PUBLICLOOP (v0.6.0)

Vývoj skvělé, neuvěřitelné, úžasné a spektakulární aplikace pro vyhledávání spojení mezi bodem A a B ve veřejném prostoru krok za krokem.

Registrace novéhu uživatele

Nejdříve si namodelujeme proces nového uživatele v diagramové notaci BPMN (https://cs.wikipedia.org/wiki/Business_Process_Model_and_Notation). Model procesu konstruovat v nástroji Camunda (https://camunda.com/download/modeler/). Pro ukládání procesů si vytvoříme adresář {publicloop_root}/app/processes/. Spustíme si nástroj Camunda pomocí souboru "Camunda Modeler.exe" a vytvoříme si nový BPMN diagram (FIle - New File - BPMN diagram). Nový diagram si uložíme do souboru signup.bpmn (File - Save File As).

V dalším kroku si vymodeluje proces registrace nového uživatele v aplikaci PublicLoop. Dodržujete sintaxy a sémantiku notace BPMN (viz https://www.bpmn.org/).

Pro inspiraci si můžete prostududovat vzorový proces registrace (viz soubor signup.bpmn).

Implementace vymodelovaného procesu

Aktivita Odeslání formuláře

Načteme hodnoty prvků Nick, Email a Password a odešleme na adresu /signup.

```
<script>
    document.getElementById("signUp").addEventListener('click', function (event) {
        event.preventDefault();

        const floatingNick = document.getElementById("floatingNick").value;
        const floatingInput = document.getElementById("floatingInput").value;
        const floatingPassword = document.getElementById("floatingPassword").value;

        const myHeaders = new Headers();
        myHeaders.append("Accept", "application/json");
        myHeaders.append("Content-type", "application/json");

        const myBody = JSON.stringify({nick: floatingNick, email: floatingInput, password: floatingPassword});

        const myOptions = {method: "POST", headers: myHeaders, body: myBody,};

        const myRequest = new Request("http://localhost/backend/user/signup", myOptions);
        fetch(myRequest).then((response) => {
            if (!response.ok) {
                throw new Error(`HTTP error, status = ${response.status}`);
            }
            return response.text();
        }).then((text) => {
```

```
            let retObj = JSON.parse(text);
            if (retObj.login) {
                //TODO
            }
        }).catch((error) => {
            var p = document.createElement("p");
            p.appendChild(document.createTextNode(`Error: ${error.message}`));
            document.body.innerHTML = "";
            document.body.appendChild(p);
        });
    })
</script>
```

Aktivita Validace hodnot formulářových prvků (frondend)

Validace je činnost, kdy ověřujeme zda zadané hodnoty jsou v souladu s definovanými pravidly. Například zkontrolujeme, zda:

- Nick je dlouhý min dva znaky
- Email obsahuje @
    - Heslo je dlohé alespoň 6 znaků, musí obsahovat alespoň jedno malé písmeno, jedno velké písmeno, jedno číslo

```
var error = {};

let nickValid, emailValid, passwordValid;
nickValid = emailValid = passwordValid = true;

if (floatingNick.length <2)
    error.floatingNick = "Nick must be at least 2 characters";

if (!floatingInput.includes("@"))
    error.floatingInput = "Email address must contain a single @";

let pattern = /^(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{6,}$/;
if (!pattern.test(floatingPassword))
    error.floatingPassword = "Password must contain uppercase, lowercase, number and must be least 2 characters

if (Object.keys(error).length >0 ) {
    //TODO show error
} else {
    ...
}
```

Aktivita Zobrazení chyb v zadaných hodnotách

Přidáme si prvky pro zobrazování chyb do html kódu.

```
<div class="form-floating mb-3">
    <input type="text" class="form-control" id="floatingNick" placeholder="Tonda 568" required>
    <div id="floatingNickInvalidFeedback" class="invalid-feedback" style="display: none"></div>
    <label for="floatingNick">Nick</label>
```

```
</div>
<div class="form-floating mb-3">
    <input type="email" class="form-control" id="floatingInput" placeholder="name@example.com">
    <div id="floatingInputInvalidFeedback" class="invalid-feedback" style="display: none">a</div>
    <label for="floatingInput">Email address</label>
</div>
<div class="form-floating mb-3">
    <input type="password" class="form-control" id="floatingPassword" placeholder="Password">
    <div id="floatingPasswordInvalidFeedback" class="invalid-feedback" style="display: none">a</div>
    <label for="floatingPassword">Password</label>
</div>
```

