SRS - Specifikace požadavků na software

# 1.Úvod

## 1.1 Název aplikace: CompanyTrack

1.2 Účel dokumentu

Cílem tohoto dokumentu je specifikovat funkční a nefunkční požadavky na systém pro správu firem, jejich poboček a zaměstnanců. Dokument definuje rozsah systému, role uživatelů, klíčové funkcionality a požadavky na bezpečnost, výkon a použitelnost.

## 1.3 Rozsah systému

CompanyTrackje webová aplikace umožňující:

Správu firem a jejich organizační struktury (pobočky, kanceláře),

Správu zaměstnanců a jejich přiřazení k firmám/pobočkám,

Zobrazení tržeb, nákladů a zisků pomocí grafů a tabulek,

Podporu pro více uživatelských rolí (např. HR, účetní, manažer),

Přihlášení pomocí Supabase autentizace (JWT tokeny),

Plně oddělený frontend (Next.js) a backend (Nest.js) s OpenAPI specifikací.

Každá pobočka ve firmě bude moci mít určité nastavení.

## 1.4 Cílová skupina

**Superadmin –** nejvyšší oprávnění, správa celého systému

**HR (Human Resources) –** správa zaměstnanců a jejich údajů

**Manažer –** dohled nad pobočkami a výkonem týmu

**Účetní (Accountant) –** přístup k ekonomickým datům

**Zaměstnanec (User) –** omezený přístup k vlastním datům

## 1.5 Definice pojmů a zkratek

* + JWT - JSON Web Token – technologie pro bezpečné přihlášení uživatelů
  + Supabase – Backendová platforma (PostgreSQL, autentizace)
  + ERD - Entity-Relationship Diagram – model databázové struktury
  + OpenAPI / Swagger - Standard pro popis REST API

# 2 Obecný popis

## 2.1 Uživatelské role

* Superadmin – Majitel firmy, přístup všude, všechny pravomoce
* HR – správa údajů zaměstnanců firmy, úprava dat, starost o pozvání nových zam.
* Manažer – správa zaměstnanců, správa výdajů, správa docházky, přehled o statistikách a určitých výdajích (podle nastavení firmy)
* Účetní – možnost zjistit veškeré výdaje, příjmy a zisky (podle nastavení firmy)
* Změstnanec – Vidé svou docházku a cíle firmy. Nic vice.

## 2.2 Use Case scénáře

### 2.2.1 Registrace nového uživatele

### Primární aktér: Nový uživatel

### Předpoklad: Uživatel není přihlášen a dosud nemá vytvořený účet

### Popis scénáře:

### 1. Uživatel navštíví stránku registrace

### 2. Zadá e-mail a heslo

### 3. Klikne na „Registrovat“

### 4. Systém přes Supabase vytvoří účet a odešle ověřovací e-mail

### 5. Uživatel otevře svůj e-mail a klikne na ověřovací odkaz

### 6. Systém ověří e-mail a umožní přihlášení

### 7. Uživatel se může poprvé přihlásit a používat aplikaci

### Alternativní scénáře:

### • 2a. Uživatel zadá e-mail, který již existuje → systém zobrazí chybu

### • 5a. Uživatel neklikne na odkaz → při přihlášení mu systém oznámí, že e-mail není ověřen

### • 6a. Ověřovací odkaz vyprší → systém nabídne možnost zaslání nového

### **Výsledek: Uživatelský účet je úspěšně zaregistrován a ověřen**

### 2.2.2 Založení firmy

**Název případu užití**: Založení nové firmy

**Primární aktér**: Superadmin nebo nový uživatel

**Předpoklady**: Uživatel je přihlášen a nemá zatím žádnou firmu nebo chce založit druhou firmu

**Popis scénáře**:

1. Uživatel klikne na „Založit firmu“
2. Vyplní název firmy a základní údaje (IČO, adresa, kontaktní osoba…)
3. Odešle formulář
4. Systém uloží firmu do databáze
5. Firma se vytvoří se 14denní zkušební lhůtou
6. Uživatel vidí přehled své nové firmy

