DELTA – Střední škola informatiky a ekonomie, s.r.o.
Ke Kamenci 151, Pardubice

Maturitní projekt
Webová stránka pro fotbalový klub SK Dolany

Příjmení, jméno: Daniel Beneš
Třída: 4.A
Studijní obor: Informační technologie 18-20-M/01
Školní rok: 2023/2024
Konzultant: doc. Mgr. Josef Horálek, Ph.D.

Prohlášení

Prohlašuji, že jsem maturitní projekt vypracoval(a) samostatně, výhradně s použitím uvedené literatury.

V Pardubicích 31.3.2024

(vlastnoruční podpis)

Zadání maturitního projektu z informatických předmětů

Jméno a příjmení:
Pro školní rok: *2023/2024*
Třída: *4. A*
Obor: *Informační technologie 18-20-M/01*

Téma práce: Tvorba webu skdolany.cz
Vedoucí práce: Josef Horálek

Způsob zpracování, cíle práce, pokyny k obsahu a rozsahu práce:

Cílem práce je tvorba nového komplexního webu pro SKDolany (skdolany.cz). Nový web musí obsahovat všechny stávající funkcionality a obdobný design. Musí být implementováno administrátorské rozhraní pro přidávání událostí do kalendáře, fotek do galerie atd. (samostatný web, u kterého je designová stránka nedůležitá), dále je preferován backend postavený na PHP (avšak to není podmínkou), databáze buď MySQL nebo PostgeSQL, nebp je možné využít i nosql řešení.

Stručný časový harmonogram (s daty a konkretizovanými úkoly):

Analýza aktuálních funkcionalit webu
Návrh technologií a platforem
Návrh Use Case diagramu, Doménového diagramu a Class diagramu
Návrh designu webu
Realizace backend a frontend webu
Testování

Poděkování

Největší poděkování bych chtěl věnovat doc. Mgr. Josefu Horálkovi Ph.D. za odborné vedení projektu, už od první chvíle při výběru technologií až po konzultování správného formátu dokumentace a celkového odborného vedení. Velké díky také patří Martinu Zemanovi za komunikaci s fotbalovým klubem a poskytnutí všech potřebných materiálů.

Anotace

Vytvořená webová aplikace pro fotbalový klub SK Dolany slouží jako efektivní prostředek pro informování veřejnosti o činnosti klubu. Klubová sekce poskytuje prostor pro interní komunikaci a organizaci zápasů, což zlepšuje efektivitu správy klubu. Díky aplikaci se zvyšuje propojenost mezi členy klubu a zlepšuje se celková koordinace aktivit.

Klíčová slova

Techonologie, Diskuze, Admin, Backend, Frontend, Webová aplikace

Annotation

The web application created for the football club SK Dolany serves as an effective means of informing the public about the club's activities. The club section provides a space for internal communication and organization of matches, which improves the efficiency of club administration. Thanks to the application, the connectivity between club members is increased and the overall coordination of activities is improved.

Keywords

Techonology, Discussion, Admin, Backend, Frontend, Web App

Obsah

1 Úvod *7*

1.1 Cílová skupina **7**

2 Technologie *8*

2.1 React **8**

2.2 Typescript **8**

2.3 Tailwind CSS **8**

2.4 Node JS **9**

2.5 Express.js **9**

2.6 PostreSQL **9**

3 Architektura *10*

3.1 Use case diagram **11**

3.2 Backend **12**

3.2.1 Připojení na databázy 12

3.2.2 Autentizace a autorizace uživatelů 12

3.2.3 Diskuze a příspěvky v diskuzích 15

3.2.4 Novinky 18

3.2.5 Resetování hesla 19

3.3 Frontend **21**

3.3.1 Tailwind CSS 23

4 Webová aplikace *24*

4.1 Navigační menu **24**

4.2 Patička **24**

4.3 Úvodní stránka **25**

4.4 Vytvoření diskuze **27**

4.5 Profil **29**

4.6 Přihlášení **30**

4.7 Registrace **31**

4.8 Změna hesla **31**

5 Admin *34*

6 Budoucnost aplikace *36*

6.1 Co v aplikaci chybí? **36**

6.2 Co bude dál **36**

7 Závěr *36*

8 Literatura *37*

9 Seznam obrázků *38*

1 Úvod

V dnešní digitální době se fotbalové kluby často potýkají s obtížemi v organizaci a komunikaci s jejich členy a veřejností. S tímto vědomím byla vytvořena webová aplikace pro fotbalový klub SK Dolany, která má za cíl efektivněji informovat veřejnost o činnosti klubu a zlepšit interní komunikaci a správu klubových aktivit.

Webová aplikace SK Dolany poskytuje prostor pro prezentaci klubových událostí, výsledků zápasů, fotografií a dalších relevantních informací pro veřejnost. Tímto způsobem je klub schopen lépe angažovat fanoušky a získávat nové členy.

Klubová sekce aplikace umožňuje interní komunikaci mezi členy klubu, což zlepšuje efektivitu organizačních procesů, jako jsou domlouvání zápasů, tréninků a řešení organizačních záležitostí. Díky aplikaci se zvyšuje propojenost mezi členy klubu a zlepšuje se celková koordinace aktivit.

Na závěr je důležité zdůraznit, že tento projekt není pouze teoretickým konceptem, ale reálným projektem pro fotbalový klub SK Dolany. Jeho cílem je od základu předělat stávající webovou stránku, která neodpovídá požadavkům klubu.

1.1 Cílová skupina

Cílovou skupinou webové aplikace pro fotbalový klub SK Dolany jsou především fotbaloví nadšenci a fanoušci klubu, kteří se aktivně zajímají o dění v klubu, výsledky zápasů a události spojené s klubovým životem. Tito uživatelé budou využívat aplikaci k pravidelnému sledování novinek, fotografií a videí ze zápasů a tréninků.

Další důležitou skupinou jsou členové fotbalového klubu SK Dolany, včetně hráčů, trenérů a vedení klubu, kteří potřebují efektivní prostředek pro interní komunikaci a organizaci klubových aktivit. Pro ně bude klubová sekce aplikace neocenitelným nástrojem pro domlouvání zápasů, tréninků, sdílení dokumentů a řešení organizačních záležitostí.

2 Technologie

Pro vývoj frontendu byl zvolen React spolu s Typescriptem, zatímco pro backend byla zvolena platforma Node.js. Jako databázový systém je aplikace napojena na PostgreSQL, což zajišťuje spolehlivé a efektivní ukládání dat.

2.1 React

React je framework pro tvorbu webových aplikací postavený na knihovně JavaScriptu. Jeho síla spočívá v modularitě a komponentovém přístupu, který umožňuje snadnou a efektivní tvorbu uživatelských rozhraní.

Díky Reactu je možné vytvářet interaktivní a dynamické webové stránky, které rychle reagují na uživatelské interakce. Tento framework je široce využíván ve vývoji moderních webových aplikací a poskytuje bohatou sadu nástrojů pro práci s uživatelským rozhraním.
React také nabízí možnost využití server-side renderingu (SSR), který umožňuje generování obsahu na straně serveru a jeho rychlé zobrazení uživatelům. To vede k lepšímu uživatelskému zážitku a vylepšuje SEO webových stránek. (1)

2.2 Typescript

Typescript je nadstavba jazyka JavaScript, která přidává statické typování, což umožňuje definovat typy proměnných, parametrů funkcí a návratových hodnot. Tímto způsobem pomáhá odhalit chyby v kódu již v době vývoje a zlepšuje jeho čitelnost a udržitelnost.

Díky typescriptu lze také využívat moderní koncepty jazyka JavaScript a nové funkce, které jsou podporovány až ve vyšších verzích JavaScriptu, což umožňuje psát kód efektivněji a bezpečněji. To vede k menšímu množství chyb a nekonzistencí v kódu a usnadňuje práci v týmu, protože ostatní členové mohou lépe porozumět struktuře a typům dat v projektu. (2)

2.3 Tailwind CSS

Tailwind CSS je moderní CSS framework, který umožňuje rychlý vývoj webových aplikací pomocí předdefinovaných utility tříd přímo v HTML kódu. Díky modularitě a flexibilitě je snadné vytvářet responzivní layouty a komponenty bez potřeby psaní vlastního CSS.

Jeho přednosti spočívají v jednoduchosti použití a možnosti rychlého prototypování designu. Tailwind CSS tak poskytuje efektivní nástroj pro vytváření moderních a responzivních webových stránek s minimálním úsilím a čistým kódem. (3)

2.4 Node JS

Node.js je open-source prostředí, které umožňuje běh JavaScriptových aplikací mimo webový prohlížeč. Jeho výkonný JavaScriptový runtime, založený na enginu V8 od společnosti Google. Jedním z hlavních rysů Node.js je možnost asynchronního programování, které umožňuje aplikacím provádět více úloh současně bez zablokování hlavního vlákna.

