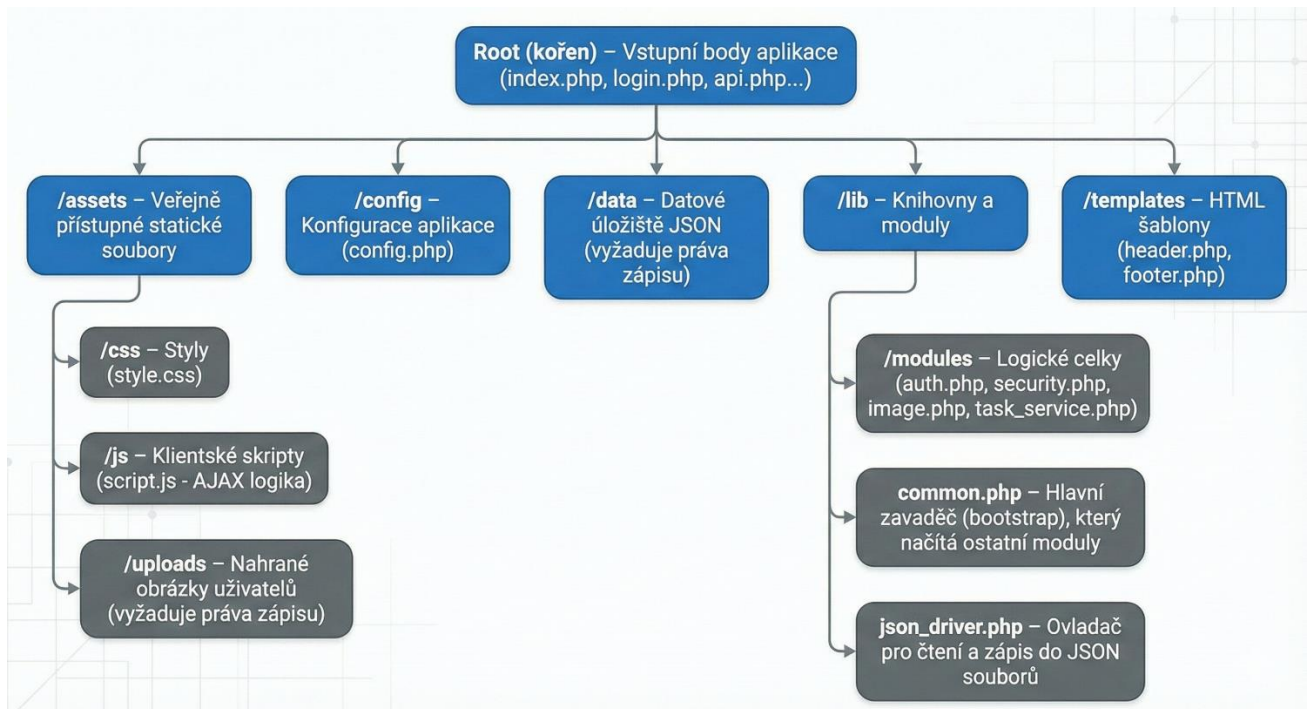
Jádro aplikace se skládá ze tří vrstev:

1. **Frontend (View):** HTML šablony v adresáři /templates a statické soubory (CSS/JS) v /assets.
2. **Backend (Logic):** PHP skripty v kořenovém adresáři (tzv. "Controllery") a pomocné knihovny ve složce /lib.
3. **Data (Model):** JSON soubory v adresáři /data, které suplují relační databázi.

1.1 Adresářová struktura

- **/assets** – Veřejně přístupné statické soubory
 - /css – Styly (style.css)
 - /js – Klientské skripty (script.js - AJAX logika)
 - /uploads – Nahrané obrázky uživatelů (vyžaduje práva zápisu)
- **/config** – Konfigurace aplikace (config.php)
- **/data** – Datové úložiště JSON (vyžaduje práva zápisu)
- **/lib** – Knihovny a moduly
 - /modules – Logické celky (auth.php, security.php, image.php, task_service.php)
 - common.php – Hlavní zavaděč (bootstrap), který načítá ostatní moduly
 - json_driver.php – Ovladač pro čtení a zápis do JSON souborů

- **/templates** – HTML šablony (header.php, footer.php)
- **Root (kořen)** – Vstupní body aplikace (index.php, login.php, api.php...)



2. INSTALAČNÍ MANUÁL

Pro bezpečné zprovoznění aplikace na produkčním serveru (Apache/Nginx) dodržujte následující postup:

2.1 Kopírování souborů

Nahrajte obsah projektu do kořenového adresáře webhostingu (např. public_html, htdocs nebo www).

2.2 Nastavení oprávnění (Permissions)

Aplikace potřebuje zapisovat do složek /data a /assets/uploads. Doporučené nastavení: Nastavte vlastníka složky na uživatele, pod kterým běží webový server (např. www-data) a práva na 755 (rwxr-xr-x). Alternativní nastavení: Pokud nemůžete měnit vlastníka, nastavte práva na 777 (rwxrwxrwx). Varování: Nastavení 777 umožní zápis komukoliv v systému. Používejte pouze pokud je to nezbytné nebo pro testování.

