

VidFlow Fejlesztői Dokumentáció

Ez a dokumentum a VidFlow projekt technikai részleteit, architektúráját és beállítási folyamatát ismerteti.

1. Bevezetés

A VidFlow egy webes alkalmazás, amely lehetővé teszi a felhasználók számára, hogy videókat böngésszenek (elsősorban a YouTube API segítségével), megtekintsenek, kedveljenek, lejátszási listákba rendezzenek, és tesztre szabják a felület megjelenését. A rendszer felhasználói fiókokat kezel regisztrációval, bejelentkezéssel (saját rendszer és Google Sign-In), jelszó-visszaállítással és adminisztrációs felülettel.

Főbb technológiák:

- **Backend:** PHP (8.0+ verzió)
- **Adatbázis:** MySQL
- **Frontend:** HTML, CSS (egyedi stílusok + Bootstrap), JavaScript
- **API-k:** YouTube Data API v3, Google Sign-In
- **Függőségkezelés:** Composer

2. Telepítés és Beállítás

Előfeltételek:

- Webszerver (pl. Apache, Nginx) PHP támogatással (javasolt PHP 8.0 vagy újabb)
- MySQL vagy MariaDB adatbázis-szerver
- Composer

Telepítési lépések:

1. **Kód letöltése:** Klónozd vagy töltsd le a projekt forráskódját a webszerver megfelelő könyvtárába.
2. **Függőségek telepítése:** A projekt gyökérkönyvtárában futtasd a `composer install` parancsot. Ez letölti a `composer.lock` fájlban rögzített függőségeket (pl. PHPMailer, Google API kliens, Dotenv) a `vendor` mappába. A `node.js` szerver futtatásához 'express' szükséges ami az `'npm i express'` paranccsal telepíthető

• **Webszerver konfigurálása:** Állítsd be a webszerver

• **Adatbázis létrehozása:** Hozz létre egy MySQL adatbázist a projekt számára, és importáld bele a `vidflow.sql` fájl tartalmát, ez létrehozza a szükséges táblákat.

- **Környezeti változók beállítása:** Hozz létre egy `.env` fájlt a projekt gyökérkönyvtárában.

- **Mappa jogosultságok:** Győződj meg róla, hogy a `/pictures` mappa írható a webszerver felhasználója számára, mivel ide kerülnek a feltöltött profilképek.

3. Projekt Struktúra

- `/` (gyökér): Fő belépési pont (`index.php`), aloldalak (`watch.php`, `liked_videos.php`, `profile.php` stb.), kliensoldali scriptek (`script.js`) és stíluslapok (`style.css`, `stylemobile.css`, `profilestyle.css`).
- `admin/`: Adminisztrációs funkciók (`usermanagement.php`).
- `login/`: Bejelentkezés (`login.php`), jelszó visszaállítás (`forgot_password.php`), Kliensoldali script (`script.js`).
- `register/`: Felhasználói regisztráció (`register.php`), email megerősítés (`verify.php`).
- `services/`: Backend API végpontok AJAX kérésekhez, ezek kezelik a felhasználói interakciókat (kedvelés, lejátszási lista, profilfrissítés stb.)
- `vendor/`: Composer által telepített külső PHP könyvtárak
- `pictures/`: Feltöltött profilképek tárolási helye
- `pageelements/`: Statikus UI elemek (képek, alapértelmezett profilkép)

4. Backend (PHP)

- **Alap Működés:** Az oldalak (`index.php`, `watch.php` stb.) PHP segítségével generálják a HTML kimenetet. A felhasználói állapotot (bejelentkezés) PHP sessionök (`$_SESSION`) segítségével követik nyomon. Az oldalak elején általában betöltik a `.env` fájlt és a Composer autoloadert (`vendor/autoload.php`).
- **Adatbázis Kezelés:**
 - o Vegyesen használ MySQLi és PDO kiterjesztéseket az adatbázis-műveletekhez
 - o A kapcsolat részleteit a `.env` fájlból olvassa be a `vlucas/phpdotenv` segítségével.
 - o Az adatbázis séma (`vidflow.sql`) a következő fő táblákat tartalmazza:
 - **users:** Felhasználói adatok (ID, felhasználónév, hash-elt jelszó, email, név, profilkép elérési útja, létrehozás dátuma, csomag (plan), `style_code`, email megerősítés állapota, megerősítő/jelszó-visszaállító tokenek).
 - **customization:** Testreszabási profilok (`style_code`, betűtípus, háttérszín, videókeret színei). Tartalmaz egy default stílust.
 - **liked_videos, playlists, watching_history:** Felhasználóhoz kötött listák.
 - **payment:** Az előfizetés adatait tartalmazza,
- **API Végpontok (services/)**

Ezek a PHP scriptek AJAX kérésekre válaszolnak, általában JSON formátumban vagy egyszerű szöveges üzenettel/HTML kóddal. Mindegyik ellenőrzi a felhasználói munkamenetet (isset(\$_SESSION['username_in'])).

