Texas Homes Project Technical Report

Jeronimo Del Valle Ocejo | Gabriel Casanova | Shahmir Masood | Ananth Kothuri

Purpose/Motivation:

We're on a mission to make it easier for people experiencing homelessness to receive the help they need. Texas Homes Project is envisioned to be an online resource for anyone looking to support the Texas homeless population. We aim to connect communities with nearby homeless shelters, support organizations that aid the homeless, and spread the word about upcoming volunteer opportunities.

User Stories:

Stories that we implemented:

- Add a picture of a happy homeless person
    - We simply added the requested image on the home page
- Add an image that conveys what the website is about to the home page - We added several images to our page in a slideshow
- Add a brief overview of what viewers will be getting when visiting your site - We added extra intro text and included our mission statement on the home page - Add coordinates on shelter models
    - We added this info for each instance of shelter
- Add a slide deck of pictures of people in the homeless shelters on the landing page - We completed this issue and the 2nd issue with a slide show on our splash page

Stories that we gave:

- Use Bootstrap, Tailwind, or another framework
- Include a photo and bio for each member of your team in the “About” page - Add images for about page and splash page
- Add a description for Foster Hope on the splash page
- Add models on the organization page

Challenges:

Phase 1:

● For all of us, this project was our introduction to the world of web development. With the team’s limited experience as a whole, it felt a bit daunting to find where to even start. On top of that, one of our team members ended up dropping the course, leaving only 4 people remaining to complete the project, rather than the typical 5. Despite these

setbacks, we made sure to start early and utilized resources including YouTube videos and Ed Discussion to answer any questions we had along the way. By starting early, we were able to split the workload into more manageable chunks where each person had a reasonable goal that wasn't too overwhelming to handle. Additionally, we made an effort to meet in person to collaboratively code and help each other debug.

Hosting:

We used AWS Amplify for website deployment and hosting.

API Documentation

Our API documentation involves `GET` requests to poll information about our different shelters, counties, and events. These requests include retrieving all model instances, or specific model instances by ID. For more details about these requests, please refer to our Postman documentation: [Postman Documentation]

Models & Instances:

Currently, to add more model data or instances to the website, this data should be stored within one of the model `JSON` files in the `front-end/src/data` folder. The data must follow the following format for it to work properly:

- **Shelters:**
  - `id`: internal id
  - `name`: name of shelter
  - `address`: address of shelter
  - `city`: city where it's located
  - `state`: state where it's located
  - `zip_code`: zip code where it's located
  - `location`: coordinates of latitude and longitude
  - `phone_number`: the phone number of the shelter
  - `email_address`: the contact email address
  - `fax_number`: the fax number of the shelter
  - `official_website`: the website of this shelter
  - `twitter`: the twitter link for this shelter
  - `facebook`: the facebook link for this shelter
  - `instagram`: the instagram link for this shelter
  - `description`: a textual description of this shelter and what it's goals/purpose is

- `photo_urls`: a list of photos associated with this shelter
- `update_datetime`: the last time this information was updated
- `video_url`: an optional video url for this shelter
- `related_models`: a list of related shelters, events, or counties and their ids.

- **Counties**
  - `id`: internal id
  - `name`: name of this county
  - `population`: the population of this county
  - `housing`: the number of housing units in this county
  - `website_url`: the wikipedia url that describes this county
  - `description`: a textual description and summary of this county
  - `image_url`: an image url of this county
  - `map`: a possible map image of this county
  - `images`: an additional list of images for this county
  - `related_models`: a list of related shelters, events, or counties and their ids.

- **Events**
  - `id`: internal id
  - `title`: the title of this event
  - `organization`: the name of the organization running this event
  - `image`: image url for this event
  - `description`: a textual description of this event and what it consists of
  - `date_posted`: the date this event was posted
  - `time`: the time this event will occur
  - `cause_areas`: what causes this event hopes to help with
  - `location`: the location of this event
  - `skills`: skills that are sought after for this event
  - `requirements`: requirements for people participating in this event ○ `related_models`: a list of related shelters, events, or counties and their ids.

Frontend Architecture:

The front-end of our website is a React.js project hosted on AWS Amplify. We use numerous Bootstrap components to ensure our website dynamically scales to all screen sizes. We also have a dynamic splash page (Home) and About page, as well as pages for Shelters, Events, and Counties models and instances.