Pridáme si JavaScript kód pro obsluhu zobrazení chyb.

```
document.getElementById("floatingNick").addEventListener('input', function (){
this.classList.remove("is-invalid");
document.getElementById("floatingNickInvalidFeedback").style.display = "none";
document.getElementById("floatingNickInvalidFeedback").textContent= "";
});

document.getElementById("floatingInput").addEventListener('input', function (){
    this.classList.remove("is-invalid");
    document.getElementById("floatingInputInvalidFeedback").style.display = "none";
    document.getElementById("floatingInputInvalidFeedback").textContent= "";
});

document.getElementById("floatingPassword").addEventListener('input', function (){
    this.classList.remove("is-invalid");
    document.getElementById("floatingPasswordInvalidFeedback").style.display = "none";
    document.getElementById("floatingPasswordInvalidFeedback").textContent= "";
});
```

A v případě výskytu chyby, chybu zobrazíme.

```
if (typeof error.floatingNick != 'undefined') {
    document.getElementById("floatingNick").classList.add("is-invalid");
    document.getElementById("floatingNickInvalidFeedback").style.display = "block";
    document.getElementById("floatingNickInvalidFeedback").textContent= error.floatingNick;
}
if (typeof error.floatingInput != 'undefined') {
    document.getElementById("floatingInput").classList.add("is-invalid");
    document.getElementById("floatingInputInvalidFeedback").style.display = "block";
    document.getElementById("floatingInputInvalidFeedback").textContent= error.floatingInput;
}
if (typeof error.floatingPassword != 'undefined') {
    document.getElementById("floatingPassword").classList.add("is-invalid");
    document.getElementById("floatingPasswordInvalidFeedback").style.display = "block";
    document.getElementById("floatingPasswordInvalidFeedback").textContent= error.floatingPassword;
}
```

Odeslání dotazu na adresu /signup

Viz krok "Aktivita Odeslání formuláře"

Aktivita Validace hodnot formulářových prvků (backend)

Na backendu je validace chyb stejná jako na frondendu.

Aktivita Zavedení nového uživatele do DB

1. Otestujeme, zda již není email uživatele v databázi zaveden.
2. Pokud email v databázi neexistuje, vygenerujeme unikátní kód, který bude odeslán klientovi na email k ověření a tento kód spolu s novým uživatelem zavedeme do databáze. Je dobré, aby platnost kódu byla omezena (např. 2 hodiny).

Aktivita Odeslání seznamu chyb v zadaných hodnotách

Vygeneruji response.

Aktivita Odeslání zprávy o úspěšném vytvoření účtu

Vygeneruji response.

Aktivita Odeslání emailu klientovi o úspěšném vytvoření účtu

Nejdříve se seznámíme s mailováním v nodu (viz kapitola Mailování v Nodu)

Aktivita Zobrazení zprávy o úspěšném vytvoření účtu

Zobrazím uživateli zprávu.

Vytvoření stránky activate_account.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Activate account</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="css/bootstrap.css">
    <link rel="stylesheet" href="css/styles.css">
</head>
<body>
<script src="js/bootstrap.js"></script>
<script src="js/app.js"></script>
</body>
</html>
```

Aktivita Odeslání dotazu na adresu /activate_account

Po zobrazení stránky se zobrazí loader a odešle se dotaz na server s kódem, který je v query stringu odkazu.

Aktivita Aktivování uživatelského účtu

Zjistíme, zda tento kód je zaveden v DB a čeká na aktivaci a zda je stále platný.

Aktivita Sestavení a odeslání výsledku aktivace

Vygeneruji response.

Zobrazení výsledku aktivace účtu

Na stránce /activate_account.html vypnu loader a zobrazím výsledek aktivace, který mi byl zaslán v odpovědi.

Mailování v Nodu

SMTP server Mailtrap

Testování zasílání mailů můžeme provádět pomocí služby Mailtrap, která poskytuje falešný SMTP server. Mailtrap funguje tak, že vytvoří dočasný SMTP server, který je přístupný prostřednictvím jedinečného názvu hostitele a čísla portu.

1. Ve službě Mailtrap https://mailtrap.io/ si vytvoříme účet.
2. Vybereme E-mail testing a v Setting inboxu si nastavíme "Integrations: Nodemailer" a získáme tajné přihlašovací údaje.

Odesílání emailů pomocí Nodemaileru

Nainstalujeme si Nodemailer https://www.nodemailer.com/.

```
{publicloop_root}/app/backend/npm install nodemailer
```

V backendu apliace si vytvoříme transporter Nodemaileru. V souboru {publicloop_root}/app/backend/index.js si přidáme následující zdrojový kód.

```
const nodemailer = require('nodemailer');

var transporter = nodemailer.createTransport({
host: "sandbox.smtp.mailtrap.io",
port: 2525,
auth: {
    user: "ecf0e8a2d33207",
    pass: "**********20c5"
}
});
```

Dále si do souboru {publicloop_root}/app/backend/index.js přidáme funkci pro odeslání mailu sendMail.

```
async function sendMail(from, to, subject, text, html) {
    return await transporter.sendMail({
      from: from,
      to: to,
      subject: subject,
      text: text,
      html: html
  });
}
```

Nyní si přidáme routu /mail/send pro otestování funkčnosti.