Díky svým vlastnostem je Node.js často využíván pro vývoj serverových aplikací, API, mikroslužeb a dalších projektů, které vyžadují rychlé a efektivní zpracování dat. S rozsáhlou a aktivní komunitou vývojářů se Node.js stává populárním nástrojem pro moderní webové aplikace a backendové systémy. (4)

2.5 Express.js

Express.js je open-source webový framework pro Node.js, který umožňuje jednoduché a efektivní vytváření webových aplikací a API. Jeho hlavním cílem je usnadnit proces vytváření serverové logiky a řízení tras v aplikacích postavených na Node.js. (5)

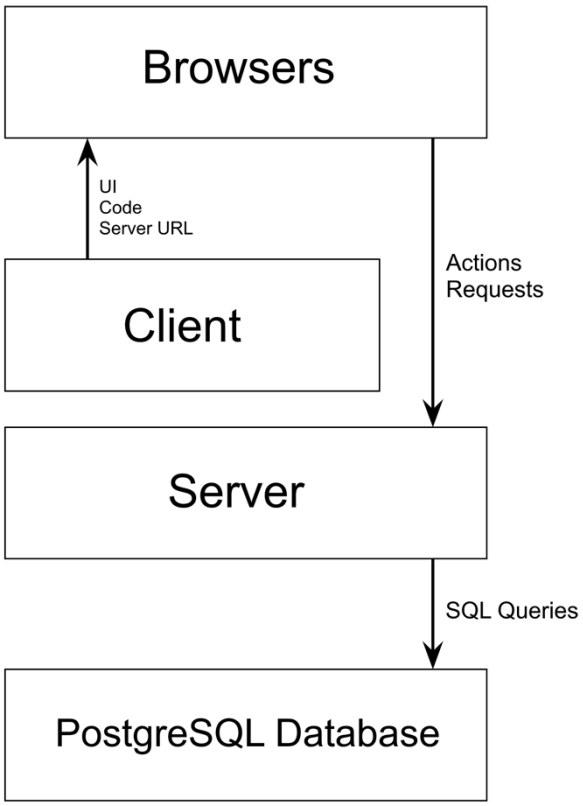
2.6 PostreSQL

PostgreSQL je open-source relační databázový systém s vysokým výkonem a rozšířenými možnostmi. Jeho flexibilita a široká škála funkcí umožňují efektivní správu a manipulaci s daty v různých typech aplikací, od malých projektů po velké podnikové systémy.

Díky své spolehlivosti, robustnosti a bezpečnosti je PostgreSQL často používán pro ukládání citlivých dat, jako jsou například účetní záznamy, osobní informace nebo transakční data. Nabízí pokročilé funkce jako jsou tranzakce, dotazování pomocí jazyka SQL, podpora JSON datového typu, indexování, a mnoho dalších, což umožňuje efektivní správu a analýzu dat v různých aplikacích a prostředích. (6)

3 Architektura

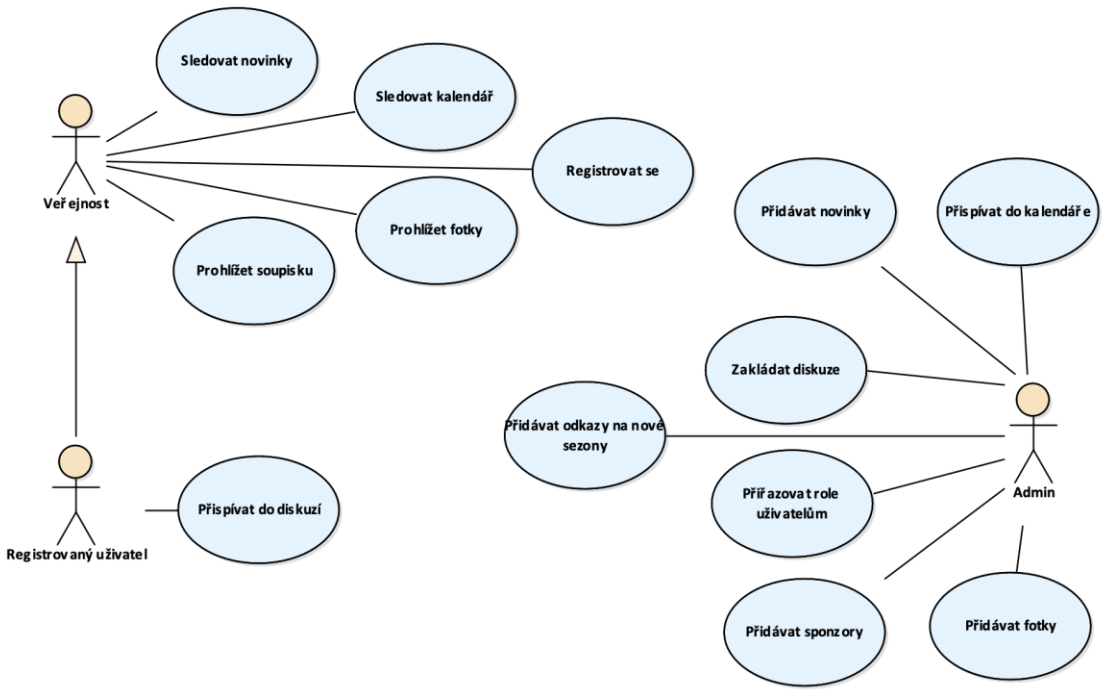
Pern stack je zkratka označující kombinaci čtyř hlavních technologií používaných při vývoji webových aplikací. Patří sem PostgreSQL, který slouží jako relační databáze, Express.js, což je framework pro vývoj serverové části aplikace v jazyce JavaScript, React pro tvorbu interaktivního a responzivního uživatelského rozhraní, a Node.js, který poskytuje běhové prostředí pro spouštění serverového kódu. Tato kombinace technologií umožňuje vytvoření moderních webových stránek. (7)



Obrázek 1 Architektura projektu [zdroj: (7)]

3.1 Use case diagram

Obdržel jsem use case diagram od zákazníka, který mi pomohl lépe porozumět fungování aplikace. S touto vizualizací a konzultacemi se zákazníkem jsem se mohl efektivně řídit celým vývojem.



Obrázek 2 use case diagram [zdroj: Martin Zeman]

3.2 Backend

Backend aplikace SK Dolany využívá kombinaci Node.js a Express.js pro poskytnutí aplikačního rozhraní a zpracování požadavků klienta. S pomocí JWT tokenů je zajištěna autentizace a ověření uživatelů. Tato část aplikace je plně oddělená od frontendu, který je vyvinut pomocí Reactu a Typescriptu. Díky této architektuře je zajištěna účinná správa uživatelských dat a maximalizace rychlosti a efektivity komunikace mezi klientem a serverem.

3.2.1 Připojení na databázi

Konfiguraci připojení k databázi PostgreSQL pomocí modulu **pg** a vytváří pool pro přístup k databázi prostřednictvím specifikovaných údajů, jako jsou uživatelské jméno, heslo, host, port a název databáze, které jsou získány z proměnných.

```
const Pool = require("pg").Pool;

const pool = new Pool({
  user: process.env.DB_USER,
  password: process.env.DB_PASS,
  host: process.env.DB_HOST,
  port: process.env.DB_PORT,
  database: process.env.DB_NAME,
});
```

Obrázek 3 databázové připojení [zdroj: autor]

3.2.2 Autentizace a autorizace uživatelů

Modul **auth.js** obsahuje různé cesty pro správu uživatelů v aplikaci. První cesta **/register** umožňuje uživatelům provést registraci. Po přijetí požadavku se zkontrolují poskytnuté údaje a nový uživatel se uloží do databáze. Pokud registrace proběhne úspěšně, vrátí se informace o nově vytvořeném uživateli.

```
router.post("/register", async (req, res) => {
  try {
    const { username, password, name, surname, email, role } = req.body;
    const userRole = role || "public";
    const user = await createUser(
      username,
      password,
      name,
      surname,
      email,
      userRole
    );

    res.json(user.rows[0]);
  } catch (err) {
    res
      .status(500)
      .json({ message: "Registration failed", error: err.message });
  }
```

Obrázek 4 /register endpoint [zdroj: autor]

Druhá cesta **/login** slouží k ověření uživatele při přihlašování. Uživatel poskytne své přihlašovací údaje a požadavek se zkontroluje pomocí lokální strategie ověření. Pokud jsou poskytnuté údaje platné, uživatel je úspěšně autentizován a vygeneruje se JWT token, který se vrátí uživateli společně s jeho informacemi.

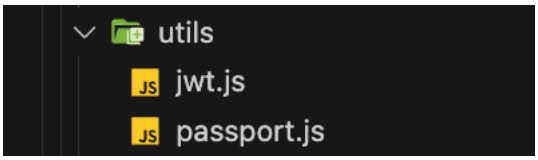
```
router.post("/login", (req, res, next) => {
  passport.authenticate("local", { session: false }, (err, user) => {
    if (err) {
      return res.status(500).json({ message: "Internal Server Error" });
    }
    if (!user) {
      return res.status(401).json({ message: "Authentication failed" });
    }

    req.login(user, { session: false }, (err) => {
      if (err) {
        res.send(err);
      }

      const token = jwt.sign({ id: user.id }, secretKey);
      res.json({ token, user });
    });
  })(req, res, next);
});
```

Obrázek 5 /login endpoint [zdroj: autor]

Ve složce utils se nachází dva soubory, jeden je jwt.js a druhý passport.js



Obrázek 6 adresář utils [zdroj: autor]

jwt.js zajišťuje použití JWT (JSON Web Token) pro ověření identity uživatelů. Využívá knihovnu Passport.js a Passport-JWT pro implementaci ověřování JWT tokenů. Při autentizaci se provádí ověření tokenu získaného z hlavičky autorizace HTTP požadavku. Poté se pomocí identifikátoru uživatele v JWT payloadu zjišťuje uživatelský záznam v databázi. (8)

```
passport.use(
  new JWTStrategy(
    {
      jwtFromRequest: ExtractJWT.fromAuthHeaderAsBearerToken(),
      secretOrKey: process.env.JWT_SECRET,
    },
    async (jwtPayload, done) => {
      try {
        const user = await findById(jwtPayload.id);
        if (!user) return done(null, false);

        return done(null, user);
      } catch (err) {
        return done(err);
      }
    }
```

