Integrácia a zosúladenie modulov podporného kurzového systému

ŠKOLITEL: MGR. JÁN KĽUKA, PHD.

KONZULTANT: DOC. RNDR. MARTIN HOMOLA, PHD.

RICHARD TÓTH

Podporný kurzový systém Courses

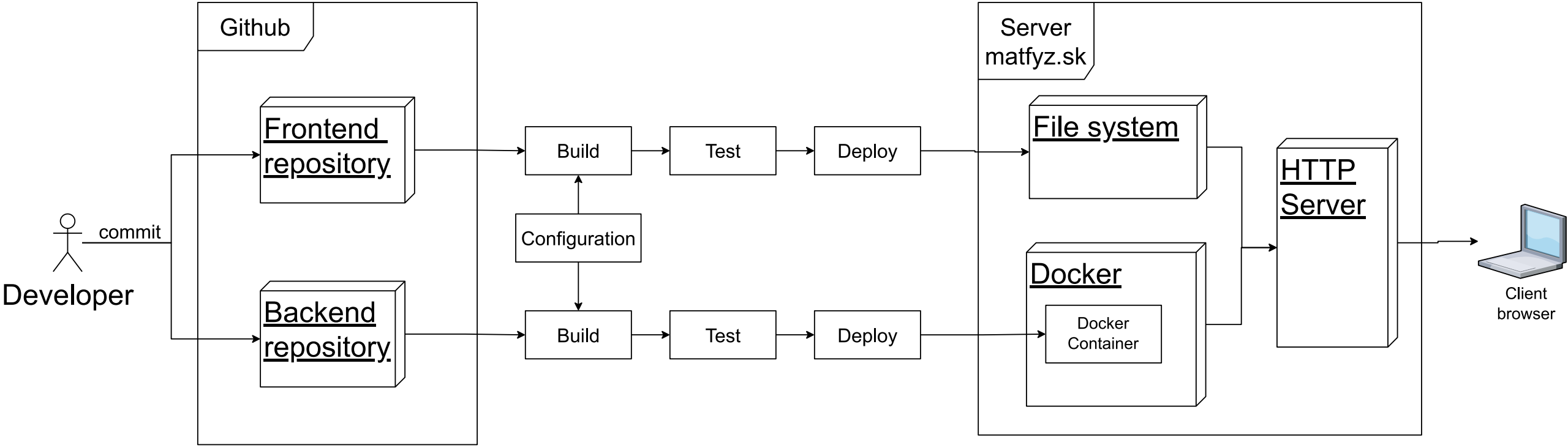
- Webová aplikácia
- Zameraná na:
  - Podporu kolaboratívneho učenia
  - Využitie technológií sémantického webu na automatické odporúčanie výučbového obsahu
- Frontend a Backend je v Javascripte

Ciele práce

- Sú rozdelené na 2 časti:
  1. Automatická integrácia zmien a nasadenie systému (CICD)
     a) Výber nástrojov
     b) Návrh a implementácia postupov vo vybranom nástroji
  2. Zjednotiť a užšie integrovať moduly systému a zadefinovať rozhrania pre nové moduly
     a) Kritické zhodnotenie kódu existujúcich modulov
     b) Zadefinovanie spoločných rozhraní na prístup k dátam
     c) Zosúladenie existujúcich modulov so zadefinovanými rozhraniami
     d) Návrh a implementácia komponentov na zdieľanie obsahu a funkcionality modulov

Hotové časti z letného semestra

- Automatická integrácia zmien a nasadenie systému
- Automatická itegrácia zmien => každá vykonaná zmena v repozitári je automaticky verifikovaná
- Verifikácia zahŕňa automatický build a testovanie aby zachytili integračnú chybu
- Automatické nasadenie systému => po úspešnej integrácii prebieha nasadenie systému na server