2.3 Zabezpečení dat (.htaccess)

Složka /data obsahuje citlivé informace. Je nutné zamezit přímému stahování souborů přes prohlížeč. Ve složce /data vytvořte soubor .htaccess s obsahem:

Deny from all

Tím zajistíte, že k datům bude mít přístup pouze PHP skript, nikoliv návštěvník webu.

2.4 Konfigurace

Otevřete soubor /config/config.php a ověřte nastavení cesty k datům. Konstanta DATA_DIR musí obsahovat absolutní cestu ke složce s JSON soubory.

Příklad správného nastavení: `define('DATA_DIR', DIR . '/../data/');`

3. DATOVÝ MODEL (JSON SCHEMA)

Data jsou ukládána do textových souborů ve formátu JSON. Integrita dat při zápisu je zajištěna pomocí zámků souborů (LOCK_EX) implementovaných v souboru lib/json_driver.php.

3.1 Uživatelé (users.json)

```
.json

[
  {
    "id": 1,
    "email": "student@vse.cz",
    "password": "$2y$10$...",
    "role": "user"
  }
]
```

Ukládá identitu uživatelů. Hesla jsou hashována algoritmem Bcrypt pomocí funkce password_hash.

(Poznámka: Role může nabývat hodnot "user" nebo "admin".)

3.2 Úkoly (tasks.json)

Hlavní entita aplikace. Vazba na uživatele je realizována přes klíč user_id. Volitelná vazba na předmět je přes subject_id.

```
.json

[
  {
    "id": 101,
    "user_id": 1,
    "subject_id": 5,
    "title": "Semestrální práce",
    "description": "Napsat dokumentaci",
    "due_date": "2025-01-20",
    "status": "pending",
    "image": "654a...png"
  }
]
```

(Poznámka: Status může být "pending" = nevyřízeno, nebo "completed" = hotovo. Položka image obsahuje název souboru, nebo null.)

(Poznámka 2: Vysvětlení formátu data: due_date (string): Datum je ukládáno ve formátu ISO 8601 (YYYY-MM-DD), např. "2025-12-31". Tento formát je zvolen z důvodu snadného chronologického řazení a kompatibility s HTML input type="date". Pro zobrazení

uživateli je datum v PHP (task_service.php) konvertováno do českého formátu (d.m.Y.).

3.3 Předměty (subjects.json)

Číselník předmětů vázaný na konkrétního uživatele.



```
[  
  {  
    "id": 5,  
    "user_id": 1,  
    "name": "Matematika"  
  }  
]
```

4. API ROZHRANÍ A AJAX

Aplikace obsahuje interní API endpoint `api.php`, který vrací filtrovaná data ve formátu JSON. Tento endpoint využívá frontendový JavaScript (`script.js`) pro asynchronní operace (filtrování bez reloadu).

Endpoint: GET `/api.php`

Parametry:

- `status` (string): Filtrování podle stavu (`pending` / `completed`).
- `subject_id` (int): Filtrování podle ID předmětu.

Příklad odpovědi (JSON):



```
[
  {
    "id": 101,
    "title": "Semestrální práce",
    "days_left": 5,
    "is_overdue": false,
    "thumb_url": "assets/uploads/thumb_img.png"
  }
]
```

5. POPIS KLÍČOVÝCH KOMPONENT

5.1 Bezpečnost (lib/modules/security.php)

- **XSS (Cross-Site Scripting):** Veškerý výstup dat do HTML šablony prochází funkcí `h()`, která volá `htmlspecialchars()`.
- **CSRF (Cross-Site Request Forgery):**
 1. Při startu session se generuje unikátní token funkcí `generate_csrf()`.
 2. Všechny odesílací formuláře obsahují tento token ve skrytém poli (hidden input).
 3. Odkazy na kritické akce (např. smazání) obsahují token v URL (`delete.php?token=...`).
 4. Před provedením akce se validuje shoda tokenů funkcemi `verify_csrf()` nebo `verify_csrf_token()`.

5.2 Zpracování obrázků (lib/modules/image.php)

Při uploadu obrázku v souborech `create.php` nebo `edit.php` probíhá následující proces:

1. Obrázek se uloží v plné kvalitě na server.
2. Funkce `make_thumb()` zkontroluje formát a pomocí grafické knihovny GD vytvoří zmenšeninu (šířka 300px).
3. Script zachovává průhlednost u formátů PNG a GIF.

5.3 Filtrování a logika úkolů (lib/modules/task_service.php)

Veškerá logika pro manipulaci s daty úkolů je centralizována. Funkce `get_filtered_tasks` zajišťuje, že uživatel vidí pouze svá data. Funkce `enrich_task_data` dynamicky dopočítává zbývající dny do termínu a určuje CSS třídy pro zobrazení (např. červené zvýraznění pro úkoly po termínu).

5.4 Mazání dat (admin.php / delete.php)

Mazání je implementováno kaskádově pro zachování konzistence dat. Při smazání uživatele:

1. Systém vyhledá všechny úkoly patřící danému uživateli.
2. Fyzicky se smažou soubory obrázků (originál i náhled) z disku.
3. Uživatel a jeho záznamy v JSON souborech jsou trvale odstraněny.