Obrázek 7 jwt.js [zdroj: autor]

passport.js implementuje strategii ověření uživatelů pomocí jména a hesla. Používá knihovnu Passport.js a strategii Passport-Local. Při autentizaci se porovnává zadané heslo s heslem uloženým v databázi pro uživatele se zadaným uživatelským jménem. (9)

```
passport.use(
  new LocalStrategy(async (username, password, done) => {
    try {
      const user = await findByUsername(username);
      if (!user) return done(null, false);

      const passwordMatch = await bcrypt.compare(password, user.password);
      if (!passwordMatch) return done(null, false);

      return done(null, user);
    } catch (err) {
      return done(err);
    }
  })
);
```

Obrázek 8 passport.js [zdroj: autor]

Model **User.js** obsahuje funkce pro manipulaci s databázovou tabulkou uživatelů. Obsahuje metody pro vytváření nového uživatele, vyhledávání uživatele podle uživatelského jména, ID nebo e-mailu. Tyto funkce umožňují práci s uživatelskými údaji v databázi, jako je vytváření nových účtů, vyhledávání existujících uživatelů a aktualizace jejich informací. Pro zabezpečení hesel se využívá hashování pomocí knihovny bcrypt. (10)

```
async function createUser(username, password, name, surname, email, role) {
  const hashedPassword = await bcrypt.hash(password, 10);
  return pool.query(
    "INSERT INTO users(username, password, name, surname, email, role) VALUES($1, $2, $3, $4, $5, $6) RETURNING id",
    [username, hashedPassword, name, surname, email, role]
  );
}
```

Obrázek 10 User.js [zdroj: autor]

```
    const user = await pool.query("SELECT * FROM users WHERE username = $1", [
      username,
    ]);

    if (user.rows.length === 0) {
      return null;
    }

    return user.rows[0];
  } catch (error) {
    throw error;
  }
}
```

Obrázek 9 User.js [zdroj: autor]

Soubor **protected.js** obsahuje konfiguraci pro zabezpečený endpoint **/profile**, který vyžaduje ověření pomocí JWT. To umožňuje získání profilových informací o přihlášeném uživateli. Při požadavku je uživatel ověřen pomocí JWT strategie, která je implementována v souboru jwt.js v adresáři utils. Pokud je uživatel úspěšně ověřen, je načten jeho profil z databáze a vrácen klientovi ve formátu JSON. Pokud uživatel není nalezen nebo dojde k chybě, server odpoví s odpovídajícím HTTP kódem a zprávou o chybě.

```
router.get(
  "/profile",
  passport.authenticate("jwt", { session: false }),
  async (req, res) => {
    try {
      const user = await findByUsername(req.user.username);

      if (!user) {
        return res.status(404).json({ message: "User not found" });
      }

      const userProfile = {
        username: user.username,
        name: user.name,
        surname: user.surname,
        email: user.email,
        role: user.role,
      };

      res.json({ message: "This is a protected route", user: userProfile });
    } catch (err) {
      console.error(err.message);
      res.status(500).json({ message: "Internal server error" });
    }
  }
);
```

Obrázek 11 protected.js [zdroj: autor]

3.2.3 Diskuze a příspěvky v diskuzích

Diskuze

Soubor **discussions.js** obsahuje implementaci funkcionalit diskuzí. Tento soubor definuje různé body API pro manipulaci s diskuzemi.

Při vytváření nové diskuze jsou poskytnuty informace o tématu a roli diskuse. Před přidáním do databáze je ověřeno, zda je uživatel autorizován pomocí JWT tokenu.

```
router.post("/discussions", async (req, res) => {
  try {
    const { topic, role } = req.body;
    const author_id = jwt.verify(
      req.headers["x-auth-token"],
      process.env.JWT_SECRET
    ).id;
    const newDiscussion = await pool.query(
      "INSERT INTO discussions (author_id, topic, role) VALUES($1, $2, $3 ) RETURNING *",
      [author_id, topic, role]
    );
    res.json(newDiscussion.rows[0]);
  } catch (err) {
    console.error(err.message);
    res.status(500).send("Server Error");
  }
});
```

Obrázek 12 /discussions endpoint [zdroj: autor]

Pro získání informací o existujících diskuzích slouží koncový bod GET /discussions. Tento bod vrací seznam všech dostupných diskuzí v aplikaci. Pro získání konkrétní diskuze na základě zadaného identifikátoru je k dispozici koncový bod GET /discussions/:id.

```
router.get("/discussions", async (req, res) => {
  try {
    const allDiscussions = await pool.query(
      "SELECT discussions.*, users.username FROM discussions JOIN users ON discussions.author_id = users.id"
    );
    res.json(allDiscussions.rows);
  } catch (err) {
    console.error(err.message);
    res.status(500).send("Server Error");
  }
});
```

Obrázek 13 /discussions endpoint [zdroj: autor]

```
router.get("/discussions/:id", async (req, res) => {
  try {
    const { id } = req.params;
    const discussion = await pool.query(
      "SELECT discussions.*, users.username FROM discussions JOIN users ON discussions.author_id = users.id WHERE discussion_id = $1",
      [id]
    );
    res.json(discussion.rows[0]);
  } catch (err) {
    console.error(err.message);
    res.status(500).send("Server Error");
  }
});
```

Obrázek 14 /discussions/:id endpoint [zdroj: autor]

Příspěvky do diskuzí

Soubor **discussion_posts.js** definuje body API pro práci s příspěvky v diskuzích.

Koncový bod pro vytvoření nového příspěvku v diskuzi POST **/discussion_posts** umožňuje uživatelům přidávat nové příspěvky do diskuzí. Při vytváření nového příspěvku jsou poskytnuty informace o obsahu příspěvku a identifikátoru diskuze, do které má být příspěvek přidán. Před přidáním do databáze je uživatel ověřen přes JWT token.

```
router.post("/discussions_posts", async (req, res) => {
  try {
    const { content } = req.body;
    console.log(req.body);
    const author_id = jwt.verify(
      req.headers["x-auth-token"],
      process.env.JWT_SECRET
    ).id;
    const newDisscussionPost = await pool.query(
      "INSERT INTO discussion_posts (discussion_id, author_id, content) VALUES($1, $2, $3) RETURNING *",
      [req.body.discussion_id, author_id, content]
    );
    res.json(newDisscussionPost.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});
```

Obrázek 15 /discussions_posts endpoint [zdroj: autor]

Pro získání seznamu všech příspěvků v diskuzích GET **/discussion_posts** vrací seznam všech dostupných příspěvků. Pro získání příspěvků na základě zadaného identifikátoru GET **/discussion_posts/byDiscussionId/:discussion_id**, který umožňuje získat seznam příspěvků specifické diskuse.

```
router.get("/discussions_posts", async (req, res) => {
  try {
    const allDiscussionsPosts = await pool.query(
      "SELECT * FROM discussion_posts"
    );
    res.json(allDiscussionsPosts.rows);
  } catch (err) {
    console.error(err.message);
  }
});
```

Obrázek 16 /discussions_posts [zdroj: autor]

```
router.get(
  "/discussions_posts/byDiscussionId/:discussion_id",
  async (req, res) => {
    try {
      const { discussion_id } = req.params;
      const discussionsPostsByDiscussion = await pool.query(
        "SELECT discussion_posts.*, users.username FROM discussion_posts JOIN users ON discussion_posts.author_id = users.id WHERE discussion_id = $1",
        [discussion_id]
      );
      res.json(discussionsPostsByDiscussion.rows);
    } catch (err) {
      console.error(err.message);
    }
  }
);
```

Obrázek 17 /discussions_posts/byDiscussionsId/:discussion_id endpoint [zdroj: autor]

3.2.4 Novinky

Modul **news.js** definuje koncové body API pro práci s novinkami. Tyto koncové body umožňují uživatelům přidávat nové novinky a získávat seznam všech novinek.

Koncový bod pro vytvoření nové novinky POST **/news** umožňuje uživatelům přidávat nové novinky do aplikace.

```
router.post("/news", async (req, res) => {
  try {
    const { title, content } = req.body;
    const newNews = await pool.query(
      "INSERT INTO news (title, content) VALUES($1, $2) RETURNING *",
      [title, content]
    );
    res.json(newNews.rows[0]);
  } catch (err) {
    console.error(err.message);
  }
});
```

Obrázek 18 /news endpoint [zdroj: autor]

Bod pro získání seznamu všech novinek GET **/news**. Každá novinka obsahuje informace o nadpisu a obsahu.

```
router.get("/news", async (req, res) => {
  try {
    const allNews = await pool.query("SELECT * FROM news");
    res.json(allNews.rows);
  } catch (err) {
    console.error(err.message);
  }
});
```

Obrázek 19 /news endpoint [zdroj: autor]

3.2.5 Resetování hesla

Soubor **mailer.js** obsahuje koncové body API pro správu obnovy hesla pomocí e-mailů.

První koncový bod POST **/requestResetPassword** slouží k odeslání e-mailu s odkazem na obnovu hesla uživateli. Při zavolání tohoto endpointu uživatel poskytne svůj e-mail, na který mu bude zaslán e-mail s instrukcemi pro obnovu hesla.