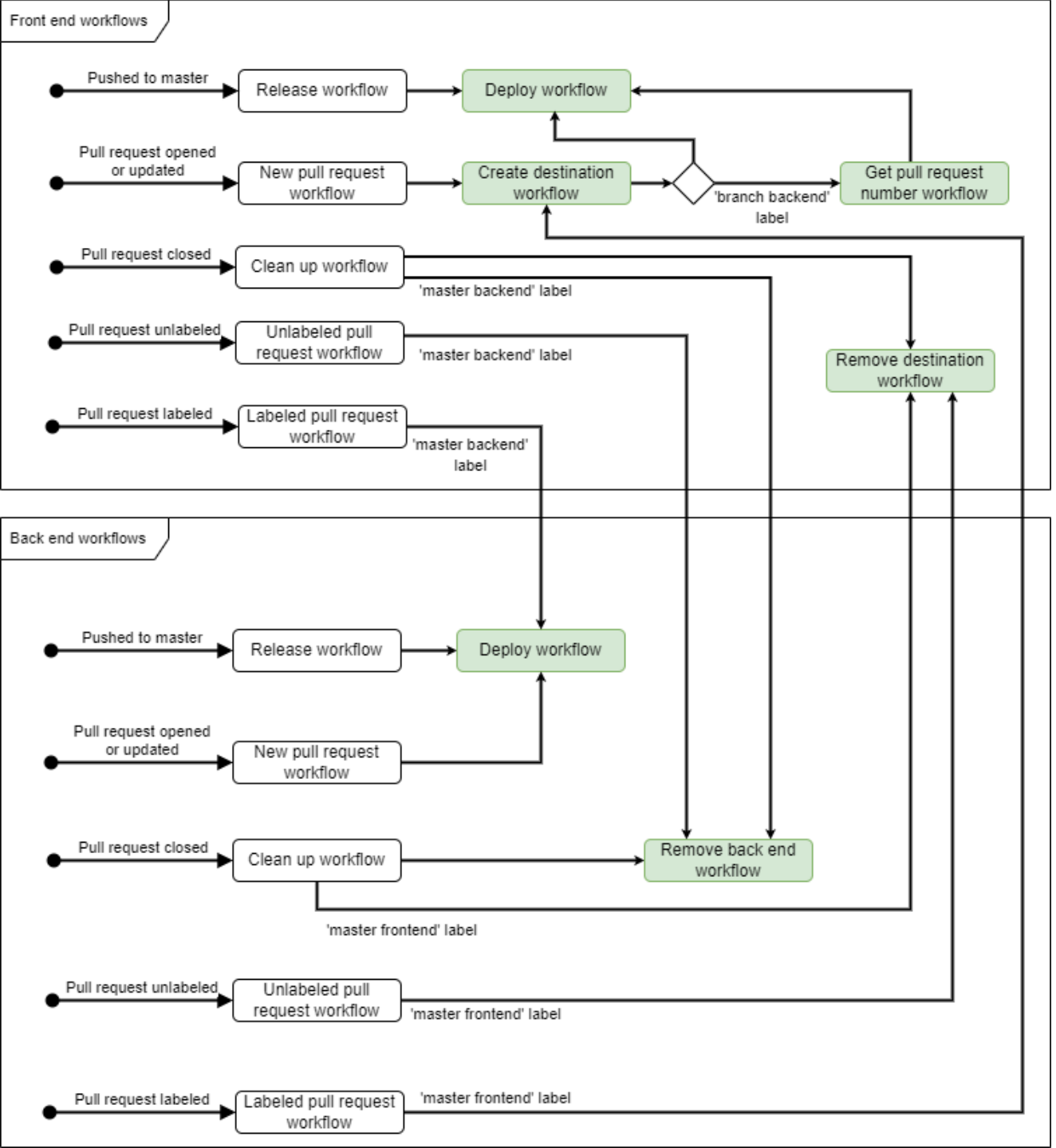

Github
Frontend repository
Backend repository
commit
Developer
Build
Test
Deploy
Configuration
Build
Test
Deploy
Server matfyz.sk
File system
Docker
Docker Container
HTTP Server
Client browser

Hotové časti z letného semestra

- Pri výberu som zvážil viaceré možnosti ako: GitHub actions, Jenkins, Concourse CI
- Naštudoval som si o každom nástroji čo ponúka
- Jenkins a Concourse CI potrebuje neustály beh na nejakom servery
- Github Actions beží na serveroch Github
- Vybral som nástroj Github Actions:
  - Netreba žiadna inštalácia
  - Veľká komunita
  - Pre verejné repozitáre je to bezplatné
  - Všetko na jednom mieste => kód + postupy nasadenia

