PUBLICLOOP (v0.2.0)

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

Připojení do databáze z WebStormu

Spustíme WebStorm a otevřeme okno Database (View -> Tool Windows -> Databas). Následně otevřeme dialog pro vytvoření spojení s databázovým serverem (New -> DataSource -> MariaDb -> MariaDB). V dialogu vyplníme:

- name: @localhost
- host: localhost
- port: 3306
- username: root

> V případě, že nemáme nainstalovaný řadič, zobrazí se nám ve spodní části dialogu "Download missing driver files". Kliknutím na slovo Download se nám automaticky řadič nainstaluje.

Následně můžeme otestovat spojení a v případě úspěchu můžeme spojení uložit. Po uložení spojení se nám automaticky otevře okno "console", prostřednictvím kterého můžeme s databází komunikovat.

> Okno "console" můžeme otevřít pomocí výběru konkrétního spojení a tlačítka New -> Query Console (Ctrl + Shift + Q)

Založení nového uživatele a databáze

Pro vytvoření nové databáze publicloop napíšeme do okna console níže uvedený SQL dotaz a provedeme Execute (tlačítko zelené šipky).

```
CREATE DATABASE publicloop CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_520_ci;
```

Pro vytvoření nového uživatel publicloop s veškerými právy pro databázi publicloop napíšeme do okna console níže uvedený SQL dotaz a provedeme Execute.

```
CREATE USER publicloop IDENTIFIED BY 'publicloopppassword';
GRANT ALL PRIVILEGES ON publicloop.* TO 'publicloop'@'%' WITH GRANT OPTION;
```

Připojení do databáze publicloop

V oknu Database (View -> Tool Windows -> Databas) otevřeme dialog pro vytvoření spojení s databázovým serverem (New -> DataSource -> MariaDb -> MariaDB). V dialogu vyplníme:

- name: publicloop@localhost
- host: localhost
- port: 3306
- username: publicloop
- password: publicloopppassword
- database: publicloop

Tvorba datového modelu v nástroji MySQL Workbench

Vytvoříme si nový model (File -> New model).Následně si model uložíme File -> Save model as do adresáře {publicloop_root}/datamodel s názvem publicloop.mwb. V modelu si vytvoříme novou tabulku "Users", která bude mít atributy:

- user_key (int)
- username (varchar)
- password (varchar)

Z modelu vygenerujeme SQL dotaz.

```
CREATE TABLE IF NOT EXISTS `publicloop`.`users` (
  `user_key` INT NOT NULL AUTO_INCREMENT,
  `email` VARCHAR(255) NOT NULL,
  `password` VARCHAR(255) NOT NULL,
  PRIMARY KEY (`user_key`))
ENGINE = InnoDB;
```

Tento dotaz vložíme do konzole publicloop@localhost a provedeme Execute. V okně Database by se měla ve výpisu zobrazit tabulka users pod spojením publicloop@localhost.

> V případě, že použijete spojení @localhost, musíte nejdříve vybrat databázi, se kterou chcete pracovat:
>
> ```
> use publicloop;
> ```

CRUD

Vyzkoušejte si všechny operace CRUD:

- Cread

```
INSERT INTO `users` VALUES (null, "lukas.cegan@upce.cz", "heslo");
```

- Read

```
SELECT * FROM `users` WHERE `email` = "lukas.cegan@upce.cz";
```

- Update

```
UPDATE `users` SET  `email` ="cegan@upce.cz" WHERE `email` = "lukas.cegan@upce.cz";
```

- Delete

```
DELETE FROM `users` WHERE `email` = "lukas.cegan@upce.cz";
```

Komunikace s DB v backendu

Do backenedu aplikace přidáme balíček pro práci s databází mysql.

```
npm install mysql2
```

Do zdrojového kódu souboru {publicloop_root}/app/backend/index.js vložte podporu pro práci s DB.

```
const http = require('http');
const mysql = require("mysql2");
const hostname = '127.0.0.1';
const port = 3000;
const server = http.createServer((req, res) => {
    var body = '';  //get request body
    req.on('data', chunk => {
      body += chunk.toString(); // convert Buffer to string
    });
    req.on('end', () => {
      let reqData = JSON.parse(body);
      let username = reqData.email;
      let password = reqData.password;
      var connection = getConn();
      connection.query(
        'SELECT *  FROM users WHERE email=? and password = ?',
        [username, password],
        function (err, results) {
          if (err) {
            res.statusCode = 500;
            res.setHeader('Content-Type', 'text/plain');
            res.end('Internal Server Error');
          } else {
            var exist = false;
```

```
            results.forEach(function(item){
              exist = true;
            });
            var resData = {login: exist}
            res.statusCode = 200;
            res.setHeader('Content-Type', 'application/json');
            res.end(JSON.stringify(resData));
          }
        }
      );
    });
});
server.listen(port, hostname, () => {
  console.log(`PublicLoop running at http://${hostname}:${port}/`);
});

function getConn () {
  return mysql.createConnection({
    host: 'localhost',
    user: 'publicloop',
    password: 'publicloop‌password',
    database: 'publicloop',
  });
}
```

Do zdrojového kódu souboru {publicloop_root}/app/frondend/signin.html vložíme následující kód.