Pokud uživatel s poskytnutým e-mailem existuje v databázi, je vygenerován resetovací token pomocí JWT, který má platnost jednu hodinu. Tento token je uložen do databáze spolu s identifikátorem uživatele. Poté je zaslán e-mail obsahující resetovací odkaz na uživatelův e-mail.

```
router.post("/requestResetPassword", async (req, res) => {
  const { email } = req.body;
  const user = await findByEmail(email);

  if (!user) {
    return res.status(404).json({ message: "User not found" });
  }
  const resetToken = jwt.sign({ id: user.id }, secretKey, { expiresIn: "1h" });

  await updateResetToken(user.id, resetToken);

  let mailOptions = {
    from: "danielbenes04@gmail.com",
    to: user.email,
    subject: "Resetovaní hesla",
    text: `Pro resetovaní hesla klikněte na tento odkaz: http://localhost:3000/reset-password?token=${resetToken}`,
  };

  transporter.sendMail(mailOptions, function (err, info) {
    if (err) {
      return res.status(500).json({ message: "Internal Server Error" });
    } else {
      console.log("Email sent: " + info.response);
      res.send("Heslo bylo poslano na email");
    }
  });
});
```

Obrázek 20 /requestResetPassword endpoint [zdroj: autor]

Druhý koncový bod POST **/resetPassword** slouží k provedení samotné obnovy hesla na základě resetovacího tokenu. Uživatel poskytne nové heslo a platný resetovací token. Pokud jsou všechny potřebné údaje poskytnuty, endpoint ověří platnost tokenu. Pokud je token platný, je uživatelovo heslo aktualizováno na základě poskytnutého nového hesla.

```
router.post("/resetPassword", async (req, res) => {
  const { password, token } = req.body;

  if (!password || !token) {
    return res.status(400).json({ message: "Missing data" });
  }

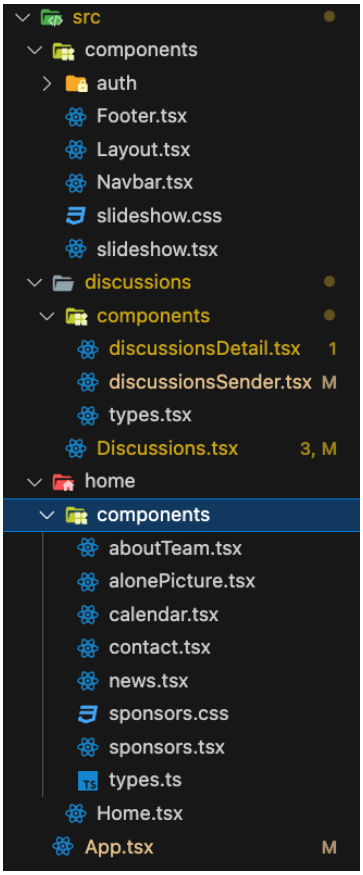
  try {
    const decoded = jwt.verify(token, secretKey);
    await updatePassword(decoded.id, password);
    res.send("Heslo bylo změněno");
  } catch (error) {
    return res.status(400).json({ message: "Invalid token" });
  }
});
```

Obrázek 21 /resetPassword endpoint [zdroj: autor]

Pro odesílání e-mailů je využita knihovna nodemailer. Konfigurační údaje pro přihlášení k e-mailovému účtu jsou uloženy přímo v souboru.

3.3 Frontend

Frontend aplikace SK Dolany je vyvinut pomocí moderních technologií jako React a Typescript. Aplikace je organizována do různých složek, jako jsou Discussions, Home a Auth. Každá z těchto složek obsahuje další podsložky s komponentami, které zajišťují různé funkcionality a uživatelské rozhraní. Tento strukturovaný přístup kódování pomáhá udržovat projekt přehledný a snadno spravovatelný. Komponenty jsou navrženy tak, aby byly opětovně použitelné a zároveň efektivně spolupracovaly s backendovou částí aplikace.



Obrázek 22 src adresář [zdroj: autor]

V adresáři **home** máme soubor **Home.tsx**, který slouží k renderování všech komponent uložených v podsložce **components**, jež se nachází uvnitř adresáře **home**. Je důležité si všimnout, že název souboru **Home.tsx** začíná velkým písmenem, zatímco názvy komponent začínají malým písmenem. Tento konzistentní styl názvů se dodržuje po celém projektu, což přispívá k přehlednosti a jednotnosti kódu.

Adresář **home** obsahuje všechny prvky související s hlavní stránkou aplikace. Zde se nacházejí komponenty pro zobrazení obrázků, informací o klubu, kalendáře, novinek, ale také kontaktní informace nebo soubory s CSS animacemi pro sponzory. Všechny tyto komponenty jsou vykreslovány v souboru **Home.tsx**.

V dalším důležitém adresáři **auth** naleznete AuthContext, který slouží jako ověřování přihlášeného uživatele v aplikaci, tam kde je to potřeba. Kromě toho zde najdete komponenty pro resetování hesla, zobrazení uživatelského profilu a také komponenty pro přihlášení a registraci uživatele.

V adresáři **discussions** se soustředíme na diskuse. Zde naleznete soubor **Discussions.tsx**, který zobrazuje všechny dostupné diskuse. Tyto diskuse jsou viditelné pouze přihlášeným uživatelům, kteří mají příslušná oprávnění. Diskuse jsou rozděleny do čtyř kategorií: veřejné, pro obyvatele Dolan, pro hráče SK a pro management. Dále v tomto adresáři najdete soubor **discussionsDetail.tsx**, který umožňuje přímý přístup k určité diskusi a přidávání příspěvků do ní. Tento adresář také obsahuje komponentu **discussionsSender.tsx**, která slouží k odesílání příspěvků do diskuse.

V adresáři **components**, který je podsložkou adresáře **src**, najdeme "globální" komponenty aplikace, které jsou přístupné z různých částí projektu. Mezi tyto komponenty patří například navigační lišta (navbar), patička (footer) a layout celé aplikace. Dále zde nalezneme komponentu **slideshow.tsx** spolu se stylovým souborem **slideshow.css**. Tato komponenta se stará o animované přechody mezi stránkami v navigační liště a je zobrazena pouze na hlavní stránce aplikace.

3.3.1 Tailwind CSS

V souboru **tailwind.config.js** definuji vlastní konfiguraci pro framework Tailwind CSS, která ovlivňuje vzhled a chování mého projektu. Jednou z klíčových funkcí této konfigurace je možnost definovat vlastní barevnou paletu, která ovlivňuje barvy používané v celém projektu.

V konkrétním kódu, který jsem poskytl, můžete vidět, že jsem specifikoval různé barvy pod klíčem **colors**. Zde mohu definovat své vlastní barvy a použít je ve svém projektu. Například **mygreen** a **mygreen2** definují moje vlastní odstíny zelené, které chci použít.

Celkově mi soubor "tailwind.config.js" umožňuje plně přizpůsobit vzhled a chování Tailwind CSS podle potřeb mého projektu.

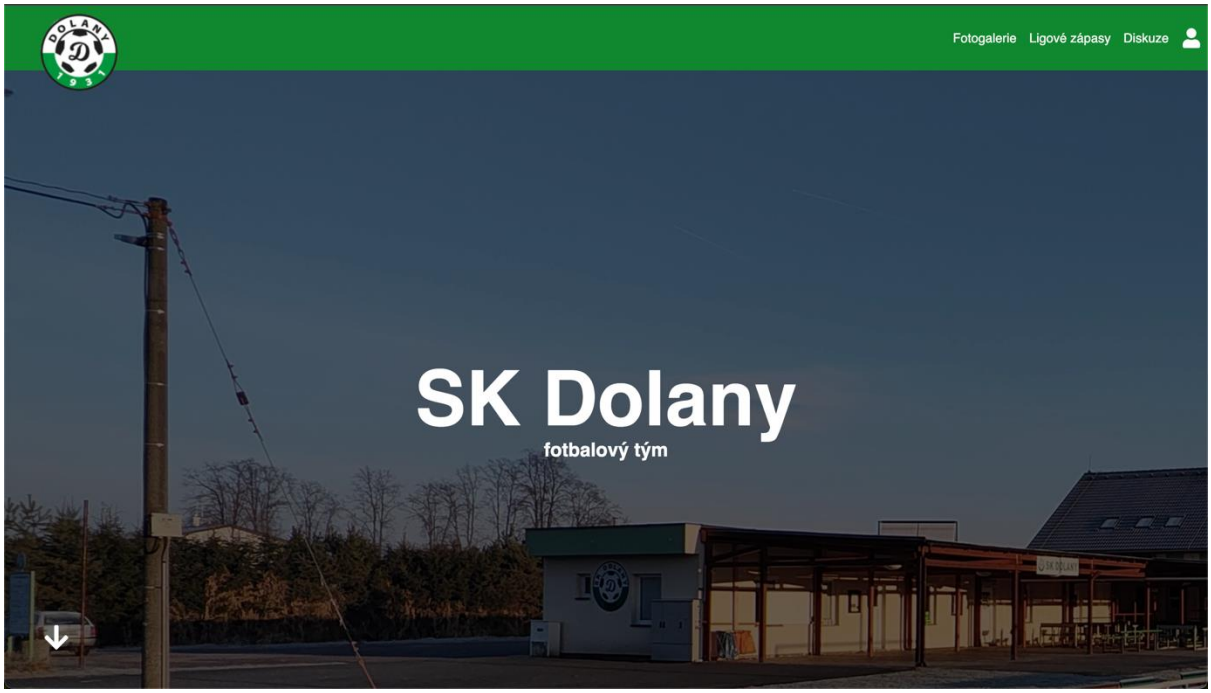
```
module.exports = {
  content: ["./src/**/*.{js,jsx,ts,tsx}"],
  theme: {
    extend: {
      backgroundColor: {
        default: "#ededed",
      },
      textColor: {
        white: "#fff",
      },
    },
    colors: {
      mygreen: "#128831",
      mygreen2: "#BAFFCD",
      white: "#fff",
      black: "#000",
      grey: "#ededed",
      red: "#E03D26",
    },
  },
  plugins: [],
};
```

Obrázek 23 tailwind.config.js [zdroj: autor]

4 Webová aplikace

4.1 Navigační menu

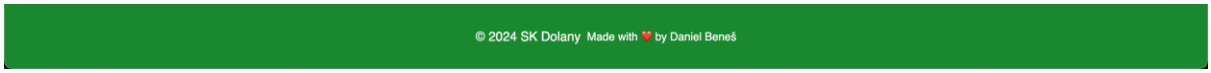
V navigačním menu najdete logo aplikace, které mírně přesahuje přes okraj a po kliknutí vás přenese na úvodní stránku. Dále zde najdete odkaz na galerii, která je hostovaná na obecním serveru, informace o ligových zápasech od FAČR, sekci pro diskuze a uživatelský profil, včetně možnosti přihlášení nebo registrace. Součástí navigačního menu na hlavní stránce je také slideshow s obrázky fotbalového hřiště a okolí.