Hotové časti z letného semestra

- „workflow“ => postupnosť krokov/akcií potrebných na vykonanie úlohy
- Navrhol a implementoval som „workflow“, ktorý integruje zmeny v produkčnom kóde a následne nasadí nový produkčný systém
- Pre testovanie nových funkcionalít v systéme som navrhol a implementoval „workflows“, ktoré umožnia nasadiť testovací systém s danou funkcionalitou
- Keďže frontend a backend majú rozdielnu integráciu a nasadzovanie, tak bolo pre oba potrebné vytvoriť vlastné „workflows“



Návrh workflowov

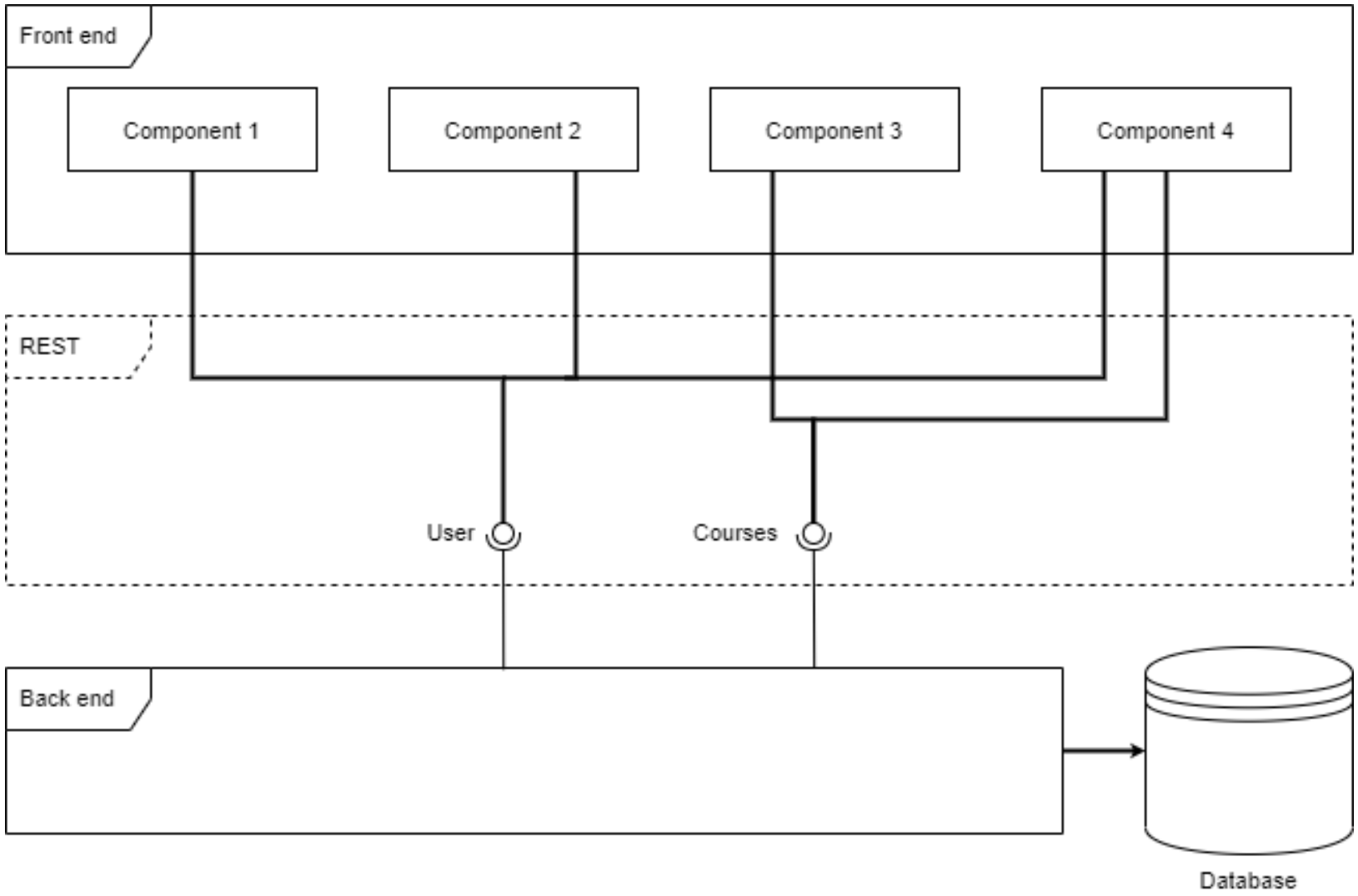
Pokrok počas semestra

- Pracoval som na druhej časti:
  - Zjednotenie a užšie integrovanie modulov systému a zadefinovanie rozhrania pre nové moduly
- Práca na front ende
- Analyzoval a zhodnotil som súčasný stav zdrojového kódu
- Navrhol som riešenie
- Implementoval som navrhnuté riešenie

Analýza a zhodnotenie súčasného stavu

- Front end pozostáva z 5 modulov: Core, Assignments, Quizzes, Results, Documents
- V každom module sa používajú rôzne spôsoby volania REST requestov na back end
- Duplicita kódu => to isté volanie implementované vo viacerých moduloch
- Neprehľadnosť => volania sú rozhádzané v komponentoch modulov
- Pracoval som na Core module a aktuálne sa dokončuje

Analýza a zhodnotenie súčasného stavu



Analýza a zhodnotenie súčasného stavu

- Príklad:
  - v module Core v komponente Profile je implementované získanie údajov o používateľovi nasledovne:
  - Na získanie údajov sa používa metódu fetch avšak sú prípady kde sa používa axios

```
fetchCurrentData() {
  fetch(`${ BACKEND_URL }data/user/${ getUserID() }
    method: 'GET',
    headers: authHeader(),
    mode: 'cors',
    credentials: 'omit',
  })
    .then(response => {
      if(!response.ok) throw new Error(response);
      else return response.json();
    })
    .then(data => {
      if(data && data['@graph']) {
        const user = data['@graph'][0]
        this.setState({user})
      }
    })
}
```

