

## Contents

Getting Started.....	1
Initiate Application.....	1
Create Database.....	2
API Endpoints.....	2
Register .....	2
Login.....	2
Logout .....	3
Create Survey.....	3
Fetch Survey.....	4
Answer Question.....	4

## Getting Started

### Initiate Application

Running the application is simple, and can be done from the command line. The application is built with NodeJS, so that will be required, and I have opted to use the `yarn` package manager.

Clone this github repository to your local machine. The structure of the application is split into `src` and `server` directories, for the frontend and backend respectively. Both modules of the app need to be initialized.

From the root of the application, run the following commands:

```
yarn install
yarn start
cd server
yarn install
yarn start
```

Now you're good to go!

## Create Database

This application also uses Postgresql as the relational database on the backend. Once Postgresql is installed on your local machine, please follow the steps below in order to create the database and tables required. The `scripts.sql` file can be found in the `server/` directory.

```
su – postgres  
  
psql  
  
(copy and paste the code from scripts.sql and press return)
```

Please also ensure the credentials for accessing the database are correct. The file containing these credentials is located in `server/db/connect.js`. The application will attempt to connect to the database at `localhost` on `port 5432`, which is as standard for Postgresql. If the password for user `postgres` in `connect.js` is different from your own please change this.

## API Endpoints

### Register

**Purpose:** Registration of a new survey publisher account

**Body format:** JSON

**Body example:** { "first\_name": "John",  
                  "last\_name": "Smith",  
                  "email": "john.smith@test.com",  
                  "password": "password1234" }

**Definition:** POST `http://localhost:5000/signup`

**Success Response:** 201 Created

**Failure Response:** 400 Bad Request

### Login

**Purpose:** Login with a previously registered survey publisher account

**Body format:** JSON

**Body example:** { "email": "john.smith@test.com",  
                  "password": "password1234" }

**Definition:** POST `http://localhost:5000/signin`

**Success Response:** 200 OK

**Success Response body:** {"publisher\_id", "email", "token"}

**Failure Response:** 400 Bad Request

**Failure Response body:** {"signin\_error": "Email/password does not match"}

This failure response will return in the event that the password does not match an existing email, or the email does not exist in the accounts database.

## Logout

**Purpose:** Logout of an account - if a survey publisher is currently logged in

**Body format:** JSON

**Definition:** POST http://localhost:5000/logout

**Success Response body:** {"first\_name": "John",  
"last\_name": "Smith",  
"email": "john.smith@test.com",  
"token": "random token"}

**Failure Response:** 400 Bad Request

## Create Survey

**Purpose:** Using a registered survey publisher account, add a survey and questions to the database.

**Body format:** JSON

**Body example:** {"survey\_title": "My Survey",  
"question": "What is your name?",  
"answer": "John",  
"email": "john.smith@test.com"}

**Definition:** POST http://localhost:5000/create\_survey

**Success Response:** 201 Created

**Failure Response:** 400 Bad Request

This endpoint will add an answer option to a question, a question to a survey, and a survey to the database if it does not already exist – referenced by survey title and the logged in account that is sending the request.

## Fetch Survey

**Purpose:** Send a request to retrieve the question and available answer of a survey that is in storage.

**Body format:** JSON

**Body example:** { "survey\_title": "My Survey" }

**Definition:** POST http://localhost:5000/fetch\_survey

**Success Response:** 200 OK

**Success Response body:** { "survey\_title": "My Survey",  
    "questions":  
        { "question": "What is your name?",  
          "answer": "John" },  
    ... }

**Failure Response:** 400 Bad Request

This endpoint will retrieve the available answers and questions associated with a survey based on the survey title. If the survey title does not exist, no response will be received.

## Answer Question

**Purpose:** Allows a user to select an answer to a question of a specific survey.

**Body format:** JSON

**Body example:** { "survey\_title": "My Survey",  
    "question": "What is your name?",  
    "answer": "John" }

**Definition:** POST http://localhost:5000/answer\_question

**Success Response:** 200 OK

**Success Response body:** { "survey\_title": "My Survey",  
    "questions":  
        { "What is your name": "John", },  
    ... }

**Failure Response:** 400 Bad Request