Obrázek 24 Navigační menu [zdroj: autor]

4.2 Patička

V patičce se nachází aktuální rok a název fotbalového klubu, taky se zde nachází moje jméno.



Obrázek 25 patička [zdroj: autor]

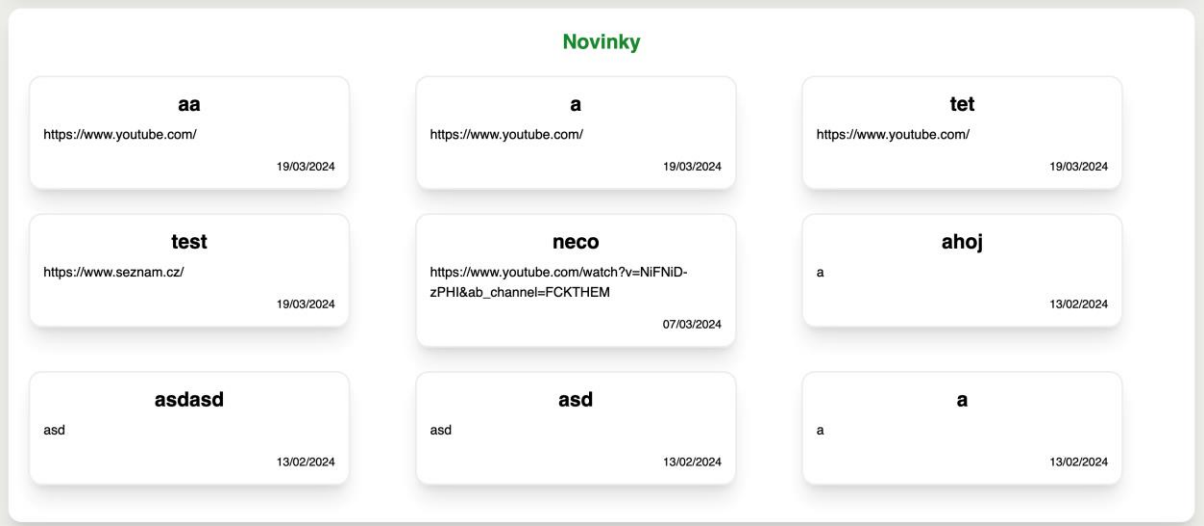
4.3 Úvodní stránka

Na úvodní stránce nás vítá krátký článek o historii klubu, doprovázený fotografií samotného sportovního areálu. Dále následuje sekce s další fotografií spojenou s klubem a integrovaným Google kalendářem, propojeným s oficiálním účtem klubu. V další části stránky se zobrazují nejnovější příspěvků, přičemž je vždy vidět pouze 9 nejaktuálnějších příspěvků. Pod novinkami najdeme kontaktní informace klubu a animovanou komponentu s logy sponzorů, doplněnou o odkazy vedoucí na jejich webové stránky.



Obrázek 26 Úvodní stránka [zdroj: autor]

Novinky na úvodní stránce.



Obrázek 27 novinky [zdroj: autor]

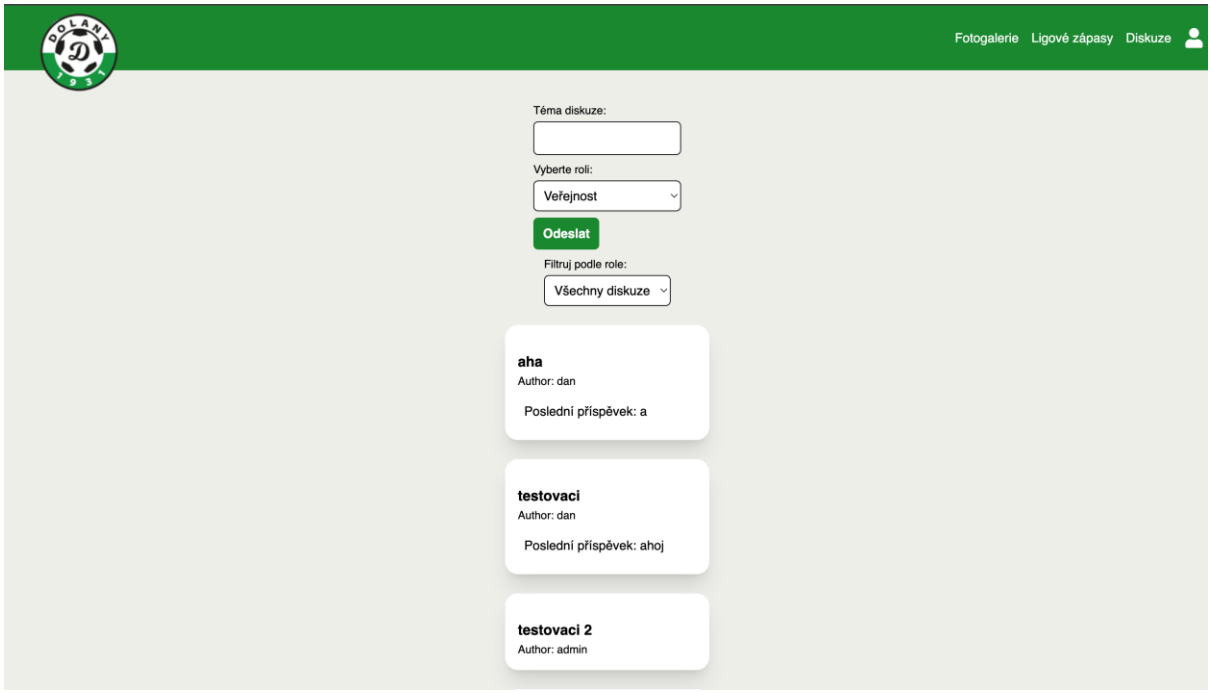
Kontakt a komponent s animací sponzorů



Obrázek 28 kontakt a sponzoři [zdroj: autor]

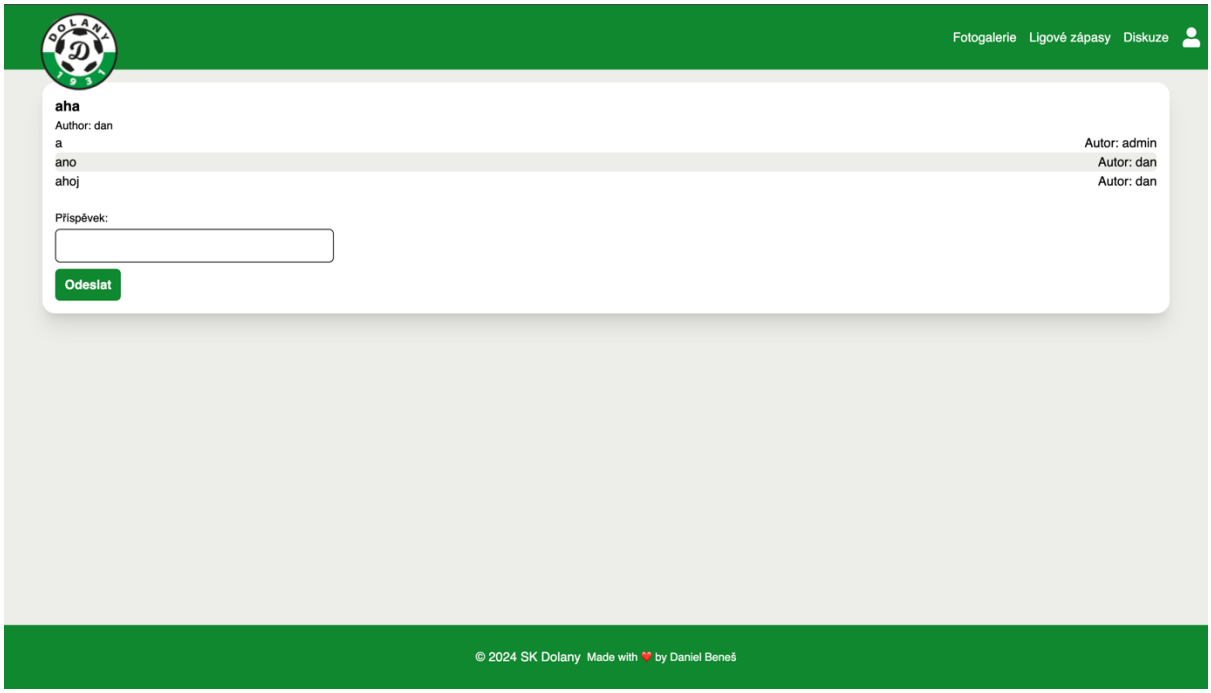
4.4 Vytvoření diskuze

Pro vytvoření nové diskuze je nejprve potřeba zadat téma diskuse a zvolit, pro koho je určena. Na stránce s diskuzemi pak uživatel uvidí všechny dostupné diskuse, na které má oprávnění přistupovat. Může také filtrovat diskuse podle role. V seznamu diskuzí je zobrazen název diskuze, její autor a informace o posledním příspěvku. Takto je uživatelovi umožněno snadné procházení a vyhledávání diskuzí, které ho zajímají. Do diskuzí se uživatel dostane skrz navigační menu.



