Slash

Development of Progressive Web Applications

Created By: Artur Kozubov, Artem Zaitsev

Instructed by: [Ing. Matúš Krajčovič](https://is.stuba.sk/auth/lide/clovek.pl?id=103003)

Lectured by: [Ing. Eduard Kuric, PhD.](https://is.stuba.sk/auth/lide/clovek.pl?id=17452)

### Requirements

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

1. registrácia, prihlásenie a odhlásenie používateľa

* používateľ má meno a priezvisko, nickName a email

1. používateľ vidí zoznam kanálov, v ktorých je členom

* pri opustení kanála, alebo trvalom vyhodení z kanála je daný kanál odobratý zo zoznamu
* pri pozvánke do kanála je daný kanál zvýraznený a topovaný
* v zozname môže cez používateľské rozhranie kanál vytvoriť, opustiť, a ak je správcom aj zrušiť
* dva typy kanálov – súkromný (private channel) a verejný kanál (public channel)
* správcom kanála je používateľ, ktorý kanál vytvoril
* 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)

1. používateľ odosiela správy a príkazy cez „príkazový riadok“, ktorý je „fixným“ prvkom aplikácie. používateľ môžeodoslať správu v kanáli, ktorého je členom
2. vytvorenie komunikačného kanála (channel) cez príkazový riadok

* kanál môže vytvoriť ľubovolný používateľ cez príkaz /join channelName [private]
* do súkromného kanála môže pridávať/odoberať používateľov iba správca kanála cez príkazy /invite nickName a /revoke nickName
* do verejného kanála sa môže pridať ľubovolný používateľ cez príkaz /join channelName (ak kanál neexistuje, automaticky sa vytvorí)
* do verejného kanála môže člen kanála pozvať iného používateľa príkazom /invite nickName
* vo verejnom kanáli môže člen „vyhodiť“ iného člena príkazom /kick nickName. ak tak spravia 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
* nickName, ako aj channelName sú unikátne
* správca môže kanál zatvoriť/zrušiť príkazom /quit

1. používateľ môže zrušiť svoje členstvo v kanáli príkazom /cancel, ak tak spraví správca kanála, kanál zaniká
2. správu v kanáli je možné adresovať konkrétnemu používateľovi cez príkaz @nickname

* správa je zvýraznená danému používateľovi v zozname správ

1. používateľ si môže pozrieť kompletnú históriu správ

* efektívny inifinite scroll

1. používateľ je informovaný o každej novej správe prostredníctvom notifikácie

* notifikácia sa vystavuje iba ak aplikácia nie je v stave “visible” (pozrite quasar docu App Visibility)
* notifikácia obsahuje časť zo správy a odosielateľa
* používateľ si môže nastaviť, aby mu chodili notifikácie iba pre správy, ktoré sú mu adresované

1. používateľ si môže nastaviť stav (online, DND, offline)

* stav sa zobrazuje používateľom
* ak je nastavený DND stav, neprichádzajú notifikácie
* ak je nastavený offline stav, neprichádzajú používateľovi správy, po prepnutí do online sú kanály automaticky aktualizované

1. používateľ si môže pozrieť zoznam členov kanála (ak je tiež členom kanála) príkazom /list
2. 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)

po kliknutí na nickName si môže pozrieť rozpísaný text v reálnom čase, predtým, ako ju odosielateľ odošle (každá zmena je viditeľná) :-)

## Database structure

#### Latest migrations

* [create\_user\_notify\_when\_tagged\_table.ts](modules/backend/database/migrations/1733666411455_create_user_notify_when_taggeds_table.ts)
  + Added new status column to control if notification are only when tagged (@nickname)
* [create\_add\_image\_to\_users\_table.ts](modules/backend/database/migrations/1731479986221_create_add_image_to_users_table.ts)
  + Added image\_path column to store user profile image
* [create\_is\_resolved\_table.ts](modules/backend/database/migrations/1731474916532_create_is_resolveds_table.ts)
  + Added is\_resolved column to chat\_kicks table if the user is unbanned
* [create\_chat\_invites\_updated\_ats\_table.ts](modules/backend/database/migrations/1731448460226_create_chat_invites_updated_ats_table.ts)
  + Added updated\_at column to chat\_invites table to control when the invite was accepted
* [create\_update\_to\_have\_created\_by\_users\_table.ts](modules/backend/database/migrations/1731446387280_create_update_to_have_created_by_users_table.ts)
  + Added created\_by\_user\_id column to chat\_kicks table to control who kicked the user

