NM Forest Industry Map and Data Collection

Website README and Moving Forward Doc

Tri Blankley, Jena Peterson
Contributors: Dominic DOnofrio
May 13th, 2025

README

Spinning up the Website:

- Creating Landowner and Industry Survey Databases
    - Check Database-Info file for SQL code and business data.
        - Use create statements to create the tables.
        - Use the csv documents to import business data and type of work.
- Create the .env file to be able to connect to the database
    - This is where the information to connect to the database is kept. If the database is not being run on a Microsoft MySQL Azure server, the connection information within server.js may need to change.

.env file should be as follows

```
Unset
Industry / Business Database
busDB_HOST=(server name)
busDB_USER=(username)
busDB_PASSWORD=(password)
busDB_NAME=(database name)

Landowner Database
landDB_HOST=(server name)
landDB_USER=(username)
landDB_PASSWORD=(password)
landDB_NAME=(database name)
```

- Spin an instance of the website with Docker. These commands may require super-user access to execute depending on your system's access settings.

    The docker file will run the node.js server connecting to the Microsoft database and the npm server providing the locally hosted instance of the website. When

you add or remove npm packages from the website, you may have to rebuild the docker container to see those changes reflected.

- Install docker on your system, if on linux distributions that use apt use
  $ apt-get install docker
- Build the docker environment with
  $ docker compose build
- Run the docker environment with:
  $ docker compose up

Moving Forward:

Perform More Rigorous User Testing with a Wide Variety of People and Use-cases

Get at least 30 users to test the website, ensuring you have a wide range of ages. If possible, get people where English is their second language to see if the survey pages are still straightforward despite a language barrier. When testing, take special note of how people interact with the map and examine how they prefer to access industry data.

Add Cybersecurity Considerations to the Landowner Database

Currently, the cybersecurity of the website and databases is entirely handled via ensuring the website is as simple as possible, and utilizing the preexisting cybersecurity infrastructure included with Microsoft Azure cloud systems. If the forestry division decides to self-host the website and databases, ensure passwords are strong, the server is behind comprehensive firewalls, and sensitive information is encrypted. If hosting via Microsoft Azure, build a more robust input cleaning system, and ensure the landowner database can't post to the website.

Implement Pen Testing and Fix Issues It Finds

Pen testing will be used to find vulnerabilities on the website. The '*Testing options before deploying the Docker containers'* section will explain this in more detail.

Change How PDFs Are Stored

Currently, the provided PDFs are converted into binary data and stored directly in the database. This is only a short-term solution to show proof of concept. This way of storage makes packets sent to the database incredibly large and can take out quite a bit of space in the database. In addition, this does not check if the PDFs have viruses or not. The best practice would be to store the PDFs in a separate database or file system and have a virus scanner scan the PDFs.

Host the website to make it Public-Facing:

- This can either be carried out by using Microsoft Azure Cloud Systems, or the website can be self-hosted by using tools like NginX for HTTPS certification, where NginX would replace node.js as the interface with the SQL tables
    - Ensure any mention of localhost is changed to match where it is being hosted.
- Note, when making the website public-facing, ensure that you build a deployment version of the website. This will compress the website, making it lighter for clients while removing the built-in Vue dev tools

Implement a Chat System, Taking in Private Landowner Information and Automatically Formatting it into a Short Email that is Sent to a Business:

The actual email formatting is rather easy, and can be carried out with a variety of python, or node.js packages. However, this system will also require access to a forestry email server, from which it would need the permissions to send and receive emails.

Depending on implementation, these chats would then entirely be handled via personal email, or, the website could display these emails in a chat box stream, while making carbon copy's of the correspondence with the use of email replies. Making an in-site chat box would necessitate the landowners and industry leaders to create an account with the website as well.

Ability for Landowners to Create "Advertisements" to Notify Businesses of Available Resources:

This system would use the work-radius column of the industry database, and compare it to landowners that fall within this range, if that landowner has consented to receive emails from the NM forestry division when they added their information via the Landowner Survey, they would reside an email from that business, advertising services or requesting resources. This would also require access to a NM forestry email server, similar to the chat system functionality.
In-Site chat functionality to connect Landowners to Business owners

Account Creation

Depending on the future growth and use of this website, it may be appropriate to have each landowner and industry leader create an account before accessing certain tools, like chat or advertisement functionality. This would require several new .vue pages to be created, to handle username and password login handling, a username and password value would have to be added to the database, where each plot of land would be tied to an account. These passwords should be encrypted, and all other cybersecurity considerations should be met to implement this function.

LinkedIn-style Website Integration:

Not all landowners have a personal website. If future students are able to create a LinkedIn-type website for the industry leaders, link that site to their entity in the business list in order to show photographs and give more information on each industry leader.
Create a Spanish mirror of the website, or Spanish Mode, updating all important text,and place special focus on the survey pages.

Testing options before deploying the Docker containers:

Testing Before Deployment

Before deploying Dockerized containers to production, it is essential to conduct thorough testing to ensure that the application is functional, secure, and resilient. Docker's isolated environments facilitate replicating real-world deployment conditions, which are crucial for reliable testing. Here are several key tools and methods used for pre-deployment testing:

OWASP ZAP (Zed Attack Proxy)

Purpose and Use Case:
OWASP ZAP is an open-source security scanner that helps identify vulnerabilities in web applications. It is especially valuable for automated security assessments, addressing common threats such as Cross-Site Scripting (XSS), SQL Injection, and various misconfigurations. Incorporating ZAP into your container testing workflow ensures the application meets modern security standards before deployment.
Benefits:

- Automatically scans for a wide range of vulnerabilities.
- Supports scripting and automation, which integrates nicely into CI/CD pipelines.
- It can be run headless inside a container or connected via a shared Docker network to test internal services.

Burp Suite

Purpose and Use Case:
Burp Suite is a powerful manual testing framework used by security professionals to test web applications. It allows for thorough inspection of traffic between the browser and the server, enabling developers to manipulate requests, explore authentication flows, and test APIs with precision and control.
Benefits:

- Provides real-time insight into how data flows through the application.
- It can identify logic flaws, authentication issues, and other non-obvious vulnerabilities.
- Includes tools like the Repeater, Intruder, and Sequencer for advanced testing scenarios.

Postman and Newman

Purpose and Use Case:
Postman is a widely used tool for manually testing REST APIs. Its command-line companion, Newman, enables the automation of these tests and allows for integration with Docker or CI/CD workflows. Both tools are essential for verifying API behavior, handling edge cases, and managing error responses.
Benefits:

- Encourages test-driven API development.

- Validates API contracts and ensures consistency across environments.
- Easily scriptable and exportable for team collaboration.

Cypress

Purpose and Use Case:
Cypress is an advanced end-to-end testing framework for browser environments optimized for real-time interaction testing. It is especially beneficial for front-end applications developed using frameworks like React, Vue, or Vite. Cypress can be added like any other npm package.
Benefits:

- Tests user flows (e.g., login, navigation, form input).
- Integrates with Docker for consistent testing environments.
- Supports headless and interactive modes for development and CI.

Backend Testing with Jest and Supertest

Purpose and Use Case:
Jest and Supertest are often used together for unit and integration testing in Node.js environments. These tools help validate internal logic, API routes, and middleware, ensuring the back end performs correctly under various scenarios.
Benefits:

- Allows automated regression testing of server logic.
- Ensures stability during code changes and refactors.
- Easy to configure in Docker as part of a build or test stage.