Obrázek 29 diskuze [zdroj: autor]

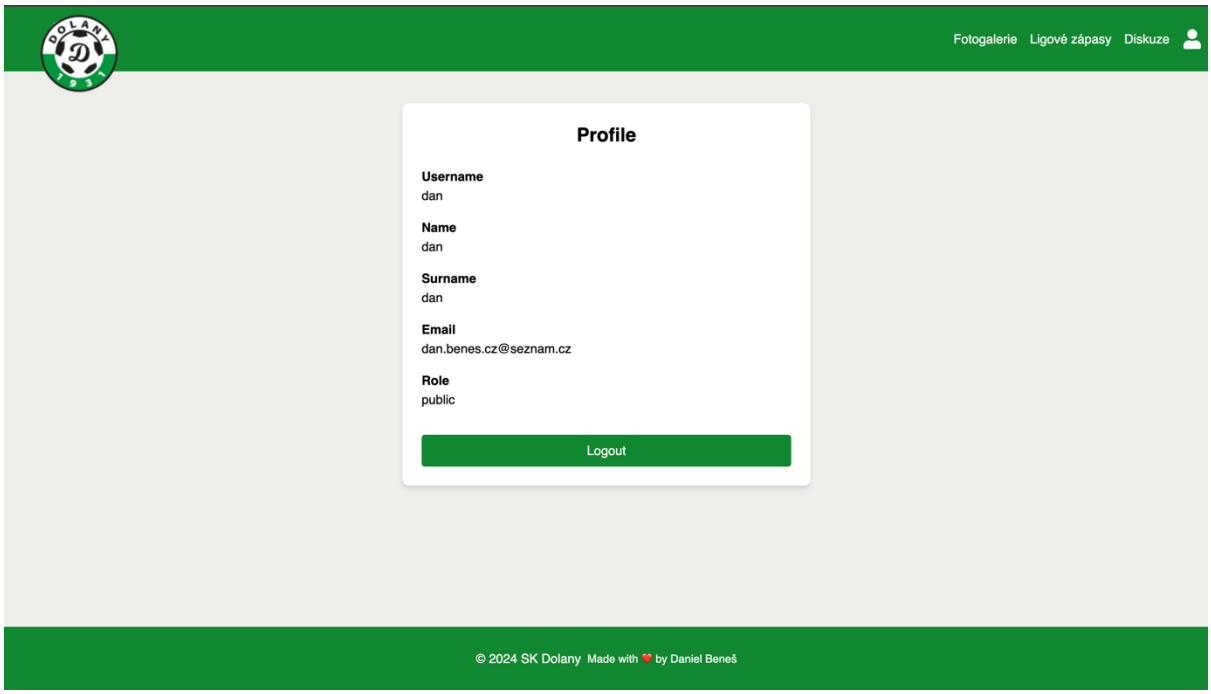
Po kliknutí na konkrétní diskuzi se uživatel přesune na stránku s detailem této diskuze. Zde má možnost přidávat příspěvky a komunikovat s ostatními účastníky, kteří mají přístup k téže diskuzi. Detail diskuze zahrnuje veškeré informace o dané diskuzi a umožňuje uživateli interagovat s ostatními účastníky.



Obrázek 30 detail diskuze [zdroj: autor]

4.5 Profil

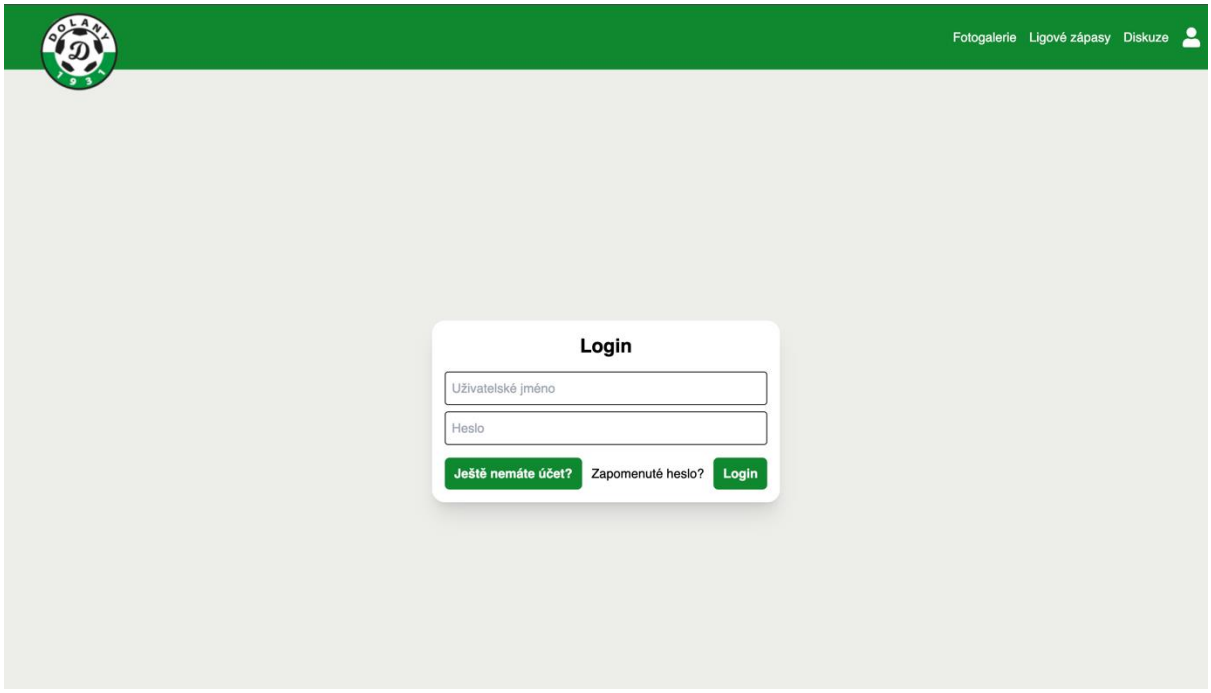
V uživatelském profilu jsou shromážděny veškeré informace, které uživatel poskytl při vytváření účtu, s výjimkou hesla. Profilová stránka také obsahuje tlačítko pro odhlášení z účtu. Do profilu se uživatel dostane skrz navigační menu.



Obrázek 31 uživatelský profil [zdroj: autor]

4.6 Přihlášení

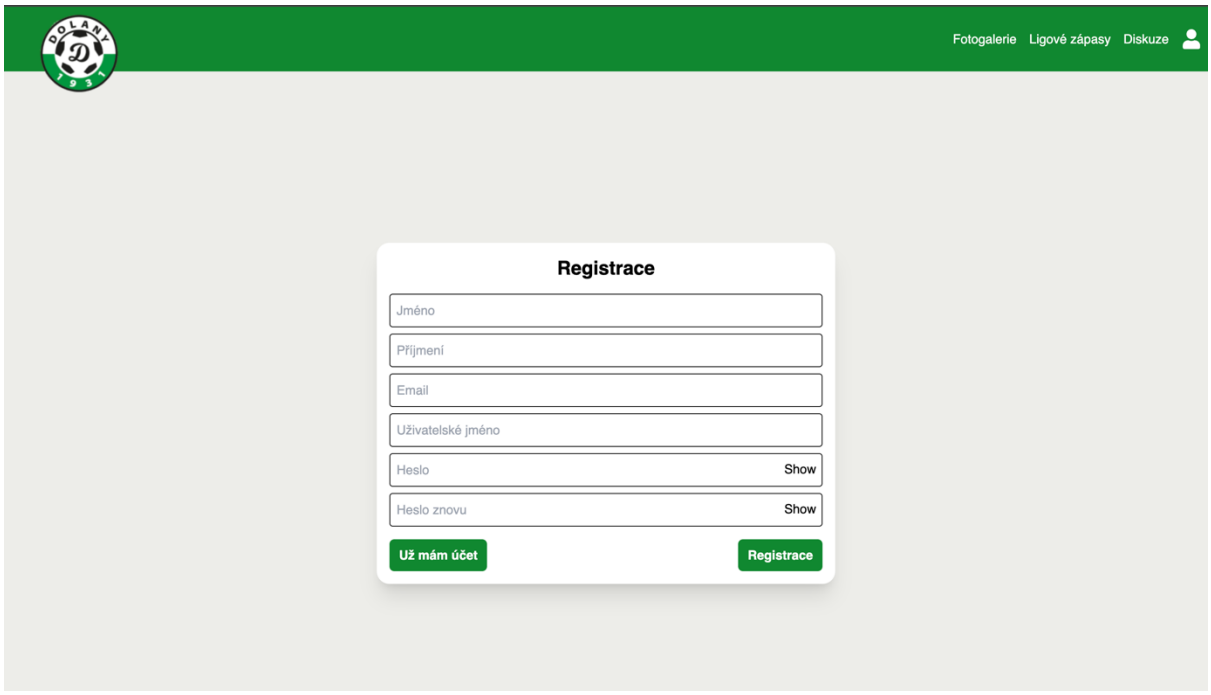
Přihlášení probíhá zadáním uživatelského jména a hesla. Potom stačí kliknout na zelené tlačítko login.



