

Slovenská technická univerzita v Bratislave  
Fakulty informatiky a informačných  
technológií

Vývoj progresívnych webových aplikácií  
Progresívna webová aplikácia na textovú komunikáciu v štýle IRC

Filip Bodor, Adam Strelec

Akademický rok: 2025/2026

Cvičiaci: Ing. Eduard Kuric, PhD.

Zadanie:

Vytvorte progresívnu webovú aplikáciu na textovú komunikáciu v štýle IRC (Slack), ktorá komplexne rieši nižšie definované prípady použitia.

Prípady použitia:

1. Registrácia, prihlásenie a odhlásenie používateľa
  - 1.1 Používateľ má meno a priezvisko, nickName a email
2. Používateľ vidí zoznam kanálov, v ktorých je členom
  - 2.1 Pri opustení kanála alebo trvalom vyhodení z kanála je daný kanál odobratý zo zoznamu
  - 2.2 Pri pozvánke do kanála je daný kanál zvýraznený a topovaný
  - 2.3 V zozname môže cez používateľské rozhranie kanál vytvoriť, opustiť a ak je správcom aj zrušiť
  - 2.4 Dva typy kanálov – súkromný (private channel) a verejný kanál (public channel)
  - 2.5 Správcom kanála je používateľ, ktorý kanál vytvoril
  - 2.6 Ak nie je kanál aktívny (nie je pridaná nová správa) viac ako 30 dní, kanál prestáva existovať (následne je možné použiť channelName kanála pre „nový“ kanál)
3. Používateľ odosiela správy a príkazy cez „príkazový riadok“, ktorý je fixným prvkom aplikácie. Používateľ môže odoslať správu v kanáli, ktorého je členom
4. Vytvorenie komunikačného kanála (channel) cez príkazový riadok
  - 4.1 Kanál môže vytvoriť ľubovoľný používateľ cez príkaz /join channelName [private]
  - 4.2 Do súkromného kanála môže pridávať alebo odoberať používateľov iba správca kanála cez príkazy /invite nickName a /revoke nickName
  - 4.3 Do verejného kanála sa môže pridať ľubovoľný používateľ cez príkaz /join channelName (ak kanál neexistuje, automaticky sa vytvorí)
  - 4.4 Do verejného kanála môže člen kanála pozvať iného používateľa príkazom /invite nickName
  - 4.5 Vo verejnom kanáli môže člen „vyhodiť“ iného člena príkazom /kick nickName. Ak tak spraví aspoň 3 členovia, používateľ má „trvalý“ ban pre daný kanál. Správca môže používateľa vyhodiť „natrvalo“ kedykoľvek príkazom /kick nickName, alebo naopak „obnoviť“ používateľovi prístup do kanála cez príkaz /invite
  - 4.6 nickName ako aj channelName sú unikátne

#### 4.7 Správca môže kanál zatvoriť/zrušiť príkazom /quit

5. Používateľ môže zrušiť svoje členstvo v kanáli príkazom /cancel. Ak tak spraví správca kanála, kanál zaniká
6. Správu v kanáli je možné adresovať konkrétnemu používateľovi cez príkaz @nickname