Installation

1. Clone the repo and ensure npm is installed
2. Run `npm install` to install all dependencies
3. Run `./install_dependencies.sh` within the `/front-end` folder
4. Run `npm start` to start the website on your local server

File System

Our front-end file system is located within the `front-end/src` folder. This folder contains multiple subfolders, each corresponding to a different part of the website:

- **Main Tabs**: Each tab in our website has its own folder for its content
  - **Home**: `/front-end/src/Home/HomePage.jsx`
  - **About**: `/front-end/src/About/AboutPage.jsx`
  - **Shelters**: `/front-end/src/Shelters/SheltersPage.jsx`
  - **Counties**: `/front-end/src/Counties/CountiesPage.jsx`
  - **Events**: `/front-end/src/Events/EventsPage.jsx`

- **Model Templates**: These are components that we reuse across all models. This allows the webpage to change dynamically based on what type of model is being used. ● **PageLayout**: `/front-end/src/ModelTemplates/PageLayout.jsx` - Creates a grid of models to be searched and filtered through
  - **InstanceCard**: `/front-end/src/ModelTemplates/InstanceCard.jsx` - A single instance card with a few attributes and features
  - **InstancePage**: `/front-end/src/ModelTemplates/InstancePage.jsx` - A page for a model instance with information and related models

- **Assets**: All of our assets are stored in the `front-end/src/assets` folder and include local images, member photos, and tool icons.

- **Components**: Other components that were used, such as the `NavBar`, are stored in the `front-end/src/components` folder.

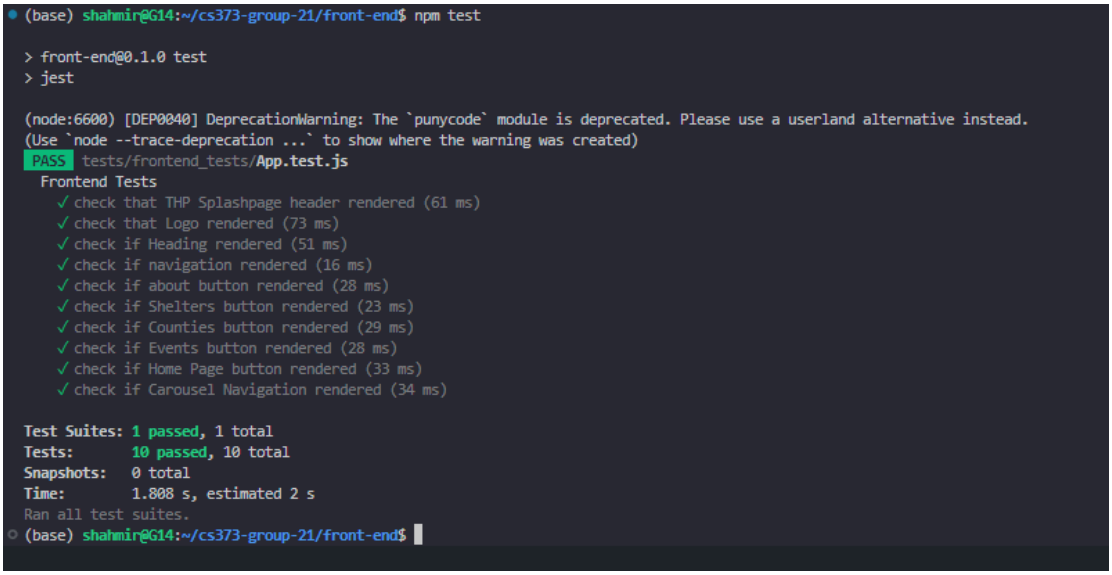
- **Static Metadata**: The `front-end/src/data` folder contains all our static metadata for the models, as well as information that is used to populate the About page.

Frontend Testing

The frontend testing using jest can be run by **`npm test`** in the <root dir>/front-end/ directory. You will likely need to run **`npm install`** `to` ensure you have the proper dependencies and installations. The tests are located in `/front-end/tests/App.test.js`

A description of what each test is doing is listed in the output.

Expected output:



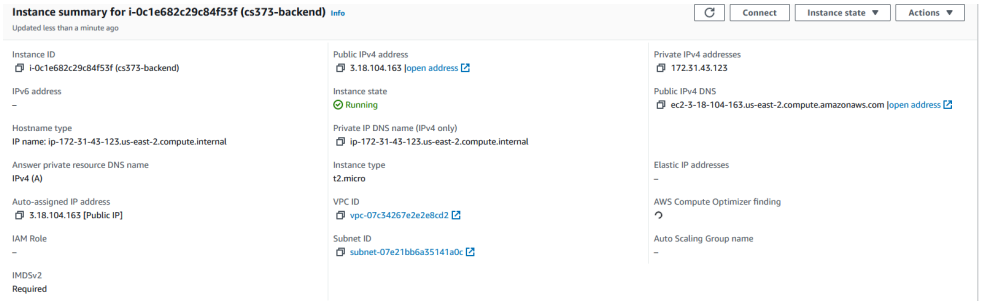
Backend Architecture:

To set up the backend hosting, we followed the tutorial here.