Obrázek 32 login [zdroj: autor]

4.7 Registrace

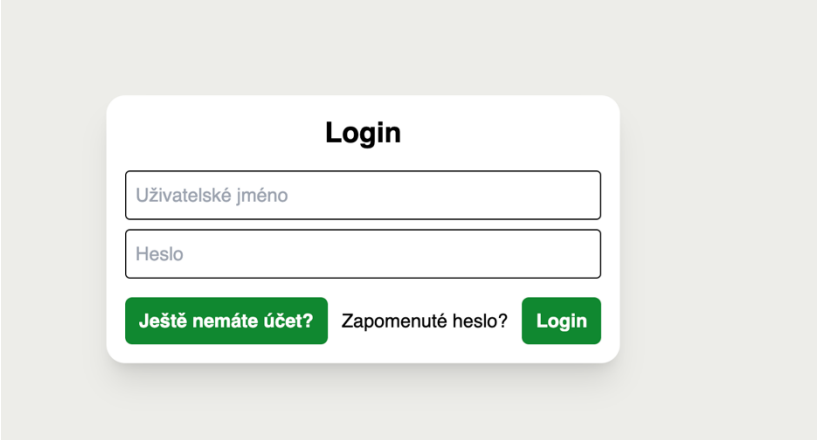
Registrace uživatele je přístupná po kliknutí na tlačítko "Ještě nemáte účet?". Po této akci je uživatel přesměrován na registrační formulář, kde zadává své údaje, jako je jméno, příjmení, e-mailová adresa, uživatelské jméno a heslo. Po úspěšném dokončení registrace je uživateli automaticky přidělena role veřejnosti.



Obrázek 33 registrace [zdroj: autor]

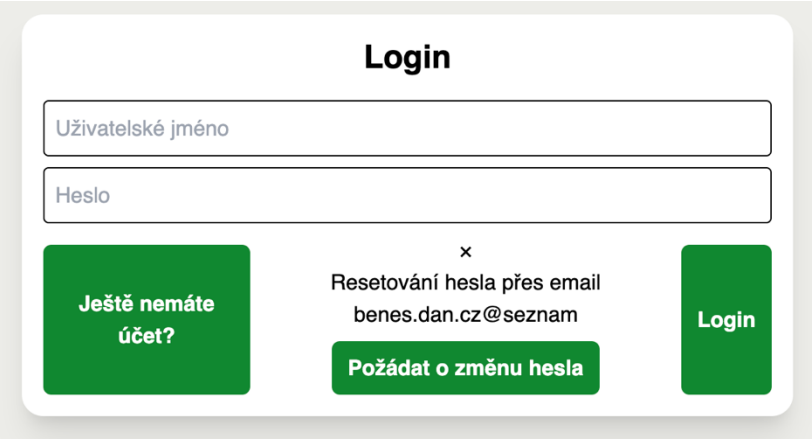
4.8 Změna hesla

Uživatel má možnost změny hesla na stránce login kde může zvolit možnost zapomenuté heslo. Login se rozbalí a uživatel, má možnost zadat jeho email, který zadal při registraci. Uživateli přijde na email link, který si rozklikne a je mu nabídnuta změna hesla.



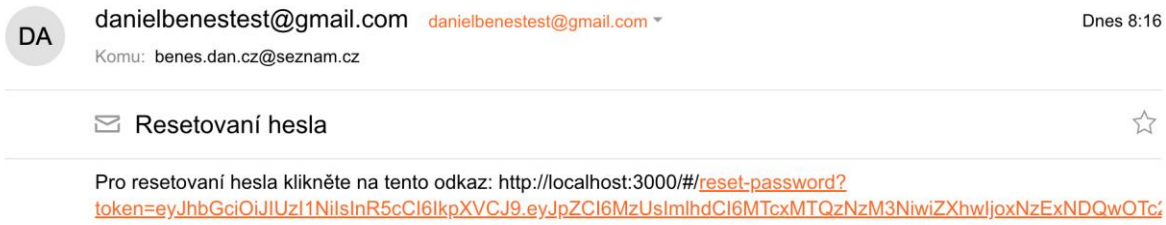
Obrázek 34 změna hesla [zdroj: autor]

Po kliknutí na “Zapomenuté heslo?“



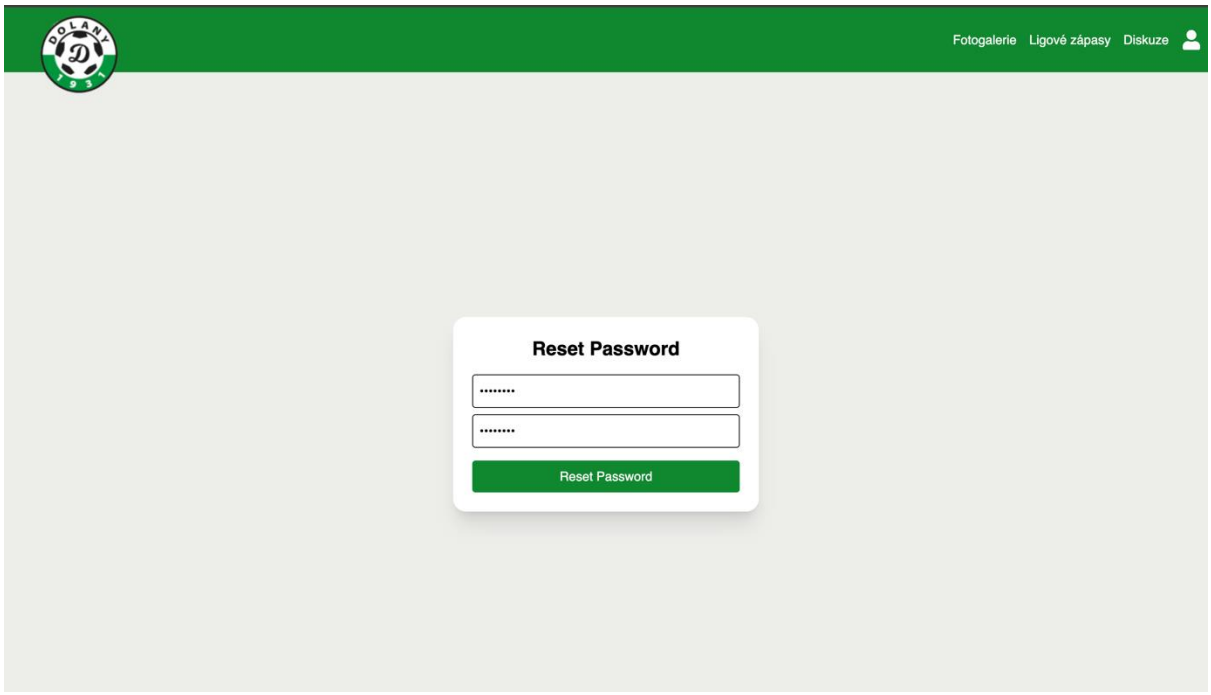
Obrázek 35 změna hesla [zdroj autor]

Email, který přijde uživateli.



Obrázek 36 email pro resetování hesla [zdroj: autor]

Místo kde je uživatel odkázán na změnu hesla.



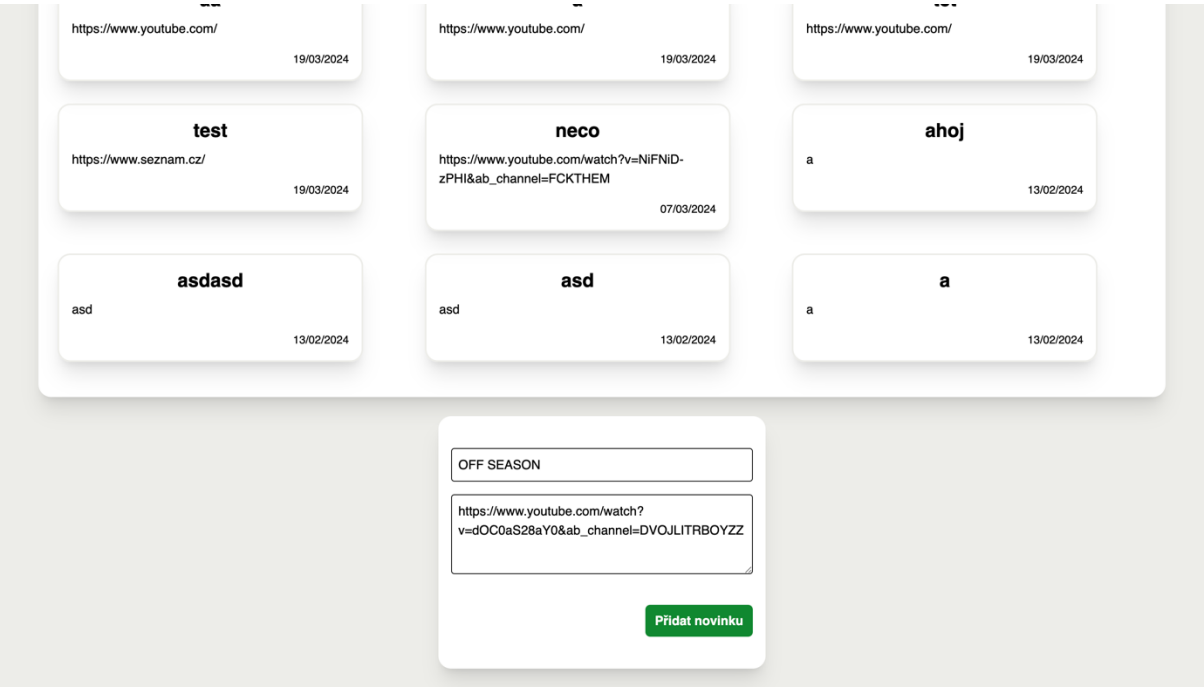
Obrázek 37 resetování hesla [zdroj: autor]

5 Admin

Administrační rozhraní slouží k řízení a správě různých funkcí a obsahu aplikace, přičemž administrátor má k dispozici rozšířené možnosti a práva, která běžný uživatel nemá. Jednou z důležitých funkcí administrátora je možnost přidávání novinek na úvodní stránce aplikace.

