**Software Design Document**

**Content Master**

Yaroslav Marukhno

23.05.2025

UPCE

1. Struktura projektu

Struktura frontend části (úplně dole)

Frontend aplikace je vytvořen pomocí Next.js a obsahuje logiku pro zobrazení příspěvků, profilů a chatovacího rozhraní.

Struktura backend části (úplně dole)

Backend využívá Express.js a je rozdělen do samostatných modulů: API, middleware, websockety.

1. Architektura aplikace

Popis toku dat:

Aplikace je rozdělena na frontend a backend, které spolu komunikují přes HTTP API a Socket.IO.  
Frontend odesílá požadavky na REST API serveru a zároveň je připojen ke kanálu Socket.IO pro chat v reálném čase.

Frontend:

* posílá HTTP požadavky (např. pro registraci, přidání příspěvku, načtení profilu)
* komunikuje v reálném čase přes Socket.IO (soukromé zprávy)

Backend:

* přijímá požadavky přes Express API
* zpracovává autentizaci (JWT)
* ukládá data pomocí Sequelize do databáze MySQL
* spravuje Socket.IO spojení

<https://www.figma.com/board/o5l4aAmzqPX7mjV47a2BmS/Untitled?node-id=0-1&t=tenSZgIVGNs6T1N8-1>

1. Datové modely
2. User

|  |  |  |
| --- | --- | --- |
| Název pole | Typ | Popis |
| user\_id | INTEGER (PK) | Primární klíč |
| username | STRING(50) | Uživatelské jméno |
| email | STRING(255) | Emailová adresa |
| password\_hash | STRING(255) | Heslo (hash) |
| full\_name | STRING(100) | Celé jméno |
| bio | TEXT | Bio uživatele |
| profile\_picture | STRING(255) | URL k profilové fotce |
| is\_active | BOOLEAN | Aktivní účet |
| role | STRING(50) | Role uživatele |
| twoFactorEnabled | BOOLEAN | 2FA aktivní |
| twoFactorSecret | STRING(255) | 2FA klíč |
| created\_at, updated\_at | DATETIME | Timestamps |

1. Post

|  |  |  |
| --- | --- | --- |
| Název pole | Typ | Popis |
| post\_id | INTEGER (PK) | Primární klíč |
| title | STRING(100) | Titulek |
| content | TEXT | Obsah |
| image\_url | TEXT (JSON) | Obrázky jako pole |
| author\_id | INTEGER (FK) | Autor (user\_id) |
| created\_at | DATETIME | Timestamps |

1. Message

|  |  |  |
| --- | --- | --- |
| Název pole | Typ | Popis |
| message\_id | INTEGER (PK) | Primární klíč |
| from\_user\_id | INTEGER (FK) | Odesílatel |
| to\_user\_id | INTEGER (FK) | Příjemce |
| content | TEXT (nullable) | Obsah zprávy |
| media\_url | STRING (nullable) | Odkaz na médium |
| type | ENUM('text', 'image') | Typ zprávy |
| created\_at, updated\_at | DATETIME | Timestamps |

1. Community

|  |  |  |
| --- | --- | --- |
| Název pole | Typ | Popis |
| community\_id | INTEGER (PK) | Primární klíč |
| name | STRING(255) | Název komunity |
| privacy | STRING(50) | Veřejná/soukromá |
| description | TEXT | Popis |
| photo | STRING(255) | URL k fotce |
| owner\_id | INTEGER (FK) | Zakladatel |
| members\_count | INTEGER | Počet členů |
| theme | STRING(255) | Vizuální téma |
| created\_at, updated\_at | DATETIME | Timestamps |

5. Follow

|  |  |  |
| --- | --- | --- |
| Název pole | Typ | Popis |
| follow\_id | INTEGER (PK) | Primární klíč |
| follower\_id | INTEGER (FK) | Sledující |
| following\_id | INTEGER (FK) | Sledovaný |
| created\_at | DATETIME | Timestamp |

1. API-endpointy
2. auth.js

|  |  |  |
| --- | --- | --- |
| POST | /auth/register | Registrace |
| POST | /auth/login | Přihlášení |
| POST | /auth/2fa/setup | Nastavení 2FA |
| POST | /auth/2fa/verify | Ověření 2FA |
| POST | /auth/2fa/verify-login | Ověření při loginu |
| POST | /auth/2fa/disable | Vypnutí 2FA |

1. user.js

|  |  |  |
| --- | --- | --- |
| GET | /user/me | Získání aktuálního uživatele |
| GET | /user/byusername/:username | Načtení uživatele podle usernamu |

1. userCommunities.js

|  |  |  |
| --- | --- | --- |
| GET | /usercommunities | Komunity, kterých je uživatel členem |

1. community.js

|  |  |  |
| --- | --- | --- |
| POST | /createcommunity | Vytvoření nové komunity |
| GET | /mycommunities | Komunity, které uživatel vlastní |

