Guaranty
SHEET METAL
& ROOFING
Since 1970

SPRINT 1

Brandon Bejarano, Jacob Carney,
Waleed Kambal, Mason
Wittkofski, Michal Zajac,

Project Focus

- Our project is focused on designing an inventory management system for Guaranty Sheet Metal based in New Orleans, LA.
- We're looking to build them a complex web application that they can use to track their current inventory, create and edit incoming and outgoing shipments for jobs and supplies received.
- We're also looking into creating a job invoice estimation tool to allow the input of inventory used on a job site to create a cost for the materials of the job.

Sprint Goal

- The main goal was to create a simple full-stack application that can persist data such as user login information, but for Sprint 1, these were our main goals:
  - Allow a user to sign in from the client side
  - Have a simple working end-to-end application that persists data.
  - Allow users to browse the current inventory and sort based on different attributes
  - Get a dashboard setup for analytics purposes
  - Get database setup on AWS
  - Get an API setup to work with the frontend and backend

Major Sprint Achievements

- Postgres on Docker
- Postgres on docker on AWS
- Api that allows for transfer of login info from front end and database, both ways
- Database Schema was made
- Setup react project structure, and made a basic UI
- Implemented login page and authentication mechanism
- Integrated the web app to the rest API

Sprint 1 Backlog

Sprint backlog:

Presentation

- [x] ~~Create presentation for Sprint Deliverable~~

Frontend Development

- [x] ~~Set up a basic project structure for the frontend (HTML, CSS, JavaScript).~~
- [x] ~~Create a landing page layout using HTML and CSS.~~
- [x] ~~Implement basic navigation using JavaScript.~~
- [x] ~~Implement Login page with Logic~~
- [x] ~~Start dashboard page with analytics. Just outline with analytic ideas~~

Backend Development

- [x] ~~Set up a basic backend framework (if applicable, based on your tech stack, e.g., Node.js/Express).~~
- [x] ~~Connect the backend with the PostgreSQL database on AWS.~~
- [ ] Implement basic CRUD (Create, Read, Update, Delete) operations to interact with the database.
- [x] ~~Connect API to Frontend to do user authentication~~
- [x] ~~Connect API to Database~~
- [ ] Enter test data in Database