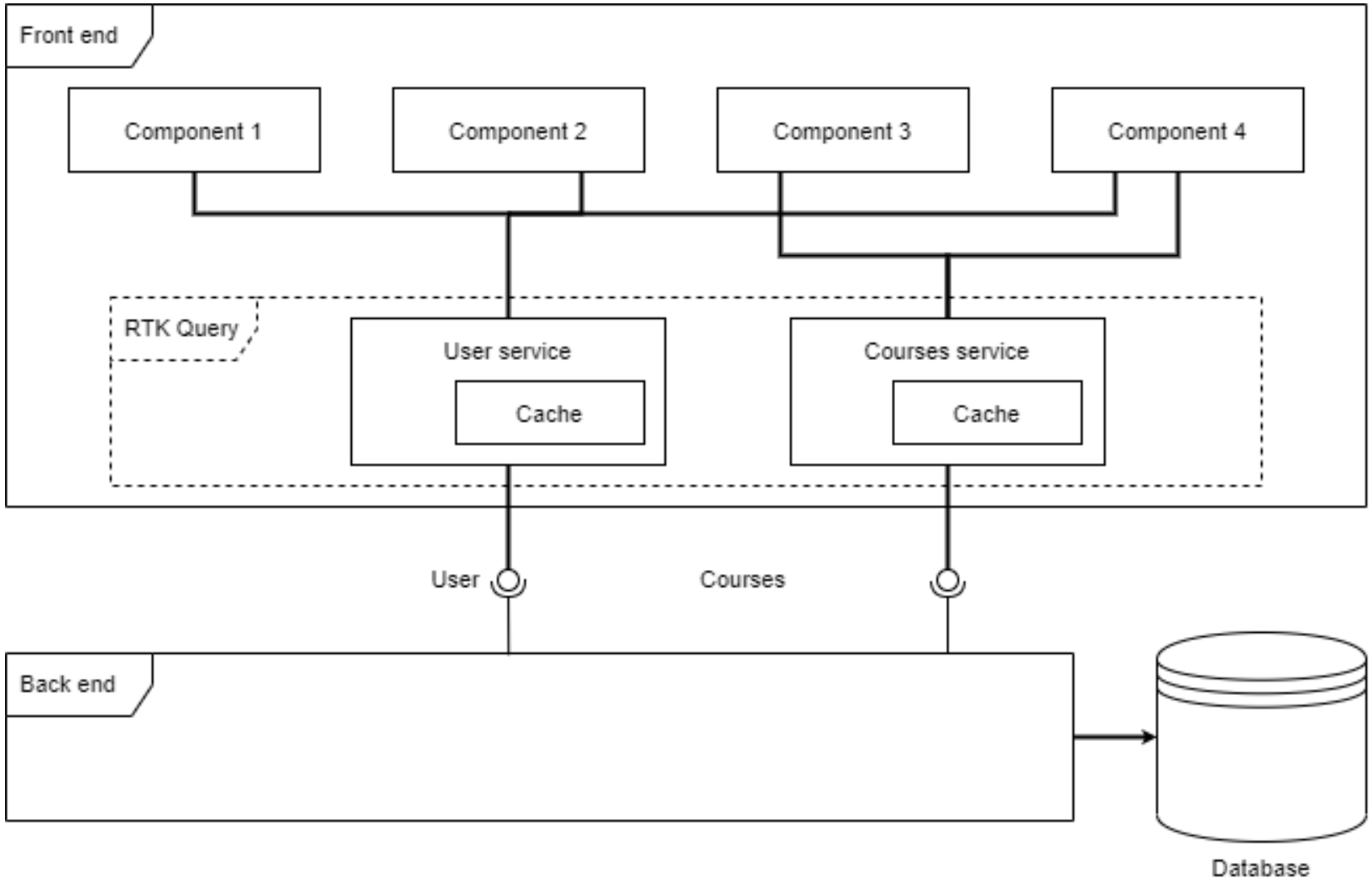
Návrh riešenia

- Čo treba spraviť:
  - Zjednotiť všetky prístupy k dátam na jedno miesto
  - Vytvorenie spoločné rozhrania pre komponenty
  - Refaktorizácia komponentov:
    - Na používanie spoločného rozhrania
    - Transformácia Class component na Functional component

- Na vytvorenie spoločného rozhrania sa použije RTK Query

- Umožňuje vytvoriť rozhranie, ktoré udržiava cache vykonaných requestov

Návrh riešenia



```
export const userApi = createApi({
    reducerPath: 'userApi',
    baseQuery: fetchBaseQuery({
        baseUrl: API_URL,
        prepareHeaders: (headers, {}) => {
            const token = getToken()
            if (token) {
                headers.set('authorization', `Bearer ${token}`)
            }
            headers.set('Content-Type', 'application/json')
            headers.set('Accept', 'application/json')
            headers.set('Cache-Control', 'no-cache')
            return headers
        },
    }),
    tagTypes: ['User'],
    endpoints: (builder) => ({
        getUser: builder.query({
            query: (id) => ({ url: `user/${id}` }),
            transformResponse: (response, meta, arg) => response["@graph"],
            providesTags: ['User'],
        }),
        getUserRequest: builder.query({
            query: (id) => ({ url: `user?requests=${id}` }),