#### 6.1 Správa je zvýraznená danému používateľovi v zozname správ

7. Používateľ si môže pozrieť kompletnú históriu správ

#### 7.1 Efektívny infinite scroll

8. Používateľ je informovaný o každej novej správe prostredníctvom notifikácie

8.1 Notifikácia sa vystavuje iba ak aplikácia nie je v stave „visible“ (pozrite Quasar dokumentáciu – App Visibility)

8.2 Notifikácia obsahuje časť zo správy a odosielateľa

8.3 Používateľ si môže nastaviť, aby mu chodili notifikácie iba pre správy, ktoré sú mu adresované

9. Používateľ si môže nastaviť stav (online, DND, offline)

9.1 Stav sa zobrazuje používateľom

9.2 Ak je nastavený DND stav, neprichádzajú notifikácie

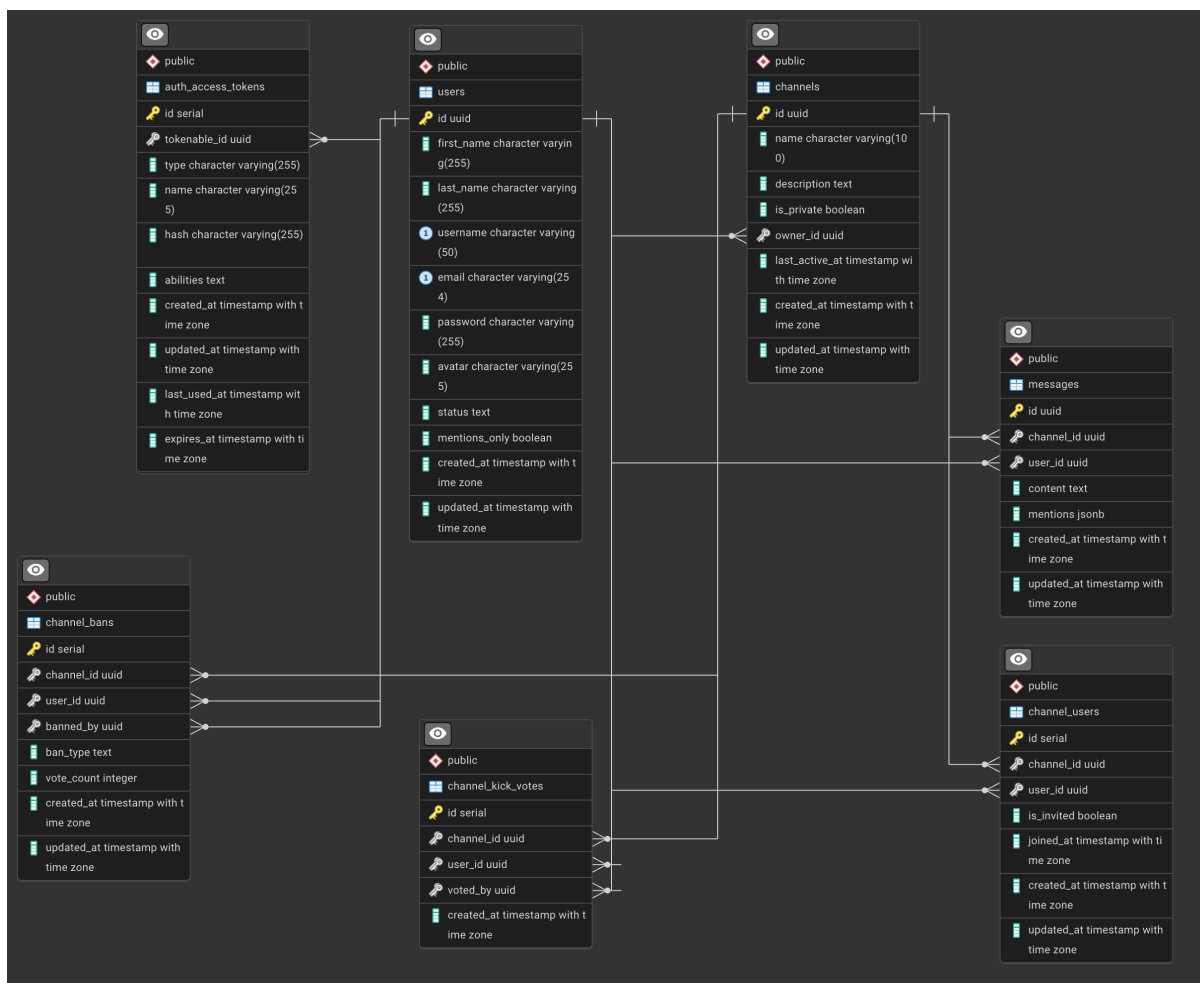
9.3 Ak je nastavený offline stav, neprichádzajú používateľovi správy, po prepnutí do online sú kanály automaticky aktualizované

10. Používateľ si môže pozrieť zoznam členov kanála (ak je tiež členom kanála) príkazom /list

11. Ak má používateľ aktívny niektorý z kanálov (nachádza sa v okne správ pre daný kanál), vidí v stavovej lište informáciu o tom, kto aktuálne píše správu (napr. „Ed is typing“)

11.1 Po kliknutí na nickname si môže pozrieť rozpísaný text v reálnom čase, predtým, ako ho odosielateľ odošle (každá zmena je viditeľná)

Fyzický model:



Zmeny:

### 1. Pridanie tabuľky channel\_bans

Uvažovali sme, že je lepšie evidovať zablokovaných používateľov v samostatnej tabuľke, namiesto toho, aby sme tieto informácie ukladali priamo v tabuľke kanálov. Tento prístup zjednodušuje správu banov, umožňuje ľahko filtrovať a spravovať zakázaných používateľov a zachováva normalizáciu dát.

### 2. Pridanie tabuľky channel\_kick\_votes

Podobne ako pri banoch, hlasovania o vylúčení používateľa z kanála je výhodnejšie uchovávať v samostatnej tabuľke. Takto vieme efektívne evidovať, kto hlasoval a proti komu, bez nutnosti komplikovaných polí v tabuľke kanálov.

### 3. Odstránenie direct\_messages

Táto tabuľka nebola podľa zadania potrebná, preto sme ju odstránili.