- **add_video_to_playlist.php:** Videót ad hozzá egy meglévő lejátszási listához (string összefűzés).
- **create_playlist.php:** Új lejátszási listát hoz létre a felhasználónak.
- **customization_update.php:** Profil testreszabás (képfeltöltés, stílus (betűtípus, színek) mentése/frissítése/visszaállítása). Kezeli a customization táblát és a users.style_code, users.profilepic mezőket. Frissíti a session változókat is.
- **get_playlists.php:** Lekérdezi és HTML-ként visszaadja a felhasználó lejátszási listáit.
- **liked_video_add.php:** Videót ad a kedvencekhez (string összefűzés a liked_videos táblában).
- **liked_video_remove.php:** Videót távolít el a kedvencekből (str_replace használatával).
- **liked_videos_get.php:** Lekérdezi a kedvelt videókat és JSON formátumban adja vissza.
- **userdata_handling.php:** Kezeli a felhasználói adatok (felhasználónév, név, email) lekérdezését (GET) és módosítását (POST). Ellenőrzi az egyediséget. Felhasználónév váltásakor kijelentkezteti a felhasználót.
- **userdata_passwordchange.php:** Jelszó módosítása (hash-eléssel).
- **watchinghistory_add.php:** Videót ad a megtekintési előzményekhez (string összefűzés).
- **watchinghistory_get.php:** Lekérdezi a megtekintési előzményeket és JSON formátumban adja vissza.

· **Függőségek (composer.json, composer.lock):**

- phpmailer/phpmailer: Email küldés (regisztráció, jelszóemlékeztető).
- vlucas/phpdotenv: .env fájlok kezelése.
- google/apiclient: Google API-k használata (YouTube Data API v3, Google Sign-In).

· **Hitelesítés és Biztonság:**

- Session alapú bejelentkezés
- Jelszó tárolása hash-elve
- Google Sign-In integráció
- Email cím megerősítése token segítségével
- Jelszó-visszaállítás token alapú
- A services/ végpontok ellenőrzik a session meglétét

5. Frontend (HTML/CSS/JS)

- **HTML:** Dinamikusan generált PHP által, tartalmazza a videólejátszót (YouTube IFrame API), videólistákat, felhasználói felület elemeit.
- **CSS:** Több CSS fájl (style.css, stylemobile.css, profilestyle.css) felel a megjelenésért. A felhasználó által beállított színek és betűtípusok dinamikusan, valószínűleg inline

stílusok vagy session alapján betöltött CSS változók segítségével kerülnek alkalmazásra.

- **JavaScript (script.js, login/script.js, oldalspecifikus inline scriptek):**
 - Felhasználói felület vezérlése (menük, panelek megjelenítése/elrejtése).
 - AJAX kérések indítása a services/ végpontok felé (pl. fetch API használatával) a háttérbeli műveletekhez (kedvelés, hozzáadás lejátszási listához, profil mentése).
 - YouTube IFrame Player API inicializálása és vezérlése (watch.php).
 - Google Sign-In gomb inicializálása és a kapott token kezelése.
 - Videó link másolása vágólapra.

6. API-k és Külső Szolgáltatások

- **YouTube Data API v3:** A google/apiclient segítségével hívják meg a videók kereséséhez (search), népszerű videók listázásához (videos.list a mostPopular charttal), és videó részleteinek lekéréséhez. A YOUTUBE_API_KEY szükséges a .env fájlban.
- **Google Sign-In:** A Google Identity Services JavaScript library segítségével valósul meg. Google Client ID szükséges.
- **SMTP:** A PHPMailer könyvtár használja emailek küldéséhez. Az SMTP szerver adatait a .env fájlból olvassa (SMTP_* változók).
- **Node.js express szerver, yt-dlp, ffmpeg (node/server.js):** Ez felel a videók lejátszásáért, ytstream.barnatech.hu domain-en fut ez a szerver, amely ha kap egy YouTube link-et ?url= paraméteren yt-dlp-vel elkezd streamelni a felhasználónak a videót a legjobb hang-és képminőségben

7. Konfiguráció (.env)

A projekt működéséhez elengedhetetlen a .env fájl megfelelő beállítása a gyökérkönyvtárban. A szükséges változók:

- **DB_SERVERNAME:** Adatbázis szerver címe.
- **DB_USERNAME:** Adatbázis felhasználónév.
- **DB_PASSWORD:** Adatbázis jelszó.
- **DB_NAME:** Adatbázis neve.
- **SMTP_HOST:** SMTP Szerver hostneve
- **SMTP_USERNAME:** SMTP felhasználónév
- **SMTP_PASSWORD:** SMTP jelszó
- **SMTP_PORT:** SMTP port
- **SMTP_FROM_EMAIL:** Küldő email címe
- **SMTP_FROM_NAME:** Küldő neve
- **YOUTUBE_API_KEY_1:** Youtube Data API v3 kulcs
- **YOUTUBE_API_KEY_2:** Youtube Data API v3 kulcs
- **YOUTUBE_API_KEY_3:** Youtube Data API v3 kulcs

Cseréld le az értékeket a saját beállításaidra.

***A három YouTube Data API v3 kulcs használata nem feltétlenül szükséges, csupán a redundancia és a megbízhatóság növelése érdekében alkalmazzuk őket.**

8. Adatbázis Séma

- **users:** Központi felhasználói tábla. id (**PK**), username, password (**hash**), email, name, profilepic (elérési út), created_at, plan, style_code (FK a customization-hoz), is_verified, token, reset_token, reset_expiry.
- **customization:** Stílusbeállítások. style_code (egyedi kulcs), fontFamilyType, backgroundColor, videoborderColor (pontosvesszővel elválasztott két szín).
- **liked_videos:** Kedvelt videók. username (FK a users-hez), video_url (pontosvesszővel elválasztott ID-k), video_title (pontosvesszővel elválasztott címek), video_length, video_uploader, video_uploader_pic (pontosvesszővel elválasztott adatok).
- **playlists:** Lejátszási listák. username (FK a users-hez), playlist_name, playlist_videos (pontosvesszővel elválasztott videó ID-k).
- **watching_history:** Megtekintési előzmények. username (FK a users-hez), video_url (pontosvesszővel elválasztott teljes URL-ek).
- **payment:** Előfizetés információk, id (**PK**), username (FK a users-hez), price, last_payment_date, next_payment_date, inactivity_date.

9. Admin Felület

Az admin/ mappában található usermanagement.php szkript biztosítja az alkalmazás felhasználóinak adminisztrációs felületét.

Elérhetőség:

- o A szkript feltételezi, hogy a felhasználó be van jelentkezve (ellenőrzi a `$_SESSION["username_in"]` meglétét). Bár a kódrészlet explicit módon nem ellenőrzi az admin státuszt (`$_SESSION["userplan_status"] == 3`), az oldal funkciói és elhelyezkedése alapján ez feltételezhető a hozzáférési ponton (valószínűleg a profile.php vagy a főoldalon lévő menüben).
- o Tipikusan egy iframe-ben jelenik meg a felhasználói profil oldalon keresztül.

Funkciók:

- o **Felhasználók Listázása:** Csatlakozik az adatbázishoz (a .env fájl alapján) és lekérdezi az összes regisztrált felhasználót az users táblából. Megjeleníti a felhasználók ID-ját, felhasználónevét, nevét, email címét és csomagját (plan) egy HTML táblázatban. Az adatok kimenetkor htmlspecialchars-el vannak védve az XSS támadások ellen.
- o **Felhasználó Törlése:** Minden felhasználó sorában található egy "Törlés" link. Ez egy GET kérést indít az oldalra a delete_username paraméterrel. A PHP kód ezt érzékeli, és egy előkészített utasítással (`DELETE FROM users WHERE username = ?`) törli a megadott felhasználót az adatbázisból. A linkre kattintva

JavaScript confirm() kér megerősítést a művelet előtt. Sikerről vagy hibáról üzenetet jelenít meg.

- o **Adminisztrátorrá Tétel:** Azoknál a felhasználóknál, akiknek a plan értéke nem 3, megjelenik egy "Hozzáadás" (Adminná tétel) link. Ez egy GET kérést indít a make_admin_username paraméterrel. A PHP kód ezt feldolgozza, és egy előkészített utasítással (UPDATE users SET plan = 3 WHERE username = ?) a felhasználó plan mezőjét 3-ra állítja. Itt is JavaScript confirm() kér megerősítést. Sikerről vagy hibáról üzenetet jelenít meg. Ha egy felhasználó már admin (plan == 3), a link helyett az "Admin" szöveg jelenik meg.