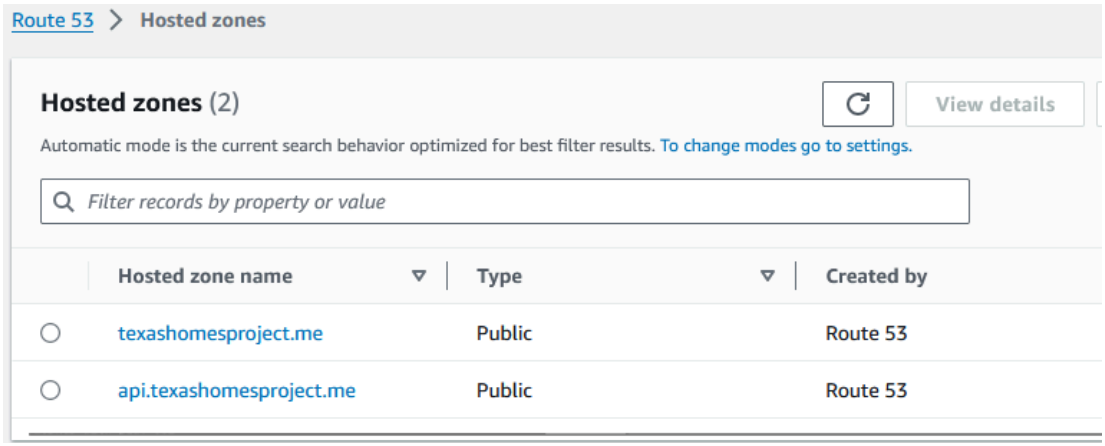
It is entirely set up on Amazon web services, and the API backend website can be found here: http://api.texashomesproject.me/

In particular, here is the structure of our server.

- **EC2 instance**: named cs373-backend
    - On the instance we have a trivial flask app and our docker installation. We currently have it set up to use the gunicorn flask production server.

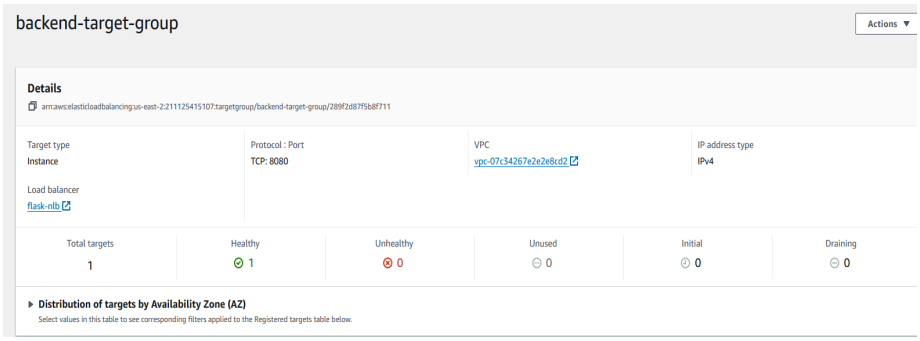


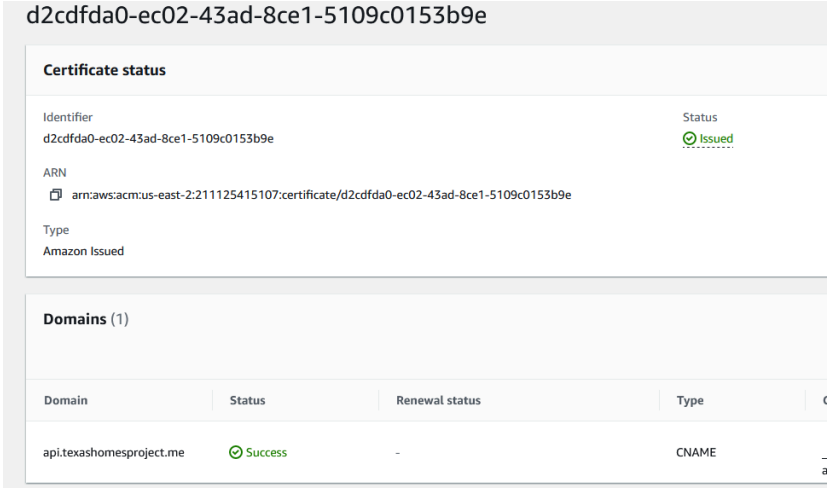
- **Route 53 Domain:** named api.texashomesproject.me
    - This hosted zone is in route 53 by copying CNAME record from Namecheap domain



    - **Network Load balancer:** named flask-nlb
        - The nlb distributes incoming traffic from our domain to our specified target group

        - **Target group:** named backend-target-group

- 

- **ACM certificate**: named d2cdfda0-ec02-43ad-8ce1-5109c0153b9e
- 

Tools Used:

These are the tools we used for phase 1 of our website:

- **`AWS Amplify`** - used to deploy our website
- **`Docker`** - used when running the GitLab pipeline
- **`GitLab`** - used for version control and collaboration among members
- **`Postman`** - used to document our future expected API behavior
- **`React`** - chosen as the frontend library for developing our interactive UI
- **`Bootstrap`** - used for uniform and visually appealing UI components