#### 4. **Pridanie tabuľky users**

Uchovávanie používateľov v samostatnej tabuľke je výhodné, pretože používateľ nemusí byť členom žiadneho kanála. To zároveň umožňuje jednoduchšie spravovať používateľské účty, autentifikáciu, nastavenia a históriu aktivít.

#### 5. **Odstránenie tabuľky command\_log**

Pôvodná tabuľka na logovanie príkazov sa ukázala ako nepotrebná, pretože podľa aktuálnych požiadaviek zadania nie je potrebné evidovať príkazy používateľov samostatne.

#### 6. **Zmena názvu tabuľky ChannelMembers na channel\_users**

Nový názov lepšie reflektuje obsah tabuľky – evidenciu používateľov priradených ku kanálom. Zároveň dodržiava konvenciu názvov malými písmenami a podčiarkovníkom.

#### 7. **Pridanie poľa is\_invited do tabuľky channel\_users**

Pole is\_invited slúži na zaznamenanie stavu používateľa v kanáli – či bol pozvaný alebo sa pripojil sám. Uľahčuje správu pozvánok a zobrazenie relevantného statusu v aplikácii.

#### 8. **Zmena názvu tabuľky ChannelMessage na messages**

Nový názov je jednoduchší, univerzálnejší a lepšie zapadá do zvyklostí pre pomenovanie tabuliek obsahujúcich správy.

#### 9. **Pridanie poľa channel\_id do tabuľky messages**

Pole channel\_id jasne určuje, ku ktorému kanálu správa patrí. Toto je potrebné, pretože všetky správy sa evidujú v jednej tabuľke a môžu patriť rôznym kanálom.