Docker and AWS Integration

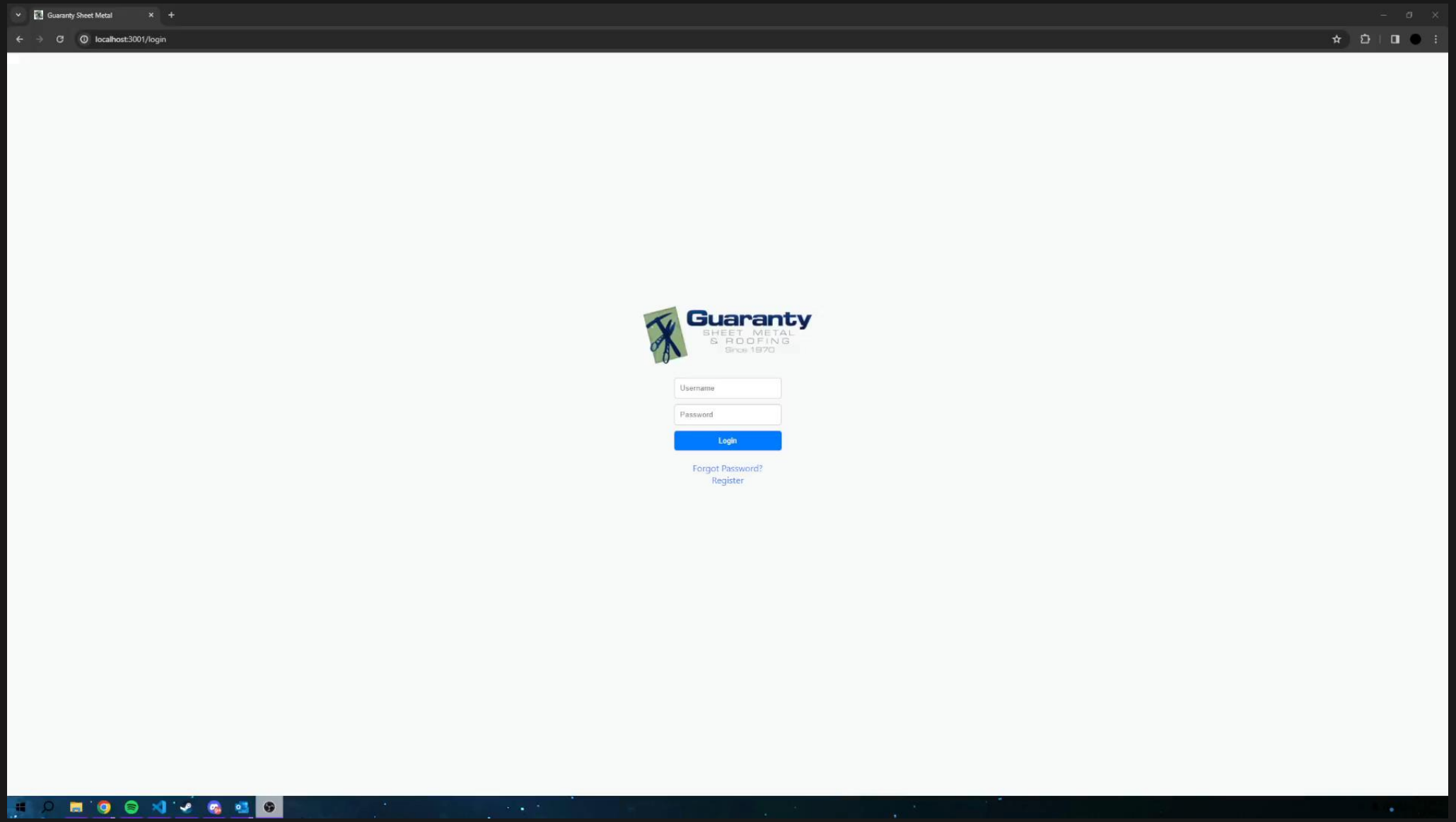
- [x] ~~Containerize the backend with Docker.~~
- [x] ~~Deploy the Docker container on AWS (using Amazon ECS or another suitable service).~~
- [x] ~~Hosting on AWS~~
- [ ] Deploy the frontend on AWS (consider using Amazon S3 and CloudFront for static site hosting).

Project Backlog

ID	Story	Estimation (hours)	Priority (1-5)	Sprint When Finished
1	Set up a PostgreSQL database on AWS using Docker	3	3	1
2	Design and implement the database schema for inventory	4	4	1
3	Populate the database with sample inventory data	2	1	
4	Set up a Node.js/Express.js project structure	3	1	1
5	Implement user authentication in the API	5	5	1
6	Create API endpoints for inventory management	8	1	
7	Set up a React project structure for the front-end	3	1	1
8	Implement login and registration pages in React	5	4	1
9	Connect the React front-end to the Node.js API for authentication	4	2	1
10	Design the inventory management UI in React	6	2	
11	Implement inventory listing in the React app	4	2	
12	Implement adding new inventory items in the React app	5	2	
13	Implement inventory warnings and notifications	4	3	
14	Deploy the Node.js API on AWS using Docker	4	1	
15	Deploy the React front-end on AWS using Docker	4	1	
16	Figure out best way to convert invoice info	2	1	1
17	Compute average based on past invoices	3	1	
18	Design the inventory management page	5	3	
19	Design the dashboard for the app	5	2	
20	Design the invoice management page	5	3	
21	Scanner for the invoices to automatically insert data	5	2	
22	Update Database Schema	2	5	1
23	Create API endpoints for login	8	5	1
24	Collapsible sidebar on all pages	4	3	1
25	Logout setup	3	4	1
26	Set up a setting page	5	2	
27				
28				
29				

What we didn't complete/ lessons learned

- We were not able to implement and endpoint on the web application that would allow the users to view the current inventory and sort it based on different attributes
- Additionally, we were not able to include the analytics on the dashboard
- Uploading an API and front end to docker, is not a fun task.


Guaranty Sheet Metal
localhost:3001/login
Guaranty
SHEET METAL
& ROOFING
Since 1970
Username
Password
Login
Forgot Password?
Register