### Entity-Relationship Diagram

A screenshot of a computer

Description automatically generated

### UML Class Diagram

## A screenshot of a computer Description automatically generated

## Project Architecture

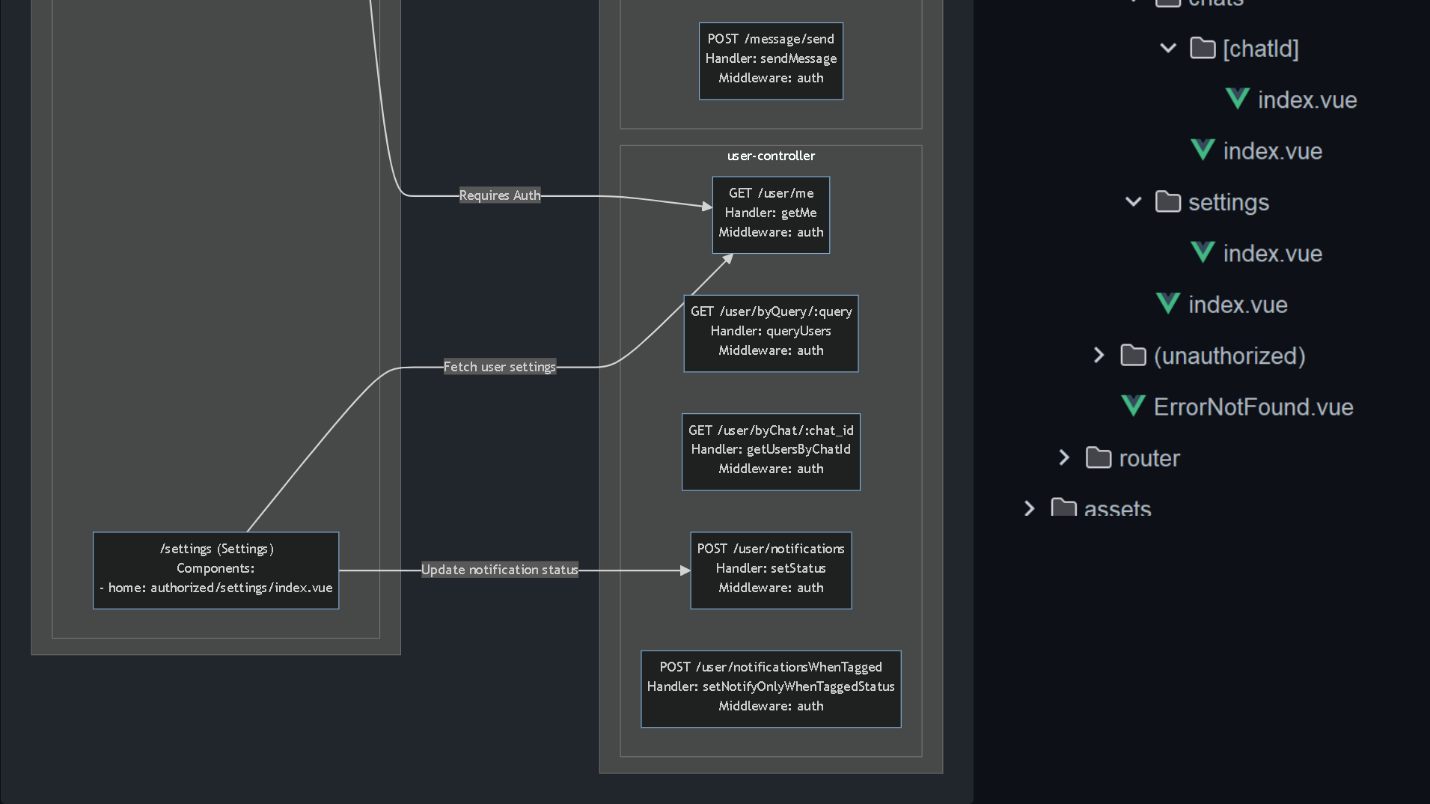
### Pages to Controller & File Structure

Next chart shows the relationship between the frontend and backend structure. The frontend is a Vue.js SPA and the backend is a REST API. Shows which page requires which endpoint to be fetched.

