Averox leads

![averox-leads-logo.png](https://i.imgur.com/IDqAbvI.png)

<br/>

Repository to hold our Averox Leads project source code.

We have a detailed tech spec doc ready at

[Confluence BIZGGRO - Averox Leads Technicial specifications](https://averox.atlassian.net/wiki/spaces/BIZGGRO/pages/10518535/Technical+Specifications)

Mainly our Application is composed of two main part.

- Frontend

- Backed

Application staging live at: https://averox-leads.herokuapp.com/

Our Frontend will reside here on the `src`, meanwhile we have a sperate folder for server/backend `/server` and it has its own `README.md`

### Starting App:

##### Installing Composer

- Download Docker [https://docs.docker.com/]

- Use Command 'docker ps' to check if services are running.

- 'docker-compose up' start services.

#### Starting Backend

Our Backed Core dependent on Docker, so make sure we have `docker-compose` installed.

Our Database is `MongoDB` with instance in a container running on `27017` with `prisma` layer running at `http://localhost:4466`.

Prisma is a 3rd party service we are using for GRAPHQL friendly subscription structure. Its an ORM.

- goto `server` directory via `cd server` command.

- start app via `docker-compose up --build`

#### Starting Frontend

- Make sure you are in `averox-leads/src`

- start app via `npm start`

# Setting up project on local machine

## Installations

Make sure you have installed the following applications.

- ![git-bash-logo.png](https://i.imgur.com/PCo3bOJ.png) Git bash [Download here](https://github.com/git-for-windows/git/releases/download/v2.27.0.windows.1/Git-2.27.0-64-bit.exe)

- ![docker-logo.png](https://i.imgur.com/XvOC96e.png) Docker [Download here](https://download.docker.com/win/stable/Docker%20Desktop%20Installer.exe)

- > Make sure running using `Linux containers`, not windows

- ![node-js-logo.png](https://i.imgur.com/c1KM9Es.png) Node JS [Download here](https://nodejs.org/dist/v12.18.2/node-v12.18.2-x64.msi)

- ![fiddler-classic-logo.png](https://i.imgur.com/oD3h1vY.png) Fiddler [Download here](https://telerik-fiddler.s3.amazonaws.com/fiddler/FiddlerSetup.exe)

- ![visual-studio-code-logo.png](https://i.imgur.com/2RHuxtI.png) Visual Studio Code [Download here](https://aka.ms/win32-x64-user-stable)

## Generate SSH Keys and Setup with Github

Add SSH keys in github account.

1. Open Git bash (command line) in your computer and run following command and keep pressing `Enter/OK` key until the setup questions completed.

> `ssh-keygen`

2. Go to SSH directory exists in user directory (Windows: `C:\Users\Username\.ssh`) and edit `id\_rsa.pub` in text editor.

3. Copy contents of `id\_rsa.pub` which are something like shown below.

> ssh-rsa AAAAB3N...

4. Login and Go to [Github SSH keys page](https://github.com/settings/keys). Click `New SSH Key` button and paste content already copied. Title is optional field.

\*\*NOTE:\*\*

if you face some unexpected behavior try [Generate SSH keys](https://docs.github.com/articles/generating-an-ssh-key/) and [Common SSH Problems](https://docs.github.com/ssh-issues/)

5. Add Global Github credentials.

- Set Username

> `git config --global user.name="User Name"`

- Set Email

> `git config --global user.email="you@email.com"`

## Source cloning

Source cloning with following command ([Repo Link](https://github.com/averox-solutions/averox-leads))

> `git clone https://github.com/averox-solutions/averox-leads.git`

### Commiting your changes:

We are using [stanardjs](https://standardjs.com/) for our linting and [prettier](https://prettier.io/) for code formating.

There's a pre-commit hook which will prevent you from pushing your code on repository unless you run the fixes.

- To Check Linting Errors use

> `npm run linting`

- To Fix Linting Errors use

> `npm run linting:fix`

- To Check Formatting errors use

> `npm run prettier`

- To Fix Farmatting errors use

> `npm run prettier:fix`

## Configurations

### Installing Dependencies

1. Install front end dependencies.

> `npm install` (assuming you're in root directory i.e. `averox-leads`)

2. Install backend dependencies.

1. Go to server directory.

> `cd server`

2. Install packages.

> `npm install`

### Environment Files Format

1. `.env` file in `root directory` has following format/keys.

###

NODE\_PATH=src/

CLIENT\_PORT=5000

PORT=5000

HTTPS=true

KEY='./ssl/private.key'

CERT='./ssl/certificate.crt'

CA='./ssl/ca\_bundle.crt'

SSL\_CRT\_FILE='./ssl/certificate.crt'

SSL\_KEY\_FILE='./ssl/private.key'

REACT\_APP\_USE\_LEADS\_FB\_APP=1

REACT\_APP\_USE\_BLEUPAGE\_FB\_APP\_FOR\_SEARCH=0

# FB App: Averox Leads

REACT\_APP\_LEADS\_FB\_APP\_ID=

REACT\_APP\_LEADS\_FB\_APP\_SECRET=

REACT\_APP\_LEADS\_FB\_API\_VERSION=

# FB App: Bleupage

REACT\_APP\_BLEUPAGE\_FB\_APP\_ID=

REACT\_APP\_BLEUPAGE\_FB\_APP\_SECRET=

REACT\_APP\_BLEUPAGE\_FB\_API\_VERSION=

REACT\_APP\_IS\_LOCAL=1

REACT\_APP\_BACKEND\_URL='https://lead.averox.com:4000'

REACT\_APP\_AMEMBER\_URL='https://leads-member.averox.com/api/'

REACT\_APP\_AMEMBER\_API\_KEY=''

REACT\_APP\_S3\_ACCESS\_KEY=''

REACT\_APP\_S3\_SECRET\_KEY=''

REACT\_APP\_S3\_BUCKET\_NAME='averox-leads'

2. `.env` file in `server directory` has following format/keys.