1. chat.js

|  |  |  |
| --- | --- | --- |
| GET | /following | Sledující uživatelé a čas poslední zprávy |
| GET | /message/:userId | Zprávy s daným uživatelem |
| POST | /upload | Nahrání obrázku do chatu |

1. follow.js

|  |  |  |
| --- | --- | --- |
| POST | /follow/:userId | Začít sledovat uživatele |
| POST | /unfollow/:userId | Přestat sledovat uživatele |
| GET | /status/:userId | Zjistit, zda uživatele sleduji |

1. search.js

|  |  |  |
| --- | --- | --- |
| GET | /search?q=... | Vyhledávání uživatelů a komunit |

1. post.js

|  |  |  |
| --- | --- | --- |
| POST | /posts/ | Vytvoření nového příspěvku |
| GET | /posts/ | Získání všech příspěvků |
| DELETE | /posts/:id | Smazání vlastního příspěvku |

1. Socket.IO - funkce chatu

Připojení k Socket.IO serveru

Na klientské straně (Next.js) se při načtení chatu vytvoří připojení k serveru pomocí Socket.IO:

const socket = io('http://localhost:5000', {

auth: {

token: `Bearer ${session.accessToken}`

}

});

Tento JWT token je následně ověřen na serveru ve socketAuthMiddleware.js, kde do socket.user vložíme informace o uživateli (user\_id, email).

Autentizace Socket.IO spojení

Každý požadavek na websocket připojení prochází middlewarem:

* Ověříme JWT token pomocí jsonwebtoken.
* V případě úspěchu získáme user\_id a email → uložíme do socket.user.
* Pokud je token neplatný, připojení je odmítnuto.

Zpracování události private\_message

Když uživatel odešle zprávu, spustí se obsluha v privateMessages.js.

Postup:

1. Zpráva se uloží do databáze (tabulka messages).
2. Server vytvoří payload obsahující:
   * odesílatele,
   * příjemce,
   * text zprávy / obrázek,
   * typ (text nebo image),
   * čas odeslání.
3. Zpráva je odeslána zpět oběma stranám (odesílatel i příjemce).

Odeslání obrázku v chatu

* Uživatel nahraje obrázek (pomocí input[type="file"]).
* Obrázek je odeslán jako FormData přes POST /api/chat/upload.
* Backend ukládá obrázek do uploads/chat\_images.
* Následně se odesílá zpráva přes socket.emit('private\_message') s typem image.

Načtení historie zpráv

Při otevření chatu s uživatelem:

* Proběhne fetch na GET /api/chat/message/:userId.
* Vrátí se celá historie zpráv mezi přihlášeným uživatelem a cílovým.

Upozornění na nepřečtené zprávy

V proměnné unreadMap je sledováno, kdo poslal novou zprávu.  
Pokud je odesílatel jiný než aktuální kontakt v chatu, zobrazí se u něj bublina s tečkou.

Socket.IO události

|  |  |  |
| --- | --- | --- |
| Událost | Směr | Popis |
| private\_message | klient → server | Odeslání textu nebo obrázku jako zprávy |
| private\_message | server → klient | Příjem zprávy |
| disconnect | automatický | Uživatelské odpojení od serveru |
| connect\_error | automatický | Chyba připojení |

1. Dvoufázové ověření

Jak to funguje v aplikaci?

1. Nastavení 2FA

* Uživatel otevře svůj profil.
* Klikne na tlačítko „Edit profile“ → zobrazí se nabídka 2FA.
* Po kliknutí na „Start 2FA setup“ backend vygeneruje tajný klíč a QR kód.
* Uživatel naskenuje QR kód do autentifikační aplikace.
* Z aplikace zadá jednorázový 6místný kód → klikne na „Verify 2FA“.
* Po ověření se 2FA aktivuje (twoFactorEnabled: true).

2. Přihlášení s 2FA

* Po zadání e-mailu a hesla se uživateli zobrazí výzva k zadání 2FA kódu.
* Pokud kód ověří (/2fa/verify-login), získá JWT token jako obvykle.

3. Vypnutí 2FA

* Uživatel může kliknout na „Disable 2FA“ → backend odstraní tajný klíč i příznak aktivace.

1. Screenshoty

Изображение выглядит как текст, круг, снимок экрана, дизайн

Контент, сгенерированный ИИ, может содержать ошибки.

1 - přechod na autorizační stránku

2 - přechod na registrační stránku

Изображение выглядит как текст, снимок экрана, дизайн, Графика

Контент, сгенерированный ИИ, может содержать ошибки.

1- e-mailové pole

2 - pole hesla

3 - pole pro dvoufaktorový ověřovací kód

4 – potvrzení

Изображение выглядит как текст, снимок экрана, веб-страница, Веб-сайт

Контент, сгенерированный ИИ, может содержать ошибки.

1 - vyhledávací pole

2 – chat

3 – user

4 - přechod na stránky

5 - pole pro vytvoření postu

6 - vkládání fotek

Изображение выглядит как текст, программное обеспечение, снимок экрана, веб-страница

Контент, сгенерированный ИИ, может содержать ошибки.