#### 10. **Tabuľka auth\_access\_tokens**

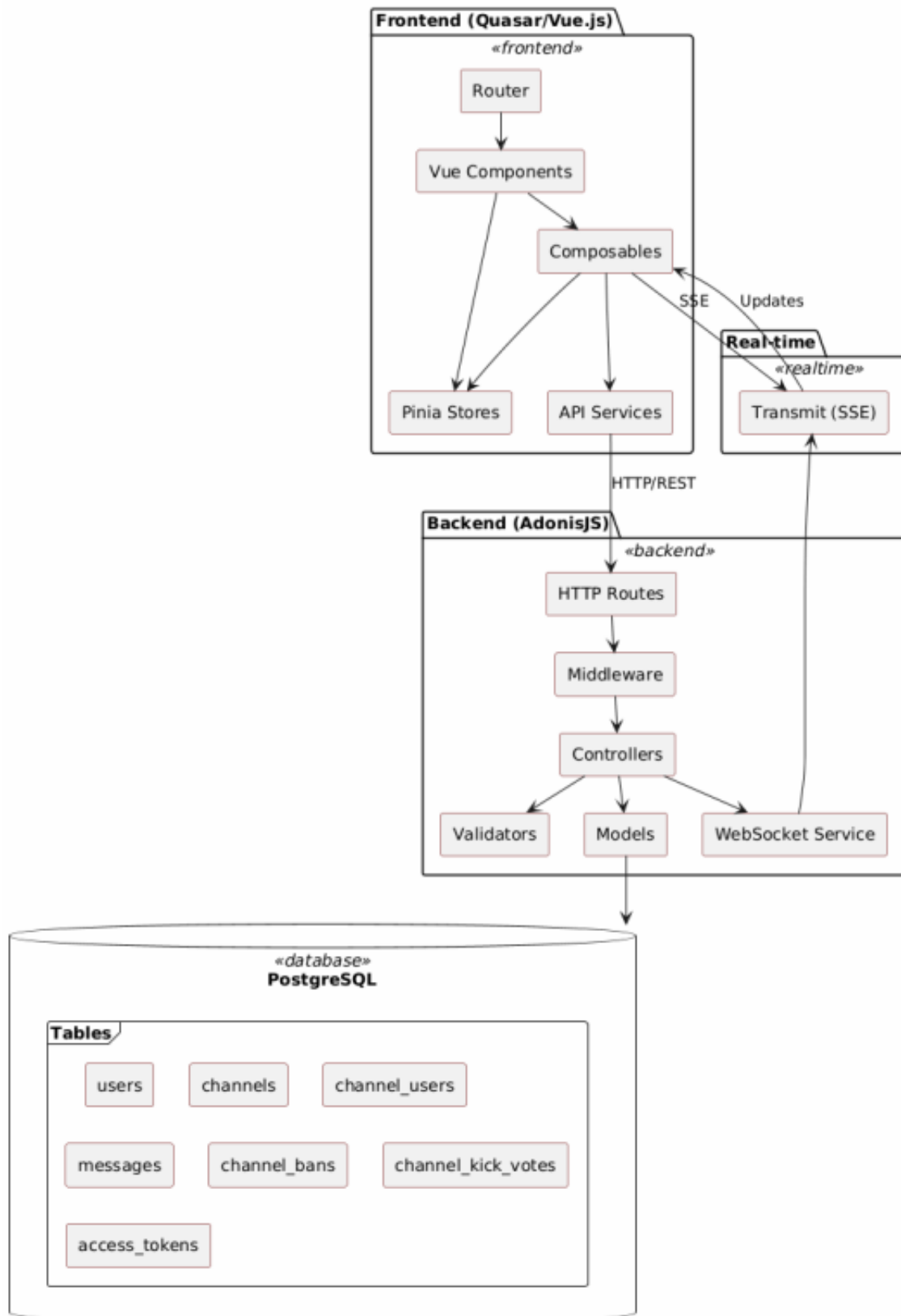
Tabuľka vytvorená automaticky AdonisJS

#### 11. **Pridanie avatara do tabuľky users**

V DB ukladáme avatarov pre rôznych userov

## Diagram architektúry

### Slack 2.0 - Application Architecture



## Frontend – technologický stack a návrhové rozhodnutia (na základe package.json)

### Framework a komponenty:

- **Vue 3 + Quasar:** Projekt využíva Vue 3 s Quasar frameworkom, ktorý poskytuje hotové komponenty (tlačidlá, formuláre, layouty) a jednoduchú podporu pre PWA. Umožňuje rýchly vývoj UI bez potreby riešiť každý komponent od nuly.

### Správa stavu:

- **Pinia:** Slúži na centralizovanú správu stavu aplikácie, napríklad pre prihláseného používateľa, zoznam kanálov alebo správy. Komponenty môžu pristupovať k týmto dátam bez zbytočného prenášania props.

### Routing a SPA navigácia:

- **Vue Router:** Rieši prechod medzi obrazovkami (login, zoznam kanálov, detail kanála, nastavenia) v rámci jednej SPA stránky bez reloadu.

### HTTP komunikácia:

- **Axios:** Používa sa na volanie REST API backendu, umožňuje jednotné spracovanie chýb a pridávanie autentifikačných tokenov do hlavičiek.

### PWA a offline podpora:

- **register-service-worker a Workbox knižnice:** Projekt je pripravený ako PWA, využíva service worker a Workbox na caching a offline prístup.

### UI doplnky:

- **@quasar/extras:** Balík fontov a ikon na rýchlu a konzistentnú integráciu UI prvkov.

### Vývojové nástroje:

- **ESLint + Prettier:** Na zachovanie jednotného štýlu kódu a odhalenie bežných chýb počas vývoja.
- **TypeScript a súvisiace nástroje:** typescript, vue-tsc, ESLint konfigurácie pre Vue a TypeScript – zvyšujú spoľahlivosť kódu a predchádzajú chybám.

## Backend – technologický stack a návrhové rozhodnutia

### Framework:

- **AdonisJS (Core + Lucid + Auth + CORS + Transmit):** Používa sa ako hlavný backend framework. Rieši routing, middleware, autentifikáciu a ORM (Lucid). CORS zabezpečuje správnu komunikáciu frontendu s API, Transmit môže riešiť real-time eventy.

### Validácia a bezpečnosť:

- **VineJS:** Kniznica na validáciu vstupov. Zabezpečuje, že backend pracuje iba s platnými dátami z formulárov alebo API requestov.
- **Argon2:** Hoci nie je explicitne v package.json, autentifikácia cez AdonisJS Auth typicky využíva bezpečné hashovanie hesiel (napr. Argon2 alebo bcrypt).

### Real-time / Eventy:

- **@adonisjs/transmit:** Umožňuje real-time komunikáciu, podobnú socket.io, na serveri – eventy môžu byť odosielané klientom okamžite, napr. pri nových správach alebo zmene členstva v kanáloch.
- **Node-cron:** Na plánovanie opakujúcich sa úloh, napr. čistenie starých dát alebo odosielanie notifikácií.

### Databáza:

- **PostgreSQL (pg) a Lucid ORM:** Rieši ukladanie používateľov, kanálov, správ a ďalších entít. Lucid poskytuje migrácie, modely a elegantnú prácu s databázou.

### Práca s časom:

- **Luxon:** Kniznica pre prácu s dátumami a časom, zabezpečuje konzistentné ukladanie a spracovanie časových údajov (napr. last\_activity\_at, timestampy správ).