###

SERVER\_PORT=4000

KEY='../ssl/private.key'

CERT='../ssl/certificate.crt'

CA='../ssl/ca\_bundle.crt'

TWILIO\_ACCOUNT\_ID=''

TWILIO\_AUTH\_TOKEN=''

TWILIO\_NUMBER='+12345678900'

MAIL\_SMTP="smtp.yourserver.com"

MAIL\_PORT=587

MAIL\_AUTH\_USER="useremail@yourserver.com"

MAIL\_AUTH\_PASS=""

MAIL\_FROM="fromemail@yourserver.com"

### Setting Database

1. Make sure `docker` is running with `Linux containers`.

2. Go to `server/prisma` directory using following command.

> `cd server/prisma`

3. Setting environment variable.

- \*\*Windows OS Users\*\*

Open Command Prompt in this directory. Run the file `prisma-auth-windows.cmd` using following command.

> `prisma-auth-windows.cmd`

\_OR\_

Copy and paste the command from file and paste in command prompt.

- \*\*Linux OS Users\*\*

Run the file `prisma-auth.shell` using following command.

> `./prisma-auth.shell`

OR

Copy and paste the command from file and paste in terminal.

4. Build App and Install the docker services.

Run the following command.

- Go to `server/prisma` directory with command.

> `cd server/prisma`

- install and run the services.

> `docker-compose up --build`

We have `nodemon` attached, it will detech future updates and restart the server if needed.

###

NOTE:

This would take a while to download and run the docker services.

### Fiddler Settings

1. Run the fiddler.

2. Open \*\*HOSTS\*\* file from \*\*\_Tools menu => HOSTS\_\*\*.

3. Check the checkbox "Enable remapping of requests..."

4. Paste the following content in text field.

> localhost lead.averox.com

5. Now you can access your local project with following URL.

> [https://lead.averox.com:5000](https://lead.averox.com:5000)

###

Next time you just need to run the fiddler and minimize. This will be resolving your project URL (https://lead.averox.com:5000) to https://localhost:5000.

## Accessing GraphQL Playground

- You can access the GraphQL playground to test/run your queries or mutations to interact with database.

- This provides a better view of available queries and mutations.

- To use this playground, you must provide the `Authorization` token. to get this token you need to run the login query.

1. Run the following query to login.

###

mutation login {

login(email:"emailusername@mailprovider.com", password:"password") {

token,

user {

name,

email

}

}

}

2. Paste the following code in "HTTP HEADERS" window.

###

{

"Authorization": "Bearer YOUR-TOKEN-HERE"

}

3. Copy the token returned by above login mutation and replace in HTTP HEADERS window. Replace `YOUR-TOKEN-HERE` with the real token, don't change anything else.

4. Now, test run the following query.

###

query myinfo {

me {

name,

email,

}

}

## Accessing Prisma Admin Dashboard

You can access the Prisma Admin Dashboard from this [LINK](http://localhost:4466/\_admin).

### Add Prisma token

1. Open command prompt in `server/prisma` directory and run the following command.

> `prisma token`

2. copy the generated token.

3. In prisma admin screen, click settings icon.

4. paste the copied token in "secret token" field and save.

<img align="left" style="padding:0;margin:0;" width="24" height="24" src="https://i.ibb.co/kDYcQ1q/docker-logo.png">

# Local Configurations and Updates

## Update local database when Schema changed

Make sure that you have pulled the latest code in which the files with updated schema has included. In our case `datamodel.prisma` and `schema.graphql` reside in `server/prisma/` and `server/src/` directories.

Follow the steps, in provided sequence, to update local database schama.

1. `git pull` (pull the latest code with schema changed)

2. `cd server/prisma` (when you're commandline cursor stands in `averox-leads/`)

3. Double click the file `prisma-auth.shell` (when Linux/OSX) or `prisma-auth-windows.cmd` (when windows environment) OR copy and paste the command from these files and paste in terminal/command prompt respectively.

4. `prisma deploy` (deploy new schema to database)

5. `npm start` (restart your serer to use updated database schema)

# Help with GraphQL Server:

- goto `server` directory via `cd server` command.

- start app via `npm start`

# Help with React Application:

This project was bootstrapped with [Create React App](https://github.com/facebook/create-react-app).

## Available Scripts

In the project directory, you can run:

### `npm start`

Runs the app in the development mode.<br />

Open [http://localhost:3000](http://localhost:3000) to view it in the browser.

The page will reload if you make edits.<br />

You will also see any lint errors in the console.

### `npm test`

Launches the test runner in the interactive watch mode.<br />

See the section about [running tests](https://facebook.github.io/create-react-app/docs/running-tests) for more information.

### `npm run build`

Builds the app for production to the `build` folder.<br />

It correctly bundles React in production mode and optimizes the build for the best performance.

The build is minified and the filenames include the hashes.<br />

Your app is ready to be deployed!

See the section about [deployment](https://facebook.github.io/create-react-app/docs/deployment) for more information.

### `npm run build` fails to minify

This section has moved here: https://facebook.github.io/create-react-app/docs/troubleshooting#npm-run-build-fails-to-minify