1 - vytváření komunity

Изображение выглядит как текст, программное обеспечение, Мультимедийное программное обеспечение, Значок на компьютере

Контент, сгенерированный ИИ, может содержать ошибки.

1 - výběr cíle komunity

2 - zavřít modální okno

Изображение выглядит как текст, программное обеспечение, Значок на компьютере, веб-страница

Контент, сгенерированный ИИ, может содержать ошибки.

1 – sidebar seznamu chatu

2 - pole pro zprávu

3 - přidat fotografii

1. Závěr

Co lze v budoucnu zlepšit:

* Přidání notifikací v reálném čase při přijetí zprávy nebo žádosti o přátelství.
* Možnost odpovídat na příspěvky nebo přidávat komentáře.
* Rozšíření komunit o diskuze, události nebo vlastní správcovský panel.
* Vylepšení responzivity UI a podpora tmavého režimu.
* Větší kontrola nad oprávněními uživatelů (např. moderátoři komunit).

Řešené složitosti a technické výzvy

* NextAuth + 2FA:  
  Přidání dvoufázového ověření přes Google Authenticator do autentizačního systému NextAuth si vyžádalo hlubší zásah do login flow. Bylo nutné ošetřit různé scénáře: běžné přihlášení, detekce potřeby 2FA a samostatné ověření kódu přes TOTP algoritmus.
* Socket.IO:  
  Reálný časový chat pomocí Socket.IO byl poprvé implementován. Bylo nutné porozumět připojení klienta přes JWT, zajištění bezpečnosti, broadcastu zpráv a správě jednotlivých relací podle user\_id.
* Autorizace a ochrana API:  
  Backend je chráněn pomocí JWT middleware, veškeré důležité operace vyžadují ověření tokenu. Bylo třeba ošetřit různé případy neautorizovaných přístupů.

Co jsem se naučil

* Jak funguje autentizační tok v NextAuth a jak ho upravit pro specifické potřeby (2FA).
* Jak navrhnout a implementovat Socket.IO systém pro privátní zprávy.
* Jak strukturovat REST API v Node.js + Express a bezpečně přistupovat k databázi přes Sequelize.
* Jak propojit front-end s back-endem pomocí Axios, zabezpečit přenos tokenu a udržet stav klienta.

# backend/

## ├── config/

│ └── db.js

│

## ├── middlewares/

│ ├── authToken.js

│ └── uploadPostImage.js

│

## ├── models/

│ ├── Community.js

│ ├── Event.js

│ ├── Follow.js

│ ├── index.js

│ ├── Message.js

│ ├── Post.js

│ ├── User.js

│ ├── UserCommunity.js

│ ├── UserEvent.js

## │ └── node\_modules/

│

## ├── routes/

│ ├── auth.js

│ ├── chat.js

│ ├── follow.js

│ ├── posts.js

│ ├── search.js

│ ├── user.js

│ ├── userCommunities.js

│ ├── userEvents.js

│ └── community.js

│

## ├── sockets/

│ ├── handlers/

│ ├── middlewares/

│ └── index.js

│

## ├── uploads/

│ ├── chat\_images/

│ ├── communities\_images/

│ ├── events\_images/

│ ├── user\_posts/

│ └── users\_profile\_pictures/

│

├── .env

├── package-lock.json

├── package.json

└── server.js

# frontend/

## ├── app/

### │ ├── (auth)/

│ │ ├── login/page.js

│ │ └── register/page.js

### │ ├── (main)/

│ │ ├── community/page.js

│ │ ├── events/page.js

│ │ ├── explore/page.js

│ │ ├── search/page.js

│ │ ├── settings/page.js

│ │ └── layout.js

### │ ├── api/

│ │ └── auth/[...nextauth]/route.js

### │ ├── chat/page.js

### │ ├── hooks/useSocket.js

### │ ├── lib/

│ ├── profile/[username]/page.js

│ ├── layout.js

│ └── globals.css

│

## ├── components/

│ ├── auth/AuthForm.js

│ ├── chat/UserListContainer.js

### │ ├── community/

│ │ ├── CommunityCard.js

│ │ └── CommunityThemes.js

### │ ├── feeds/

│ │ ├── Feed.js

│ │ ├── FeedCommunity.js

│ │ └── FeedEvent.js

### │ ├── headers/

│ │ ├── HeaderMain.js

│ │ └── HeaderWelcome.js

### │ ├── modals/

│ │ ├── createcommunity/

│ │ ├── PostOptionsModal.js

│ │ └── userModal.js

### │ ├── search/SearchResults.js

### │ ├── sidebars/

│ │ ├── NavBarLeft.js

│ │ ├── NavBarRight.js

│ │ └── NavBar.js

│ └── AddPost.js

│

## ├── public/

### │ └── img/

│ ├── file.svg

│ ├── globe.svg

│ ├── next.svg

│ ├── vercel.svg

│ └── window.svg

│

## ├── reducers/

│ ├── authSlice.js

│ └── postsSlice.js

│

## ├── styles/

## ├── utils/

├── .env.local

├── package.json

└── README.md