Software solutions | Codelogix.

Nara infra app

Inhoudsopgave

Overzicht

Deze app is ontwikkeld om de interne processen m.b.t. weekstaten, urenregistratie, beheer van gereedschappen en voertuigen en facturatieproces efficiënter te maken. De app is gebouwd met een focus op schaalbaarheid, zodat deze kan groeien naarmate het aantal gebruikers en de hoeveelheid data toeneemt.

Architectuur

De applicatie volgt een **MVC (Model-View-Controller)** architectuur, wat zorgt voor een duidelijke scheiding tussen de verschillende verantwoordelijkheden:

1. **Model**: Dit vertegenwoordigt de datalaag en bevat alle logica voor interactie met de database. In dit project is PostgreSQL de gekozen database.
2. **View**: De gebruikersinterface is opgebouwd met behulp van standaard HTML5, CSS3 en JavaScript. De app maakt gebruik van ERB (Embedded Ruby) om dynamische inhoud in de views te renderen.
3. **Controller**: De controller verwerkt gebruikersinvoer, haalt de relevante gegevens uit de modellen en zorgt ervoor dat de juiste view wordt weergegeven.

De app maakt gebruik van een **RESTful API** om externe interacties en communicatie tussen verschillende services te ondersteunen. Elke functionaliteit is geïsoleerd in discrete routes en controllers, wat zorgt voor makkelijk onderhoudbare en uitbreidbare code.

Verder zijn er verschillende services en jobs geïmplementeerd om taken op de achtergrond uit te voeren, zoals het verwerken van gegevens en het versturen van notificaties. Deze taken worden uitgevoerd met behulp van ActiveJob en Sidekiq.

Technologieën

Dit project is gebouwd met de volgende technologieën:

- **Ruby on Rails** (backend framework)
- **PostgreSQL** (database)
- **Redis & Sidekiq** (voor achtergrondtaken)
- **React** (voor interacties aan de frontend)
- **HTML5 & CSS3** (voor de basisstructuur en styling)
- **Webpacker** (voor het bundelen van JavaScript en CSS-assets)
- **Rspec & Capybara** (voor testing)

Daarnaast maakt de app gebruik van verschillende gems en libraries voor authenticatie, autorisatie, en API-communicatie.

Installatie

Vereisten

Om deze app lokaal te installeren en te draaien, heb je de volgende software nodig:

- Ruby (versie 3.0.0 of hoger)
- Rails (versie 6.0 of hoger)
- PostgreSQL
- Redis (voor Sidekiq)
- Node.js en Yarn (voor het beheren van JavaScript dependencies)

Stappen

1. Clone de repository

```
git clone [repository-URL]
```

2. Navigeer naar de directory:

```
cd naratech-production
```

3. Installeer de Ruby afhankelijkheden met Bundler:

```
bundle install
```

4. Installeer de JavaScript dependencies met Yarn:

```
yarn install
```

5. Maak de database aan en voer de migraties uit:

```
rails db:create
rails db:migrate
```

6. Start Redis voor achtergrondtaken:

```
redis-server
```

7. Start Sidekiq:

```
bundle exec sidekiq
```

8. Start de Rails server:

```
rails server
```

De applicatie zal beschikbaar zijn op http://localhost:3000

Configuratie

Zorg ervoor dat je de benodigde configuratiebestanden hebt ingesteld, zoals de .env voor omgevingsvariabelen. Je kunt een voorbeeldbestand maken met de volgende variabelen:

```
DATABASE_USERNAME=your_username
DATABASE_PASSWORD=your_password
REDIS_URL=redis://localhost:6379/0
```

Pas deze aan met de juiste waarden voor jouw omgeving.

Gebruik

Gebruikersrollen en toegang

De applicatie maakt gebruik van verschillende gebruikersrollen, zoals:

- **Admin**: Volledige toegang tot alle functionaliteiten.
- **Manager**: Kan gegevens beheren maar heeft beperkte toegang tot systeeminstellingen.
- **Gebruiker**: Heeft toegang tot alleen de noodzakelijke functies voor het uitvoeren van dagelijkse taken.

Deze rollen zijn gedefinieerd in de User-model met behulp van de Devise en Pundit gems voor authenticatie en autorisatie.

Hoofdfunctionaliteiten

1. **Projectbeheer**: Gebruikers kunnen projecten aanmaken, bewerken, en verwijderen.
2. **Weekstaten**: Gebruiker kan per week de gewerkte uren op basis van afgesproken bedragen en kostensoorten verwerken, aanpassen of verwijderen in weken
3. **Urenregistratie**: Gebruiker kan per project en week urenregistratie voor medewerkers verwerken
4. **Beheer van bedrijfsprocessen**: Gebruiker kan eigen functies, medewerkers, opdrachtgevers, opdrachtnemers, apparatuur, gereedschap etc. beheren.
5. **Dashboard**: Gebruiker kan via het dashboard een volledig overzicht inzien
6. **Facturatiemodule**: Gebruiker kan ingekomen en verzonden facturen aanmaken. Facturen zijn voorzien van functionaliteiten, zoals registreren van betaling, crediteren van facturen etc.

Tests

Om de applicatie te testen, gebruik je Rspec. Je kunt de tests uitvoeren met het volgende commando:

```
bundle exec rspec
```

De testomgeving is geconfigureerd met FactoryBot en Capybara voor het eenvoudig schrijven van unit tests en feature tests.