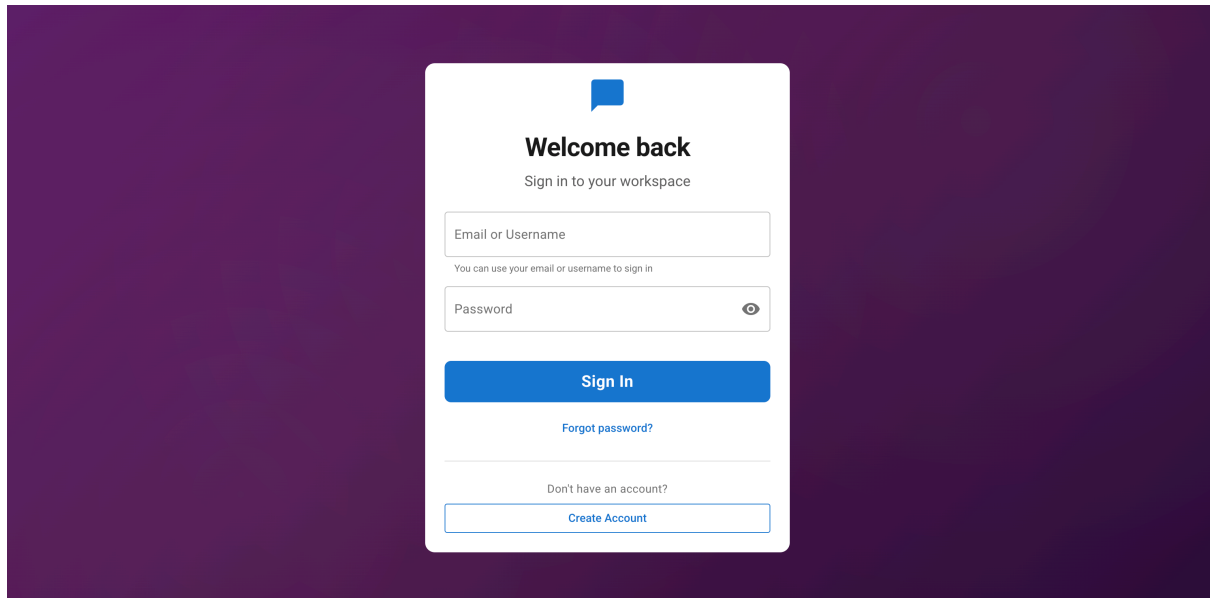
### Vývojové nástroje:

- **TypeScript:** Na statické typovanie a vyššiu spoľahlivosť kódu.
- **ESLint + Prettier:** Na jednotný štýl kódu a zachytenie chýb počas vývoja.
- **Hot-hook:** Umožňuje rýchly reload servera pri zmene kódu počas vývoja (Hot Module Reload).