```
case '/mail/send':
    var from = '"Lukáš Čegan" <lukas.cegan@upce.cz>';
    var to = "hello@world.com, ahoj@svete.cz";
    var subject = "Hello, Ahoj";
    var text = "Hello world. Ahoj světe.";
    var html = "<p>Hello world. Ahoj světe.</p>";

    sendMail(from, to, subject, text, html).then((messageId) => {
        var resData = {messageId: messageId};
        res.statusCode = 200;
        res.setHeader('Content-Type', 'application/json');
        res.setHeader('Set-Cookie', session_cookie);
        res.end(JSON.stringify(resData));
    }).catch(console.error)
break;
```

Zadáme adresu adresu http://localhost:3000/mail/send a následně zkontrolujeme v mailboxu mailtrap.io, že jsme email obdrželi.
Na výstupu bychom měli obdržet info zprávy.

Vytváření šablon pomocí Mailgen

Mailgen https://github.com/eladnava/mailgen je prostředek pro vytváření HTML šablon pro transakční e-maily. Mailgen si nainstalujemen.

```
{publicloop_root}/app/backend/npm install mailgen
```

Mailgen si naimportujeme a nakonfigurujeme.

```
var Mailgen = require('mailgen');
// Configure mailgen by setting a theme and your product info
var mailGenerator = new Mailgen({
    theme: 'default',
    product: {
        // Appears in header & footer of e-mails
        name: 'Mailgen',
        link: 'https://mailgen.js/'
        // Optional product logo
```

```
        // logo: 'https://mailgen.js/img/logo.png'
    }
});
```

A dále si vygenerujeme email.

```
var email = {
  body: {
      name: 'Lukáš Čegan',
      intro: 'Welcome to Publicloop! We\'re very excited to have you on board.',
      action: {
          instructions: 'To get started with Publicloop, please click here:',
          button: {
              color: '#22BC66',
              text: 'Confirm your account',
              link: 'https://mailgen.js/confirm?s=d9729feb74992cc3482b350163a1a010'
          }
      },
      outro: 'Need help, or have questions? Just reply to this email, we\'d love to help.'
  }


};
```

```
var emailBody = mailGenerator.generate(email);
var emailText = mailGenerator.generatePlaintext(email);

// Můžeme si vygenerovat náhled emailu a uložit na disk
const preview_file = path.resolve(__dirname, '..', 'tmp_emails/preview.html');
require('fs').writeFileSync( preview_file, emailBody, 'utf8');
```

Email nyní můžeme odeslat např. pomocí Nodemailer (viz výše).

```
var from = '"Publicloop" <info@publicloop.com>';
var to = "new.user@domain.com";
var subject = "Confirm your account!";

var mailGenerator = new Mailgen({
    theme: 'default',
    product: {
        name: 'Publicloop',
        link: 'https://publicloop.com/'
    }
});

var email = {
    body: {
        name: 'Lukáš Čegan',
        intro: 'Welcome to Publicloop! We\'re very excited to have you on board.',
        action: {
            instructions: 'To get started with Publicloop, please click here:',
            button: {
                color: '#22a0bc',
                text: 'Confirm your account',
```

```
                link: 'https://publicloop.com/confirm?s=jhfad65d4f35asd4f3as4d35f4asd3f'
            }
        },
        outro: 'Need help, or have questions? Just reply to this email, we\'d love to help.'
    }
};
var emailBody = mailGenerator.generate(email);
var emailText = mailGenerator.generatePlaintext(email);

// Můžeme si vygenerovat náhled emailu a uložit na disk
const preview_file = path.resolve(__dirname, '..', 'tmp_emails/preview.html');
require('fs').writeFileSync( preview_file, emailBody, 'utf8');

sendMail(from, to, subject, emailText, emailBody).then((messageId) => {
    var resData = {messageId: messageId};
    res.statusCode = 200;
    res.setHeader('Content-Type', 'application/json');
    res.setHeader('Set-Cookie', session_cookie);
    res.end(JSON.stringify(resData));
}).catch(console.error)
```

DotEnv

```
{publicloop_root}/app/backend/npm install dotenv
```

V adresáři backend vytvoříme soubor.env

```
{publicloop_root}/app/backend/.env
```

Do toho souboru vložíme přihlašovací údaje.

```
USER=<mailtrap user key>
PASSWORD=<mailtrap password>
```

A nyní můžeme upravit soubor {publicloop_root}/app/backend/index.js a načítat přihlašovací údaje z proměnných.

```
var dotenv = require('dotenv');
dotenv.config();

const transporter = nodemailer.createTransport({
host: 'smtp.mailtrap.io',
port: 2525,
auth: {
    user: process.env.USER,
    pass: process.env.PASSWORD,
}
});
```

> Soubor .env je dobré zařadit do souboru .gitignore, aby tento soubor byl verzovacím systémem ignorován.