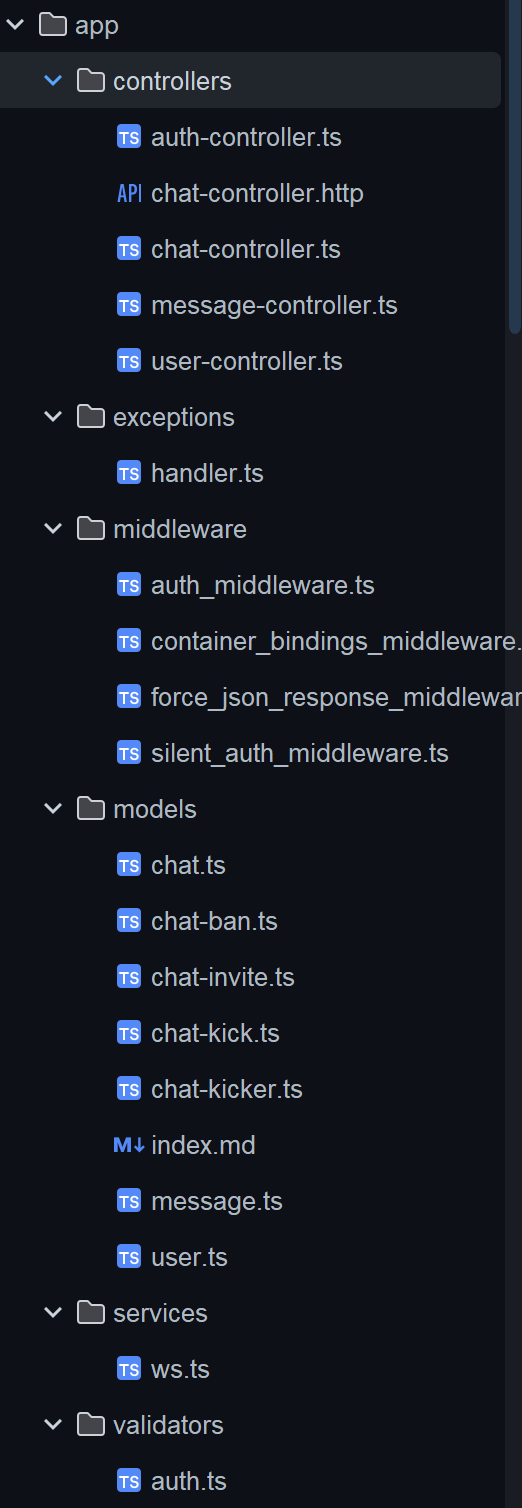
A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

**

##### frontend architectureBackend Frontend



## Project Structure

### Libraries

On the frontend side, we in additional to standard Vue.js + Quasar structure with libraries we use:

"@quasar/extras": "^1.16.4",

"axios": "^1.7.7", // Used for auth interceptors and easy-to-use HTTP requests

"core-js": "^3.31.1",

"quasar": "^2.16.0",

"vue": "^3.4.18",

"vue-router": "^4.0.12", // Used for routing in SPA. We also used a Next.js like pages structure.

"@slash/backend": "\*" // We import the backend package, as we use npm monorepo, what allows us to use the backend types and interfaces in the frontend for type safe development.

"socket.io": "^4.8.1" // Used for real-time communication (sending messages, notifications, etc.)

On the backend side, we use:

"@adonisjs/auth": "^9.2.4",

"@adonisjs/core": "^6.14.1",

"@adonisjs/cors": "^2.2.1",

"@adonisjs/lucid": "^21.3.0",

"@adonisjs/session": "^7.5.0",

"@vinejs/vine": "^2.1.0", // In register and login controllers, we use vine for validation user input

"luxon": "^3.5.0",

"node-cron": "^3.0.3", // For checking and deleting inactive chats

"pg": "^8.13.0",

"reflect-metadata": "^0.2.2",

"socket.io": "^4.8.1" // Used for real-time communication (sending messages, notifications, etc.)

## User Interface