**Snímky obrazoviek:**

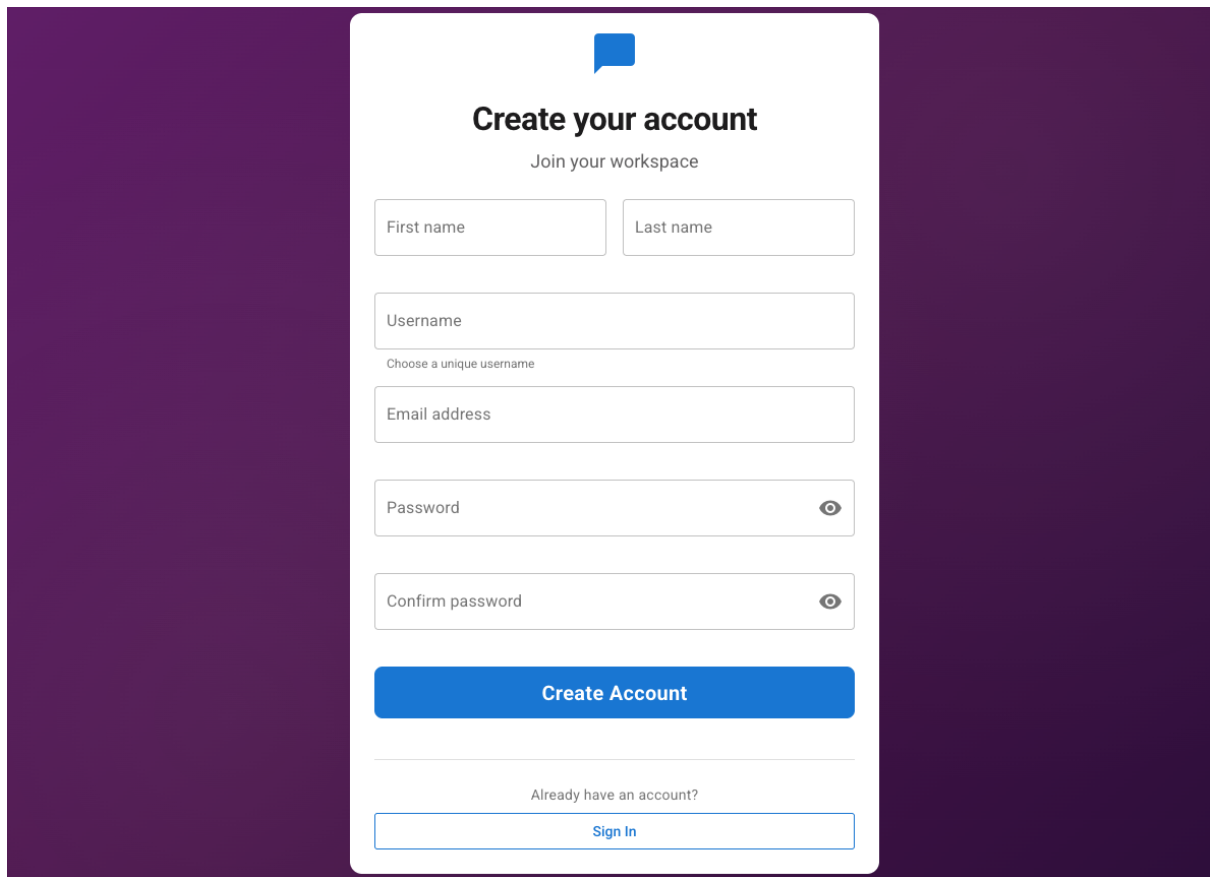
**Prihlásenie:**




The screenshot shows a login interface with a dark purple background. A white card in the center contains the following elements:

- A blue speech bubble icon at the top.
- The heading **Welcome back** in bold black text.
- The subtext "Sign in to your workspace" in a smaller black font.
- A text input field labeled "Email or Username".
- A small gray text note: "You can use your email or username to sign in".
- A password input field labeled "Password" with a toggle eye icon on the right.
- A prominent blue button labeled **Sign In**.
- A link labeled "Forgot password?" below the sign-in button.
- A horizontal separator line.
- The text "Don't have an account?" above a button.
- A white button labeled "Create Account" with a blue border.

**Registrácia:**

A registration form titled "Create your account" with a subtitle "Join your workspace". It features a blue speech bubble icon at the top. The form includes input fields for "First name", "Last name", "Username" (with a hint "Choose a unique username"), "Email address", "Password", and "Confirm password" (each with a toggle icon). A blue "Create Account" button is positioned below the password fields. At the bottom, there is a link "Sign In" preceded by the text "Already have an account?".