            transformResponse: (response, meta, arg) => response["@graph"],
            providesTags: ['User'],
        }),
```

Implementácia

- Ukážka spoločného rozhrania pre prístup k dátam o používateľoch

Implementácia

- Volanie cez spoločné rozhranie z komponentu Profile

```
9   function Profile(props) {
10    const { data, isSuccess, isError, error } = useGetUserQuery(getUserID())
11    const [updateUser, result] = useUpdateUserMutation()
12    const [user, setUser] = useState(null)
13    const [errors, setErrors] = useState({})
14    const [success, setSuccess] = useState(false)
15
16    if(isSuccess && user === null) {
17      setUser(data[0])
18    } else if (isError) {
19      throw new Error(error)
20    }
21
```

Zdroje:

- Milan Cifra. Sémantický dátový model pre podporný kurzový systém. Diplomová práca, Univerzita Komenského v Bratislave, Fakulta matematiky, fyziky a informatiky, 2020.
- Paul M Duvall, Steve Matyas, and Andrew Glover. Continuous integration: improving software quality and reducing risk. Pearson Education, 2007.
- GitHub Inc. Github actions, 2022. Dostupné z https://docs.github.com/en/actions/learn-github-actions.
- Dan Abramov and the Redux documentation authors, RTK Query, 2022. Dostupné z https://redux-toolkit.js.org/rtk-query/overview