**Alternativní scénáře**:

* 2a. Zadané IČO už existuje → systém zobrazí chybu
* 4a. Dojde k chybě při ukládání → systém oznámí selhání

**Výsledek**: Nová firma je založena a připravena k používání ve zkušebním režimu

### 2.2.3 Přidání pobočky k firmě

**Název případu užití**: Přidání nové pobočky

**Primární aktér**: manažer nebo Superadmin

**Předpoklady**: Firma již existuje

**Popis scénáře**:

1. Uživatel otevře detail firmy
2. Klikne na „Přidat pobočku“
3. Vyplní název, adresu a případně popis pobočky
4. Odešle formulář
5. Systém uloží novou pobočku do databáze

**Alternativní scénáře**:

* 3a. Nevyplněné povinné pole → systém oznámí chybu
* 4a. Poboček je více než dovoluje plán → upozornění na nutnost upgradu

**Výsledek**: Nová pobočka je přidána k firmě

### 2.2.4 Přidání zaměstnance

**Název případu užití**: Vytvoření zaměstnance

**Primární aktér**: HR, manažer nebo superadmin

**Předpoklady**: Firma i pobočka existují

**Popis scénáře**:

1. HR otevře detail firmy nebo pobočky
2. Klikne na „Přidat zaměstnance“
3. Vyplní jméno, e-mail, roli a přiřazení k pobočce
4. Odešle formulář
5. Systém vytvoří zaměstnance v databázi
6. Systém může odeslat e-mail pro nastavení hesla (budoucí krok)

**Alternativní scénáře**:

* 3a. Neplatný e-mail → systém zobrazí chybu
* 5a. Zaměstnanců je více než dovoluje plán → oznámení o překročení limitu

**Výsledek**: Zaměstnanec je přidán k firmě a pobočce

### 2.2.5 Zobrazení tržeb a nákladů

**Název případu užití**: Analýza výkonu firmy/pobočky

**Primární aktér**: Manažer nebo Účetní

**Předpoklady**: Firma a pobočky obsahují obchodní data

**Popis scénáře**:

1. Uživatel přejde do sekce „Analýza“ nebo „Dashboard“
2. Vybere firmu nebo konkrétní pobočku
3. Systém zobrazí přehled tržeb, nákladů a ziskovosti
4. Uživatel může filtrovat data podle data nebo dalších parametrů

**Alternativní scénáře**:

* 2a. Neexistují žádná data → systém zobrazí zprávu „Žádná data k dispozici“
* 4a. Uživateli chybí oprávnění → systém odepře přístup

**Výsledek**: Uživatel získá přehled o ekonomických výsledcích firmy nebo pobočky

### 2.2.6 Přihlášení uživatele

**Název případu užití**: Přihlášení do systému

**Primární aktér**: Registrovaný uživatel

**Předpoklady**: Uživatel má ověřený e-mail a existující účet

**Popis scénáře**:

1. Uživatel navštíví přihlašovací stránku
2. Zadá e-mail a heslo
3. Klikne na „Přihlásit“
4. Supabase ověří přihlašovací údaje
5. Systém vygeneruje JWT a uloží ho do cookies
6. Uživatel je přesměrován do aplikace

**Alternativní scénáře**:

* 2a. Špatné přihlašovací údaje → systém zobrazí chybu
* 4a. Účet není ověřen → systém oznámí nutnost potvrzení e-mailu

**Výsledek**: Uživatel je úspěšně přihlášen do systému a má přístup podle své role

## 2.3 Architektura

Systém CompanyTrack je navržen jako dvouvrstvá aplikace s odděleným frontendem a backendem.

**- Frontend:** Implementován v Next.js, poskytuje rozhraní pro uživatele.

**- Backend:** Nest.js, poskytuje REST API definované pomocí OpenAPI specifikace.

**- Databáze:** PostgreSQL spravovaná pomocí Prisma ORM.

**- Autentizace:** Supabase Auth s podporou JWT tokenů.

**- Hostování:** Využívá Docker kontejnery pro snadné nasazení.

# 3 Funkční požadavky

## 3.1 Registrace uživatele

* Systém umožňuje novým uživatelům registraci pomocí e-mailu a hesla.
* Registrace je realizována přes Supabase Auth.
* Po úspěšném odeslání registračního formuláře je na zadaný e-mail odeslán ověřovací odkaz.
* Uživatel musí potvrdit e-mail kliknutím na tento ověřovací odkaz, aby mohl dokončit registraci a přihlásit se do systému.
* Systém ověří, zda byl e-mail potvrzen – bez toho není umožněn vstup do aplikace.
* E-mail i heslo jsou při přenosu šifrovány.