## Create your account

Join your workspace


First name


Last name

Username

Choose a unique username

Email address

Password 

Confirm password 

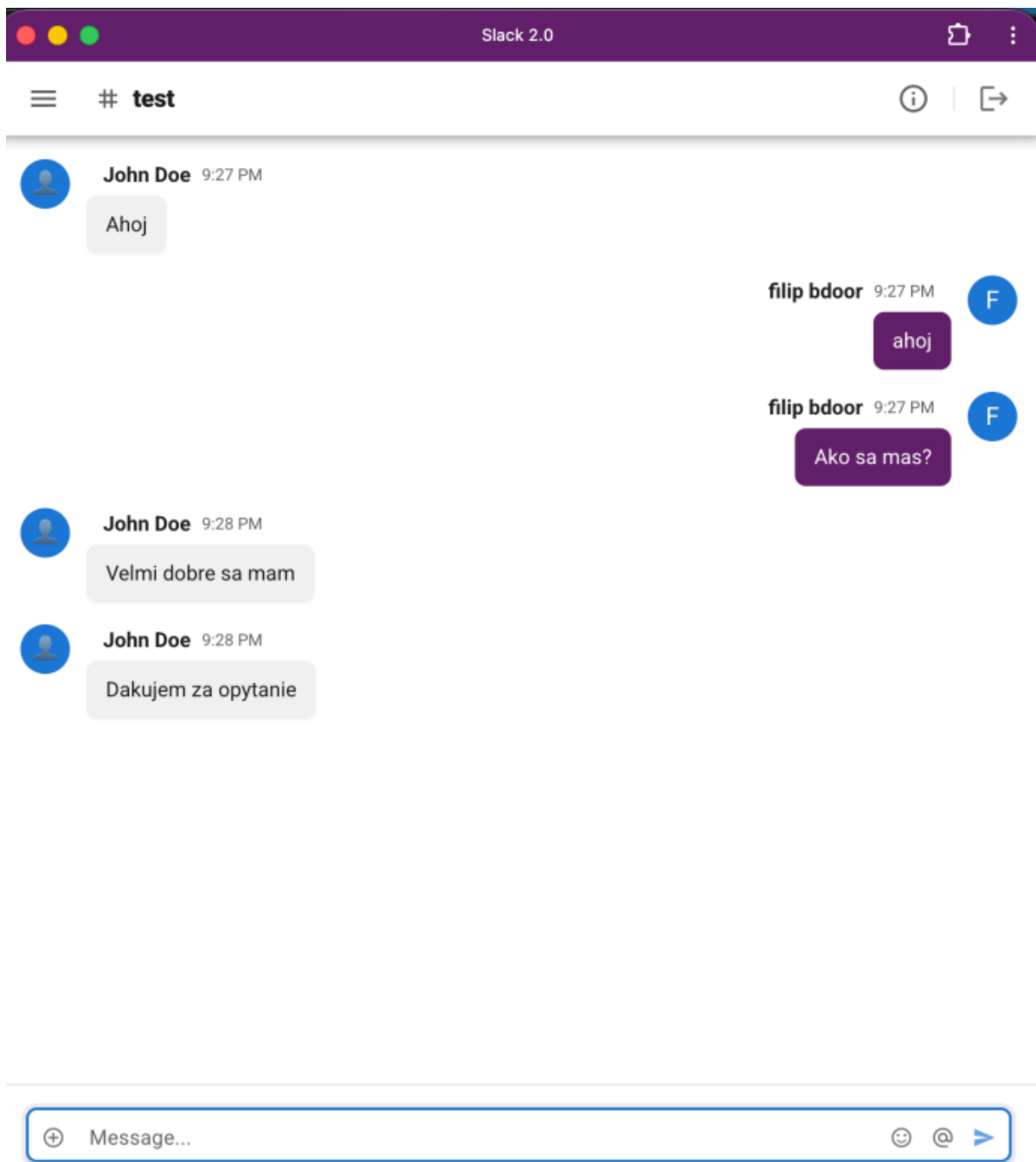
Create Account

---

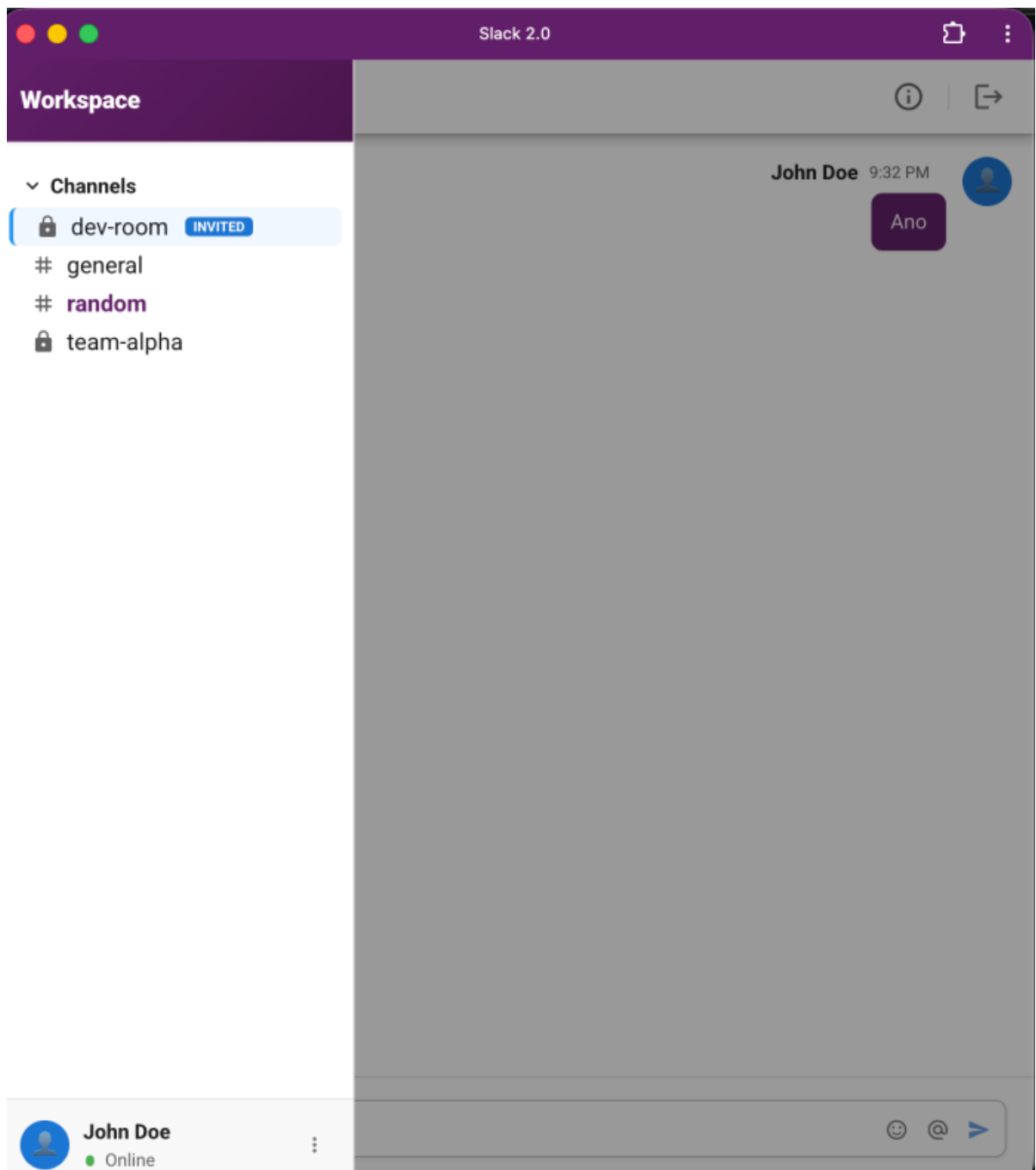
Already have an account?