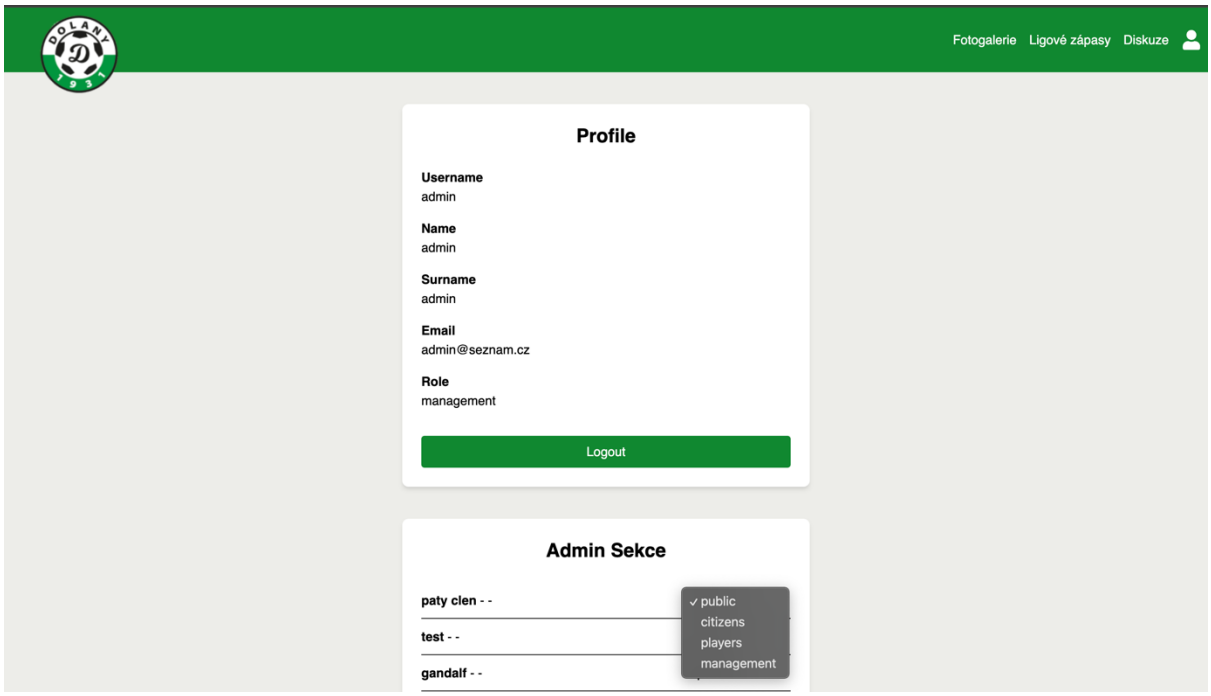
Další klíčovou úlohou administrátora je správa rolí ostatních uživatelů. Administrátor má možnost nastavovat a upravovat role jednotlivých uživatelů, tuto funkci najde ve svém uživatelském profilu.

Přidání novinky na úvodní stránce, admin zadává titulek a obsah.



Obrázek 38 admin novinky [zdroj: autor]

Změna role uživatele.



Obrázek 39 změna role [zdroj: autor]

6 Budoucnost aplikace

6.1 Co v aplikaci chybí?

V aplikaci by bylo vítané začlenění možnosti přidávat ankety do diskuzí, což by poskytlo uživatelům prostředek k hlasování o různých událostech, jako například účast na tréninku nebo zápase.

Na straně frontendu je třeba provést další optimalizace pro mobilní zařízení, aby uživatelé mohli aplikaci pohodlněji používat na svých telefonech.

Dále zde třeba chybí funkcionalita mazání vlastního účtu, nebo lepší možnosti chatování, třeba přidávání fotografií, nebo posílaná dokumentů.

6.2 Co bude dál

Aplikace bude nasazena na oficiální web klubu, a proto je nezbytné, aby všechny nedostatky byly do konce vyřešeny. I po dokončení maturitní práce budu na projektu nadále pracovat. První oblastí, kterou budu dále rozvíjet, jsou ankety v diskusích.

7 Závěr

Cílem aplikace bylo vytvořit webovou stránku pro fotbalový klub, která kombinuje informační část s částí dostupnou pouze po přihlášení. Domnívám se, že jsem úspěšně dosáhl všech stanovených cílů. Jsem spokojený s výsledkem aplikace i s designem, který jsem sám navrhl. Jediným nedostatkem, je responzivita webu, která není optimální. Při práci na tomto projektu jsem se posunul v používání technologie Tailwind CSS, což byla pro mě velká výzva. Celkově jsem přesvědčen, že volba technologií a použití PERN stacku byla správná. Teď se těším na zpětnou vazbu od uživatelů webu, která mi pomůže v dalším vylepšení a úpravách.

8 Literatura

1. EROLIN, Justice. *A Comprehensive Guide to Server-Side Rendering in React* [online]. [cit. 2024-03-27]. Dostupné z: https://www.bairesdev.com/blog/server-side-rendering-react/
2. VOROBIOV, Igor. *10 interesting facts about TypeScript* [online]. [cit. 2024-03-27]. Dostupné z: https://medium.com/@ivorobioff/10-facts-about-typescript-7204a547ae08
3. ABBA, Ihechikara. *How to Use Tailwind CSS to Rapidly Develop Snazzy Websites* [online]. [cit. 2024-03-27]. Dostupné z: https://kinsta.com/blog/tailwind-css/
4. , Asian Digital Hub. *What is Node.js and How it Work?* [online]. [cit. 2024-03-27]. Dostupné z: https://medium.com/@asiandigitalhub/what-is-node-js-and-how-it-work-490f5ecba665
5. KHAN, Shahzaib. *Building Web Applications with Express.js: A Comprehensive Guid* [online]. [cit. 2024-03-27]. Dostupné z: https://medium.com/@skhans/building-web-applications-with-express-js-a-comprehensive-guide-113a77be1b11
6. *PostreSQL* [online]. [cit. 2024-03-27]. Dostupné z: https://www.postgresql.org/about/
7. IVANOV, Vova. *Deploying a PERN stack application to AWS* [online]. [cit. 2024-03-27]. Dostupné z: https://www.pulumi.com/blog/deploying-a-pern-stack-application-to-aws/
8. , Extio Technology. *Understanding JSON Web Tokens (JWT): A Secure Approach to Web Authentication* [online]. [cit. 2024-03-27]. Dostupné z: https://medium.com/@extio/understanding-json-web-tokens-jwt-a-secure-approach-to-web-authentication-f551e8d66deb
9. RAM, Prashant. *Node JS with Passport Authentication simplified* [online]. [cit. 2024-03-27]. Dostupné z: https://medium.com/@prashantramnyc/node-js-with-passport-authentication-simplified-76ca65ee91e5
10. GRIGUTYTĖ, Monika. *What is bcrypt and how does it work?* [online]. [cit. 2024-03-27]. Dostupné z: https://nordvpn.com/blog/what-is-bcrypt/

Obrázek 1 Architektura projektu [zdroj:] 10
Obrázek 2 use case diagram [zdroj: Martin Zeman] 11
Obrázek 3 databázové připojení [zdroj: autor] 12
Obrázek 4 /register endpoint [zdroj: autor] 12
Obrázek 5 /login endpoint [zdroj: autor] 13
Obrázek 6 adresář utils [zdroj: autor] 13
Obrázek 7 jwt.js [zdroj: autor] 13
Obrázek 8 passport.js [zdroj.js] 14
Obrázek 9 User.js [zdroj: autor] 14
Obrázek 10 User.js [zdroj: autor] 14
Obrázek 11 protected.js [zdroj: autor] 15
Obrázek 12 /discussions endpoint [zdroj: autor] 16
Obrázek 13 /discussions endpoint [zdroj: autor] 16
Obrázek 14 /discussions/:id endpoint [zdroj: autor] 16
Obrázek 15 /discussions_posts endpoint [zdroj: autor] 17
Obrázek 16 /discussions_posts [zdroj: autor] 17
Obrázek 17 /discussions_posts/byDiscussionsId/:discussion_id endpoint [zdroj: autor] 17
Obrázek 18 /news endpoint [zdroj: autor] 18
Obrázek 19 /news endpoint [zdroj: autor] 18
Obrázek 20 /requestResetPassword endpoint [zdroj: autor] 19
Obrázek 21 /resetPassword endpoint [zdroj: autor] 20
Obrázek 22 src adresář [zdroj: autor] 21
Obrázek 23 tailwind.config.js [zdroj: autor] 23
Obrázek 24 Navigační menu [zdroj: autor] 24
Obrázek 25 patička [zdroj: autor] 24
Obrázek 26 Úvodní stránka [zdroj: autor] 25
Obrázek 27 novinky [zdroj: autor] 25
Obrázek 28 kontakt a sponzoři [zdroj: autor] 26
Obrázek 29 diskuze [zdroj: autor] 27
Obrázek 30 detail diskuze [zdroj: autor] 28
Obrázek 31 uživatelský profil [zdroj: autor] 29
Obrázek 32 login [zdroj: autor] 30
Obrázek 33 registrace [zdroj: autor] 31
Obrázek 34 změna hesla [zdroj: autor] 31
Obrázek 35 změna hesla [zdroj autor] 32
Obrázek 36 email pro resetování hesla [zdroj: autor] 32
Obrázek 37 resetování hesla [zdroj: autor] 33
Obrázek 38 admin novinky [zdroj: autor] 34
Obrázek 39 změna role [zdroj: autor] 35