README

Installation

Get started with Docker & Docker Compose (preferred)

tested with docker version 1.13.0 and docker-compose version 1.10.0

1. Install docker and docker compose - https://docs.docker.com/compose/install/
2. Run the services:

   ```
   cd Code
   docker-compose up -d
   docker-compose exec backend rails db:setup
   ```

3. Open http://localhost:4000
4. Use one of the following email, users have same password: `password`

   ```
   admin@example.com
   agent1@example.com
   customer1@example.com
   customer2@example.com
   customer3@example.com
   ```

5. For development

   ```
   cd Code
   docker-compose -f docker-compose.dev.yml up -d # run in development env.
   docker-compose exec backend rails db:setup
   docker-compose exec backend rails spec spinach
   ```

Install a development environment

Backend

Ruby version

- 2.3.1

System dependencies

- Mysql version 5.7
- Elasticsearch version 2.4

Configuration

- Mysql
  - use `root` and `empty password` or override in `config/database.yml`
  - in `production` environment set env variable `DATABASE_URL="mysql2://myuser:mypass@localhost/somedatabase"`
- Elasticsearch
  - should be run on `localhost:9200`
  - in `production` environment set env variable `ES_HOST="host:port"`

Prepare and run web server

1. Setup database

   ```
   cd Code/backend
   rails db:environment:set RAILS_ENV=development
   rails db:setup # creates database, load schema and seed data
   ```

2. Install gems

   ```
   bundle install
   ```

3. Run test suite (must pass)

   ```
   rails spec # Unit tests
   rails spinach # Feature tests
   ```

4. Run server on port 3000

```
rails s
```

Frontend

Node.js version

- 7.4

Configuration

1. Install packages

   ```
   cd Code/frontend
   npm install
   ```

2. Run lint and test suite (must pass)

   ```
   npm run deploy:dev
   ```

3. Run dev server

   ```
   npm run dev
   ```

4. Open the app - http://localhost:4000

5. For development

   ```
   cd Code
   docker-compose -f docker-compose.dev.yml up -d # run in development env
   docker-compose exec frontend npm run test
   docker-compose exec frontend npm run lint
   ```

Test Coverage

Backend

- unit tests (rspec) - 30.29%
- integration tests (spinach) - 96.83%

Frontend

- unit tests
  ```
  Statements   : 48.32% ( 274/567 )
  Branches     : 36.41% ( 79/217 )
  Functions    : 34.23% ( 89/260 )
  Lines        : 50.48% ( 265/525 )
  ```