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

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

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

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

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

        const myRequest = new Request("http://localhost/backend", 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) {
                   window.location.replace("http://localhost/map.html");
               }

            })
            .catch((error) => {
                var p = document.createElement("p");
                p.appendChild(document.createTextNode(`Error: ${error.message}`));
                document.body.innerHTML = "";
                document.body.appendChild(p);
            });
    })
</script>
```

Směrování klientských požadavků na backend aplikace

Pro každý klientský požadavek je na backendu vytvořena aplikační logika, která tento požadavek zpracuje. Směrování klientských požadavků na konkrétní kus zdrojového kódu obstará "Router". V našem případě bude router představovat větvení programu pomocí konstrukce "switch".
Zdrojový kód souboru {publicloop_root}/app/backend/index.js upravíme dle níže uvedeného příkladu.

```
const http = require('http');
const mysql = require("mysql2");
const hostname = '127.0.0.1';
const port = 3000;
const server = http.createServer((req, res) => {
  switch (req.url) {
    case '/user/signin':
    var body = '';  //get request body
    req.on('data', chunk => {
      body += chunk.toString(); // convert Buffer to string
    });
    req.on('end', () => {
      let reqData = JSON.parse(body);
      let username = reqData.email;
      let password = reqData.password;
      var connection = getConn();
      connection.query(
        'SELECT *  FROM users WHERE email=? and password = ?',
        [username, password],
        function (err, results) {
          if (err) {
            res.statusCode = 500;
            res.setHeader('Content-Type', 'text/plain');
            res.end('Internal Server Error');
```

```
          } else {
            var exist = false;
            results.forEach(function(item){
              exist = true;
            });
            var resData = {login: exist}
            res.statusCode = 200;
            res.setHeader('Content-Type', 'application/json');
            res.end(JSON.stringify(resData));
          }
        }
      );
    });
    break;
  case '/':
  default:
    res.statusCode = 404;
    res.setHeader('Content-Type', 'text/plain');
    res.end('Not Found');
    break;
  }
});
server.listen(port, hostname, () => {
  console.log(`PublicLoop running at http://${hostname}:${port}/`);
});

function getConn () {
  return mysql.createConnection({
    host: 'localhost',
    user: 'publicloop',
    password: 'publicloopppassword',
    database: 'publicloop',
  });
}
```

Ve zdrojovém kódu souboru {publicloop_root}/app/frondend/signin.js upravíme adresu endpointu na http://localhost/backend/user/signin.

```
...
const myRequest = new Request("http://localhost/backend/user/signin", myOptions);
...
```

Routování a práce s DB ve frameworku Express

Alternativně můžeme výše popsané implementovat pomocí frameworku Express. Zdrojový kód souboru {publicloop_root}/app/backend/index.js by vypadal následovně.

```
const express = require('express');
const mysql = require("mysql2");
const hostname = '127.0.0.1';
```

```
const port = 3000;

const app = express();
app.use(express.json());
app.post('/user/signin', (req, res) => {

    let username = req.body.email;
    let password = req.body.password;
    var connection = getConn();
    connection.query(
        'SELECT *  FROM users WHERE email=? and password = ?',
        [username, password],
        function (err, results) {
            if (err) {
                res.statusCode = 500;
                res.setHeader('Content-Type', 'text/plain');
                res.send('Internal Server Error');
            } else {
                var exist = false;
                results.forEach(function (item) {
                    exist = true;
                });
                var resData = {login: exist}
                res.statusCode = 200;
                res.setHeader('Content-Type', 'application/json');
                res.send(JSON.stringify(resData));
            }
        }
    )
})
app.get("*", (req, res) => {
  res.statusCode = 404;
  res.setHeader('Content-Type', 'text/plain');
  res.send("Not Found");
});
app.listen(3000, () => console.log(`PublicLoop running at http://${hostname}:${port}/`))

function getConn() {
  return mysql.createConnection({
    host: 'localhost',
    user: 'publicloop',
    password: 'publicloopppassword',
    database: 'publicloop',
  });
}
```

Do zdrojového kódu souboru {publicloop_root}/app/frondend/signin.js přidáme hlavičku Content-type

```
myHeaders.append("Content-type", "application/json");
```