## 3.2 Založení firmy

* Systém umožní přihlášenému uživateli založit firmu vyplněním názvu a základních údajů.
* Systém při založení nastaví 14denní zkušební období, během kterého jsou aktivní omezení (max. 2 pobočky, max. 10 zaměstnanců).
* Po uložení systém zobrazí detail firmy.
* Systém informuje uživatele, pokud firma se stejným IČO již existuje.

## 3.3 Přidání pobočky k firmě

* Systém umožní manažerovi nebo superadminovi přidat pobočku k existující firmě.
* Pobočka obsahuje název, adresu a volitelně popis.
* Systém ověří, zda firma nepřekročila maximální počet poboček dle plánu.
* Pobočka je uložena a přidána do přehledu poboček firmy.

## 3.4 Přidání zaměstnance

* HR, manažer nebo superadmin může přidat nového zaměstnance k firmě a přiřadit ho k existující pobočce.
* Záznam zaměstnance obsahuje jméno, e-mail, roli, pobočku.
* Systém validuje, zda firma nepřekročila limit zaměstnanců podle plánu.
* Systém může (volitelně) poslat e-mail zaměstnanci s výzvou k nastavení hesla.

## 3.5 Zobrazení tržeb a nákladů

* Manažer nebo účetní může zobrazit přehled výnosů, nákladů a ziskovosti na úrovni firmy nebo pobočky.
* Data jsou vizualizována pomocí tabulek a grafů.
* Uživatel může filtrovat zobrazená data podle času nebo pobočky.
* Systém kontroluje oprávnění pro přístup k těmto datům.

## 3.6 Přihlašení uživatele

* Systém umožní přihlášení pomocí e-mailu a hesla přes Supabase.
* Po úspěšném přihlášení systém uloží JWT token do cookies.
* Systém ověří, že uživatel má ověřený e-mail, než mu umožní přístup.
* Na základě JWT tokenu je určena role uživatele a jeho oprávnění.

# 4 Nefunkční požadavky

## 4.1 Výkon (Performance)

* Aplikace načte hlavní dashboard do 2 sekund na běžném připojení (min. 10 Mbps).
* CRUD operace (např. přidání zaměstnance) proběhne do 1 sekundy od odeslání požadavku.
* Grafy a přehledy musí být optimalizované tak, aby zobrazovaly výsledky do 1–2 sekund při objemu do 1000 záznamů.

## 4.2 Dostupnost a spolehlivost (Availability & Reliability)

* Backend běží na platformě Supabase s SLA ≥ 99.9 %.
* Aplikace zvládne současně obsloužit minimálně 100 uživatelů bez znatelného zpomalení.
* Při výpadku připojení k Supabase zobrazí aplikace uživatelsky srozumitelnou chybovou hlášku a umožní obnovit relaci.

## 4.3 Bezpečnost (Security)

* Autentizace probíhá pomocí Supabase a ověřeného e-mailu.
* Přístup k datům je omezen dle uživatelských rolí (RBAC – Role Based Access Control).
* JWT tokeny jsou ukládány do HttpOnly cookies s nastaveným Secure a SameSite atributem.
* Citlivá data (např. hesla) nejsou nikdy ukládána ani zobrazována ve frontendové aplikaci.
* Veškerá komunikace mezi frontendem a backendem probíhá přes HTTPS.

## 4.4 Použitelnost (Usability)

* Rozhraní bude přehledné a responzivní – použitelné jak na desktopu, tak na mobilních zařízeních.
* Aplikace musí být srozumitelná i pro uživatele bez technických znalostí (např. HR).
* Uživatelé musí být schopni vykonat klíčové akce (např. přidání zaměstnance, zobrazení tržeb) bez školení do 5 minut.

## 4.5 Údržba a rozšiřitelnost (Maintainability & Scalability)

* Backendová architektura je postavená modulárně dle vrstev (Controller, Service, Repository, DTO, Entity).
* Nové endpointy a hooky lze generovat automatizovaně pomocí Orval a Plop CLI.
* Aplikace umožňuje snadné přidání nových entit, rolí a oprávnění bez zásahu do základní architektury.

## 4.6 Kompatibilita (Compatibility)

* Aplikace musí být plně funkční v moderních prohlížečích (Chrome, Firefox, Safari, Edge).
* Frontend je testován pro minimální rozlišení 360x640 (mobilní zobrazení).

## 4.7 Lokalizace a jazyk

* Aplikace je primárně v českém jazyce, texty musí být spravovatelné pomocí překladu (i18n ready).

# 5 Omezení

Tato část specifikuje technická a provozní omezení, která mohou ovlivnit vývoj, nasazení nebo provoz aplikace.

## 5.1 Technologická omezení

* Aplikace je postavena na **Next.js**, a tedy je závislá na Node.js prostředí pro běh serverových funkcí.
* Autentizace a správa uživatelů jsou zajištěny pomocí **Supabase Auth**, což znamená, že změna poskytovatele autentizace by vyžadovala zásadní úpravy backendu i frontendu.
* Data jsou ukládána do **PostgreSQL** databáze (Supabase nebo lokální), což omezuje podporu na SQL syntaxi a PostgreSQL struktury.

## 5.2 Provozní omezení

* Aplikace je navržena pro běžné webové prohlížeče – nepodporuje starší verze (např. Internet Explorer).
* Nasazení předpokládá dostupnost cloudu (např. Vercel, Supabase hosting); offline nebo intranetové nasazení není prioritně podporováno.

## 5.3 Uživatelská omezení

* V bezplatné verzi aplikace (14denní trial) je omezení na **max. 2 pobočky** a **10 zaměstnanců**.
* Uživatelské role jsou pevně definované (Superadmin, HR, Manažer, Účetní, Zaměstnanec), nové role nelze zatím dynamicky přidávat přes UI.

## 5.4 Vývojová omezení

* Projekt je navržen s ohledem na malý vývojový tým – nepodporuje paralelní vývoj více nezávislých modulů bez koordinace architektury.
* Aplikace momentálně neobsahuje automatické testy ani CI/CD pipeline – je nutné manuální testování před nasazením.

# 6 Architektura systému

Aplikace CompanyTrack je postavena jako **webová aplikace s client-server architekturou**. Frontend i backend jsou vytvořeny pomocí JavaScriptového ekosystému s důrazem na modularitu a škálovatelnost.

Backend je postavený na vrstvené archutektiře (controllers, services, repositories, dtos)

* **Frontend**: React / Next.js, TypeScript, ChakraUI v3
* **Backend API**: Nest.js, TypeScript
* **Databáze**: PostgreSQL (hostovaná přes Supabase)
* **Autentizace**: Supabase Auth (JWT)
* **Data-fetching**: Axios + Orval (OpenAPI generovaný klient)
* **Stavová správa**: React Query

# 7 Přílohy

## 7.1 ukázka openapi

openapi: 3.0.0  
paths:  
 /auth/register:  
 post:  
 operationId: AuthController\_register  
 parameters: []  
 requestBody:  
 required: true  
 content:  
 application/json:  
 schema:  
 $ref: "#/components/schemas/RegisterDto"  
 responses:  
 "201":  
 description: ""  
 summary: Register user via Supabase  
 tags: &a1  
 - Auth

## 7.2 ERD diagram

Obsah obrázku text, diagram, číslo, účtenka

Obsah generovaný pomocí AI může být nesprávný.

# 8 Rozšíření do budoucna

- Export dat do Excelu a PDF

- Automatizované upozornění e-mailem (např. při překročení nákladů)

- Integrace s účetními systémy (např. Pohoda, ABRA)

- Mobilní aplikace pro zaměstnance

- Sledování docházky pomocí QR kódu

- Platební brána pro placení aplikace (ze začátku free2use pro testování)