[Sign In](#)

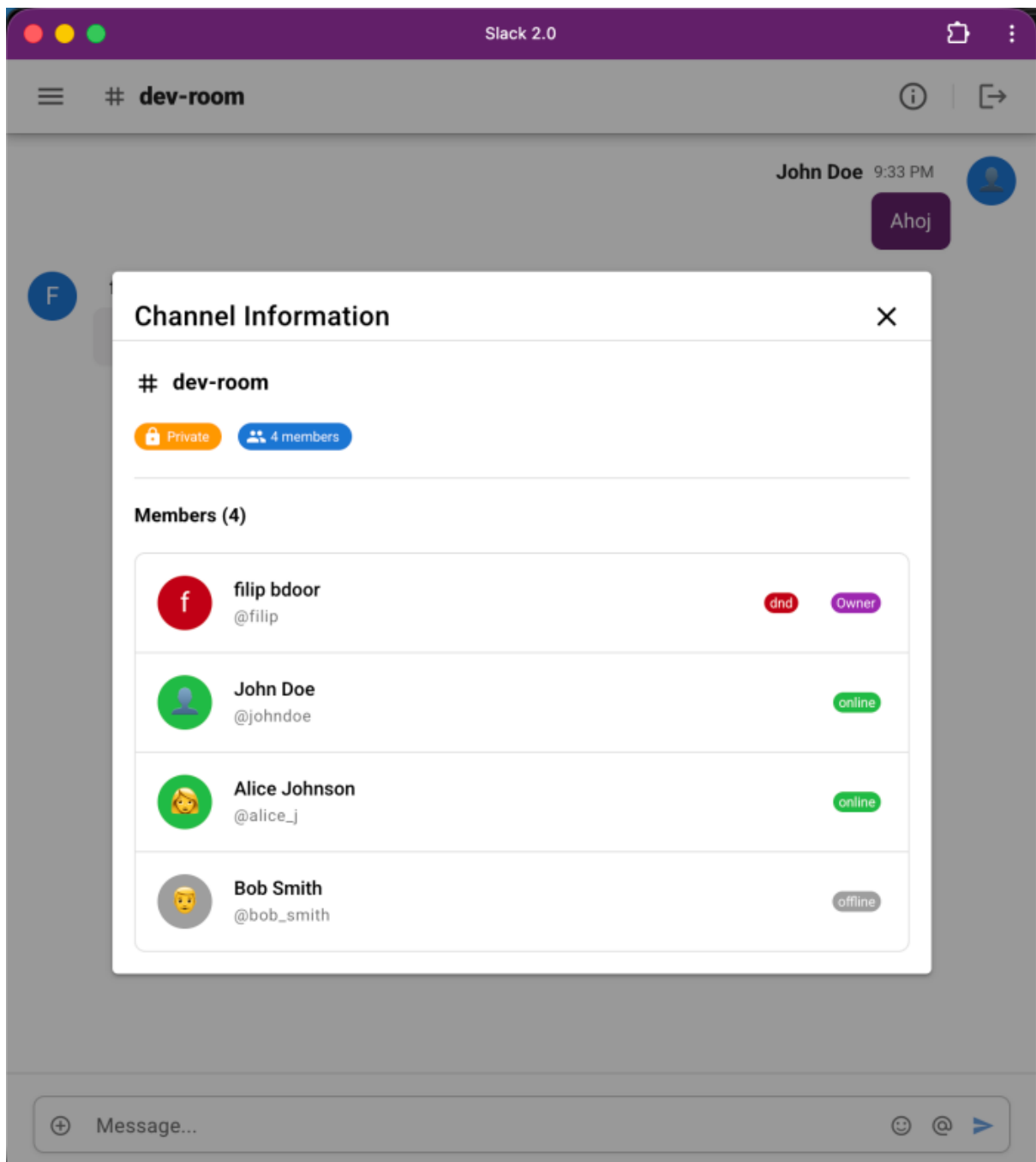
Chat (Ak nie je nahratý avatar, použijú sa iniciály z mena užívateľa):



Channels:



Channel info:



Status:

