#### **Day 3:**

**Git Stash**

#### **Step 1: Initialize a Git Repository**

#### **Step 2: Add and Commit Files**

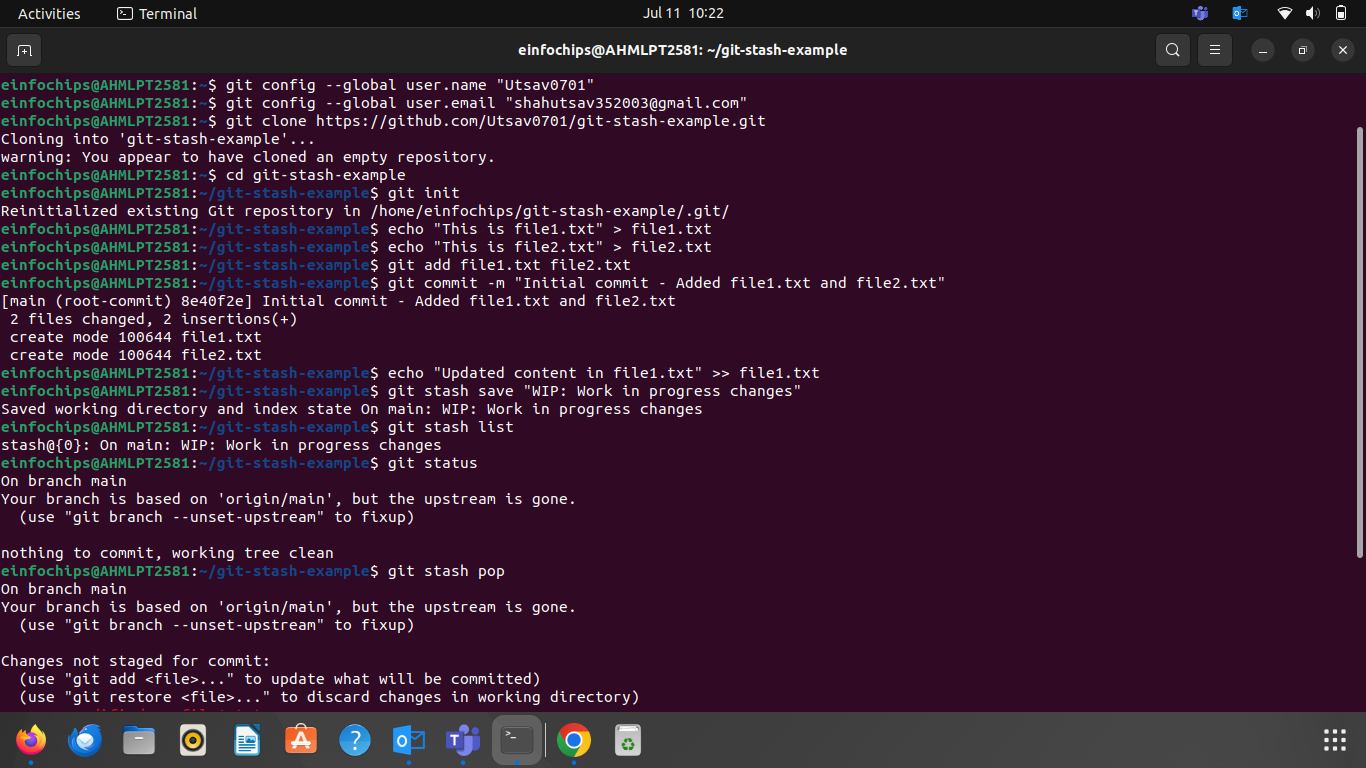
#### **Step 3: Modify Files**

#### **Step 4: Use git stas**

#### **Step 5: Verify Stash**

#### **Step 6: Check Working Directory Status**

#### **Step 7: Apply Stashed Changes**

****

### **Docker Project 01**

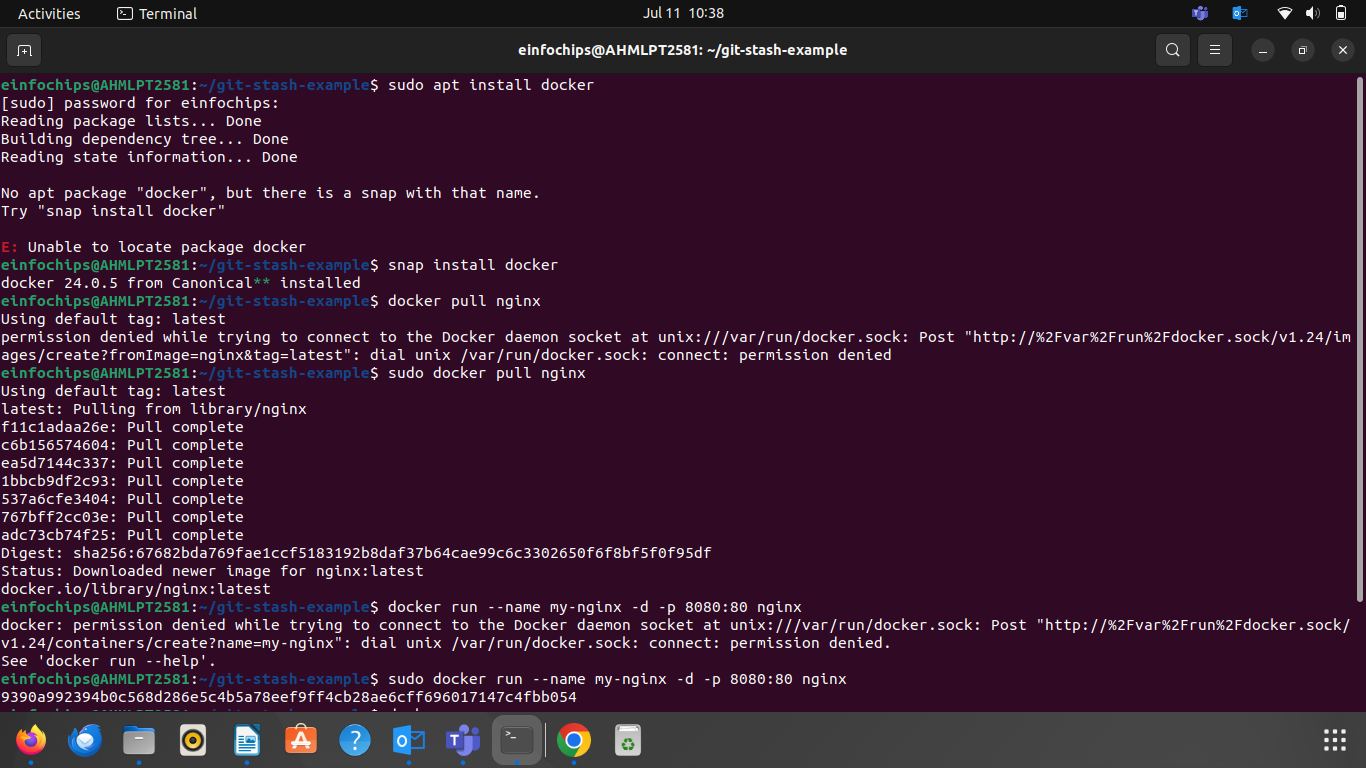
### **Part 1: Creating a Container from a Pulled Image**

**Objective:** Pull the official Nginx image from Docker Hub and run it as a container.

**Steps:**

**Pull the Nginx Image:**

**Run the Nginx Container:**



### **Part 2: Modifying the Container and Creating a New Image**

**Objective:** Modify the running Nginx container to serve a custom HTML page and create a new image from this modified container.

**Steps:**

**Access the Running Container:**

**Create a Custom HTML Page:**

**Exit the Container:**

**Commit the Changes to Create a New Image:**  
**Run a Container from the New Image:**



1. **Verify the New Container:**



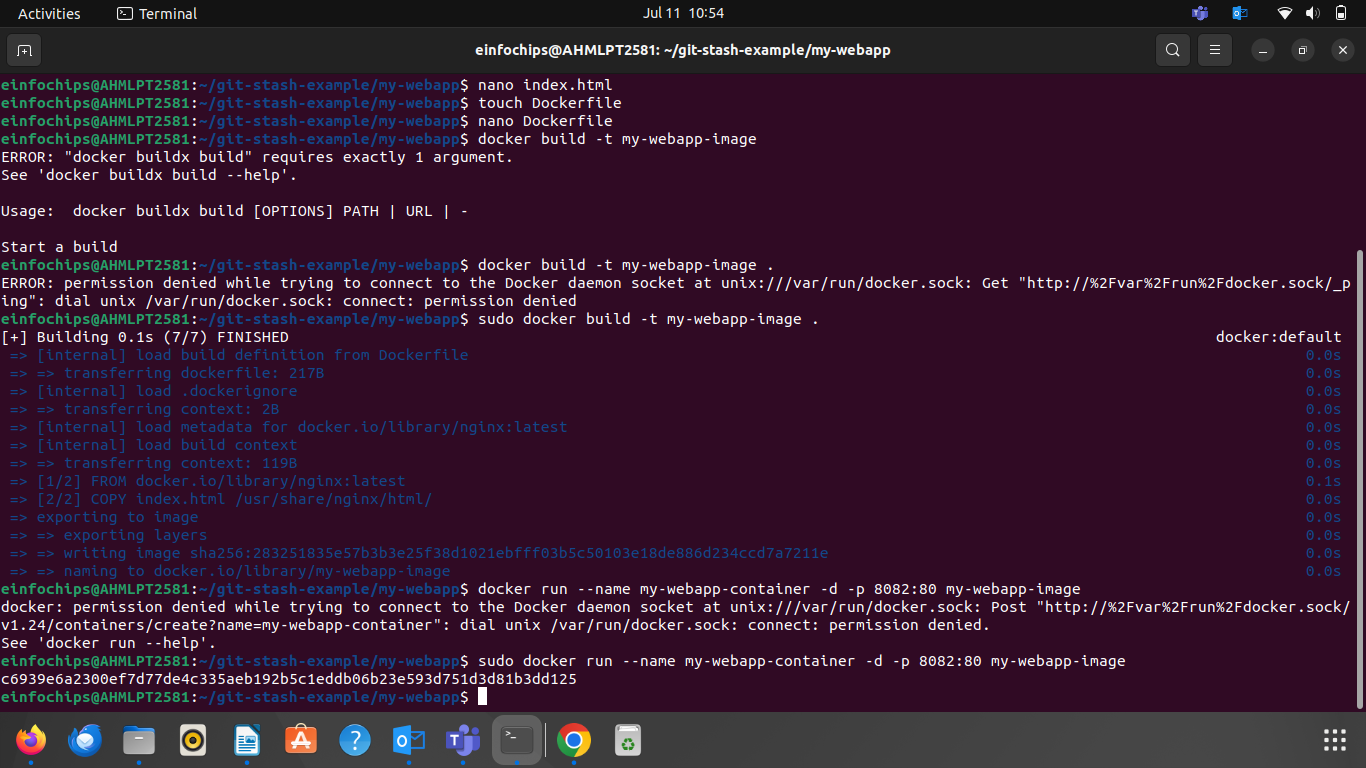
### **Part 3: Creating a Dockerfile to Build and Deploy a Web Application**

**Objective:** Write a Dockerfile to create an image for a simple web application and run it as a container.

**Steps:**

**Create a Project Directory:**

1. **Create a Simple Web Application:**
2. **Write the Dockerfile:**
3. **Build the Docker Image:**
4. **Run a Container from the Built Image:**  
   **Verify the Web Application:**



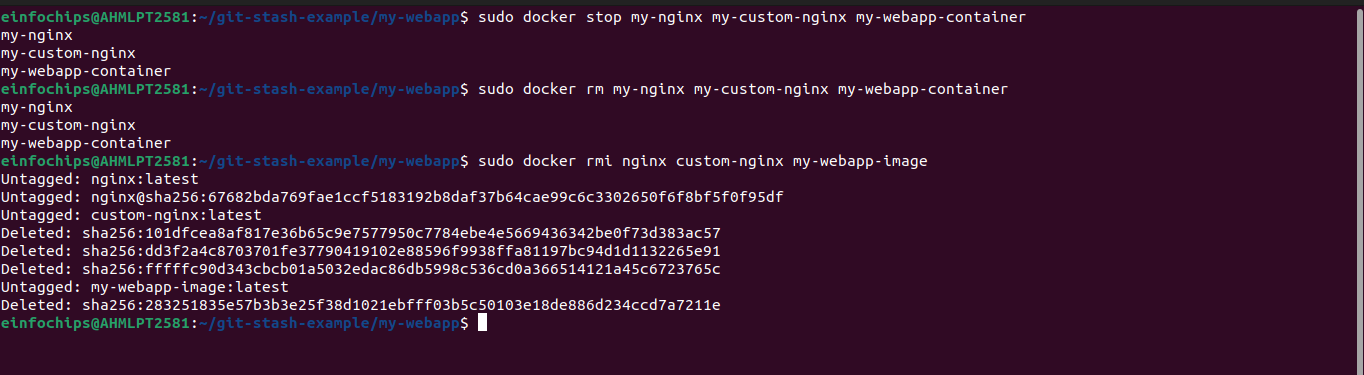
### **Part 4: Cleaning Up**

**Objective:** Remove all created containers and images to clean up your environment.

**Steps:**

**Stop and Remove the Containers:**

1. **Remove the Images:**



**Docker Project 02**

#### **Project Overview**

In this advanced project, you'll build a full-stack application using Docker. The application will consist of a front-end web server (Nginx), a back-end application server (Node.js with Express), and a PostgreSQL database. You will also set up a persistent volume for the database and handle inter-container communication. This project will take more time and involve more detailed steps to ensure thorough understanding.

### **Part 1: Setting Up the Project Structure**

**Objective:** Create a structured project directory with necessary configuration files.

**Steps:**

**Create the Project Directory:**

**Create Subdirectories for Each Service:**

1. **Create Shared Network and Volume:**

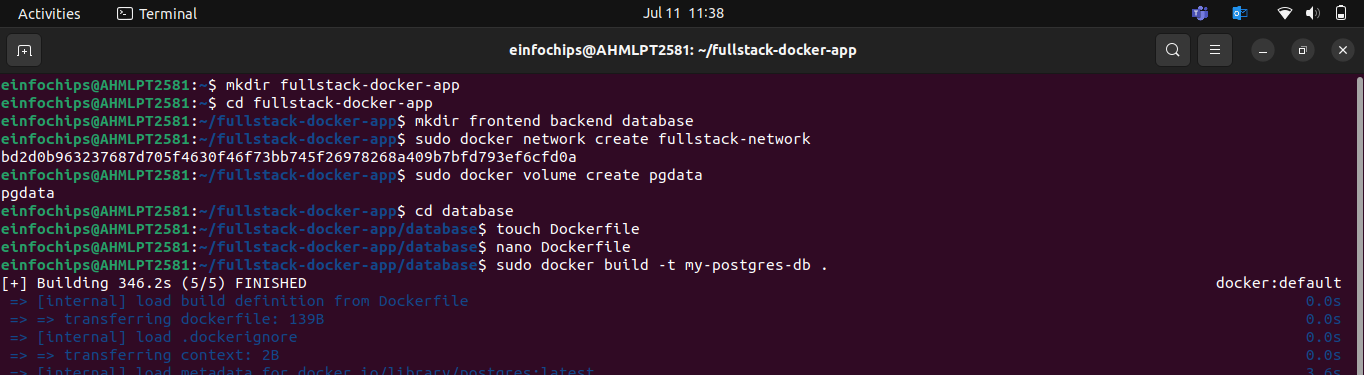
### **Part 2: Setting Up the Database**

**Objective:** Set up a PostgreSQL database with Docker.

**Steps:**

1. **Create a Dockerfile for PostgreSQL:**

**Build the PostgreSQL Image:**



**Run the PostgreSQL Container:**  
docker run --name postgres-container --network fullstack-network -v pgdata:/var/lib/postgresql/data -d my-postgres-db

### **Part 3: Setting Up the Backend (Node.js with Express)**

**Objective:** Create a Node.js application with Express and set it up with Docker.

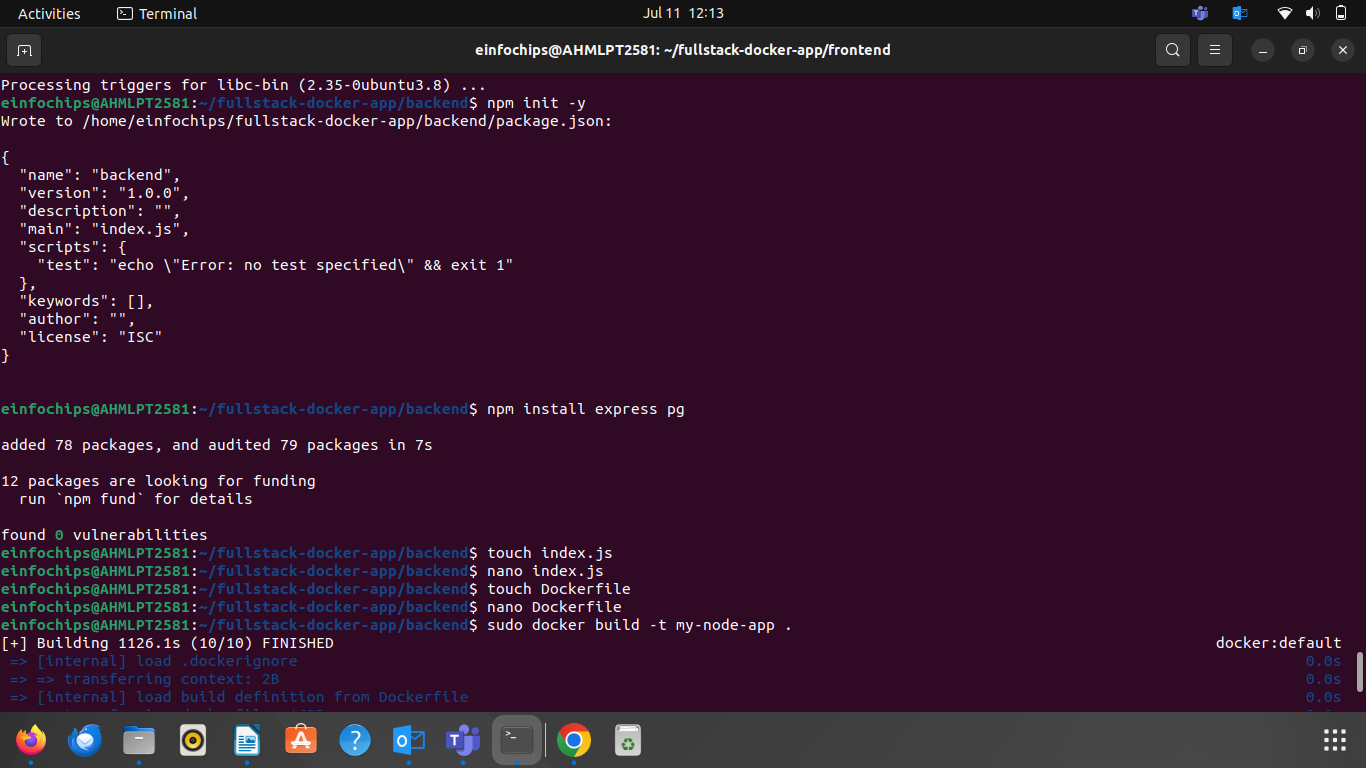
**Steps:**

**Initialize the Node.js Application:**

**1. Install Express and pg (PostgreSQL client for Node.js):**  
2. **Create the Application Code:**

**3. Create a Dockerfile for the Backend:**

**4. Build the Backend Image:**



**5. Run the Backend Container:**

### **Part 4: Setting Up the Frontend (Nginx)**

**Objective:** Create a simple static front-end and set it up with Docker.

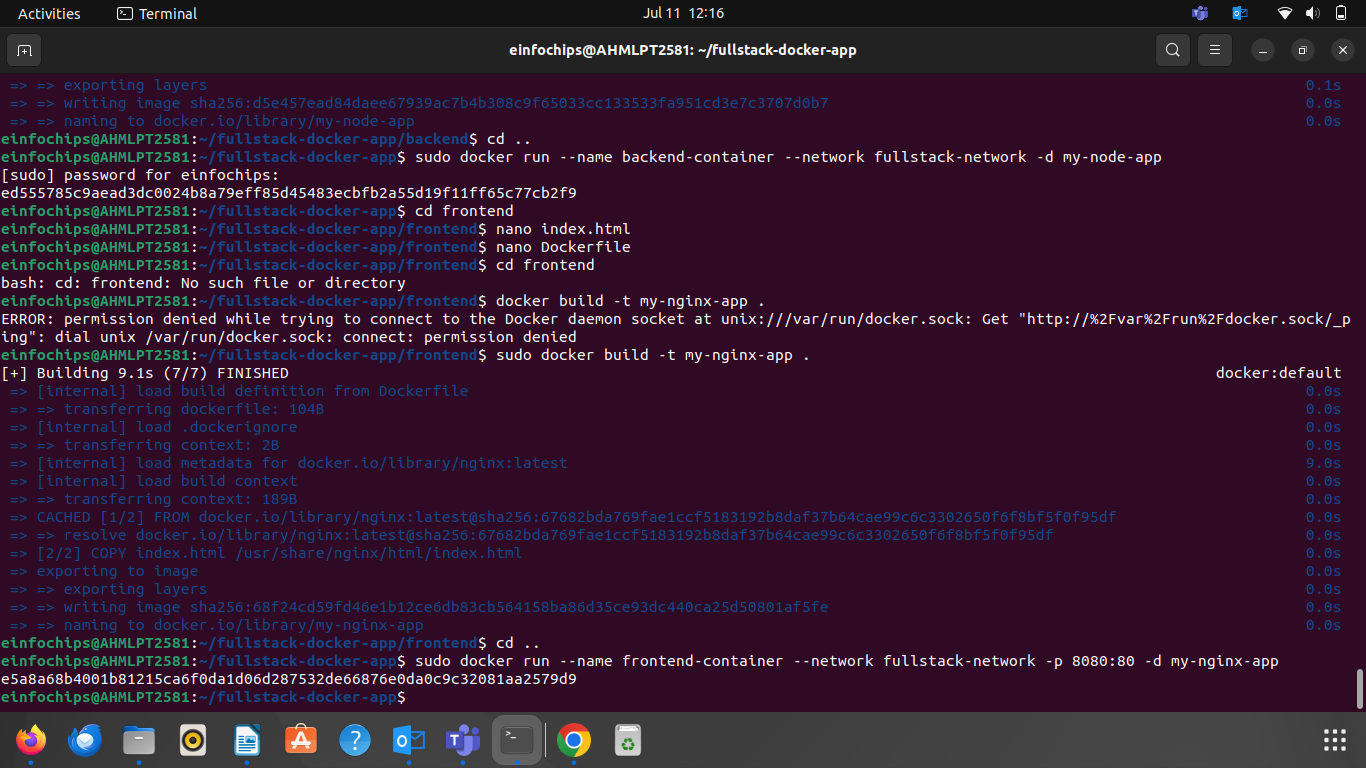
**Steps:**

**1. Create a Simple HTML Page:**

**2. Create a Dockerfile for the Frontend:**

**3. Build the Frontend Image:**

**4. Run the Frontend Container:**

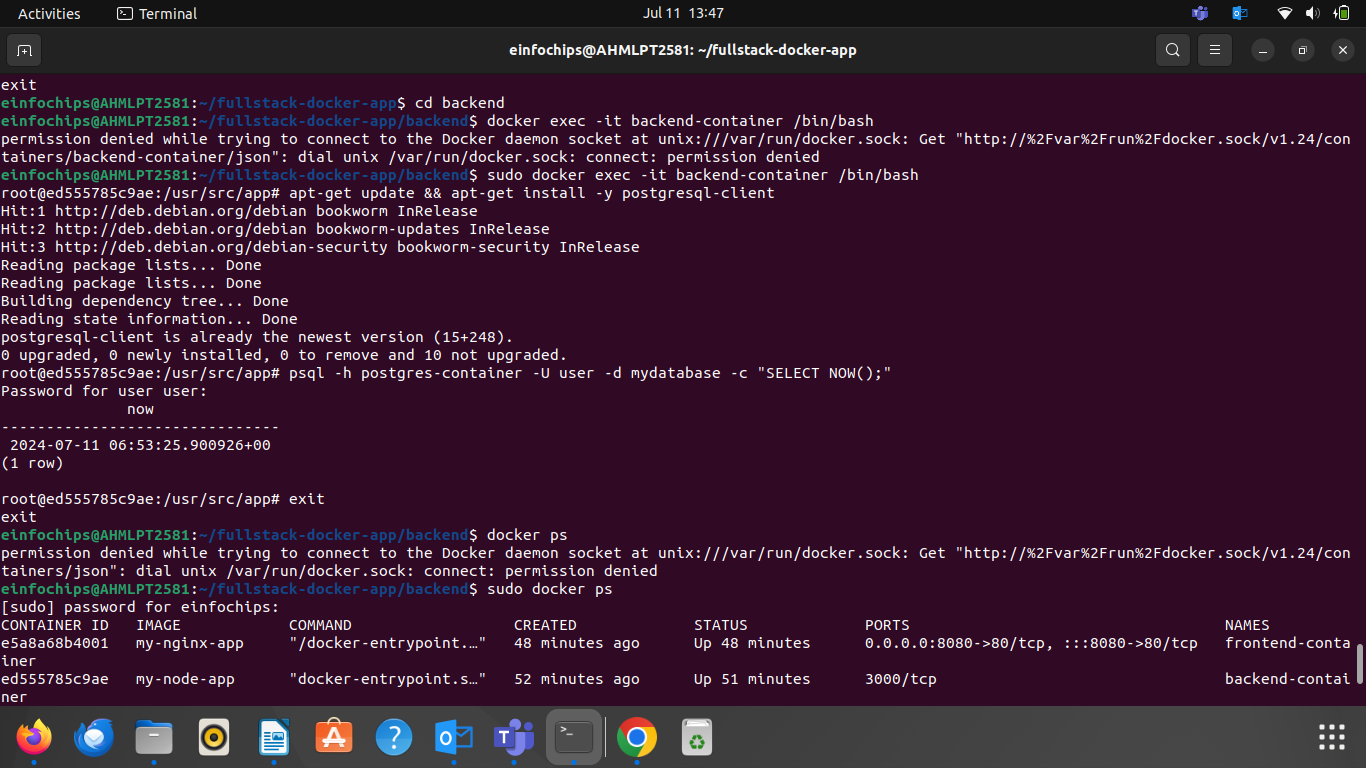


**Part 5: Connecting the Backend and Database**

**Objective:** Ensure the backend can communicate with the database and handle data requests.

**Steps:**

1. **Update Backend Code to Fetch Data from PostgreSQL:**
2. **Verify Backend Communication:**
3. **Test the Backend API:**

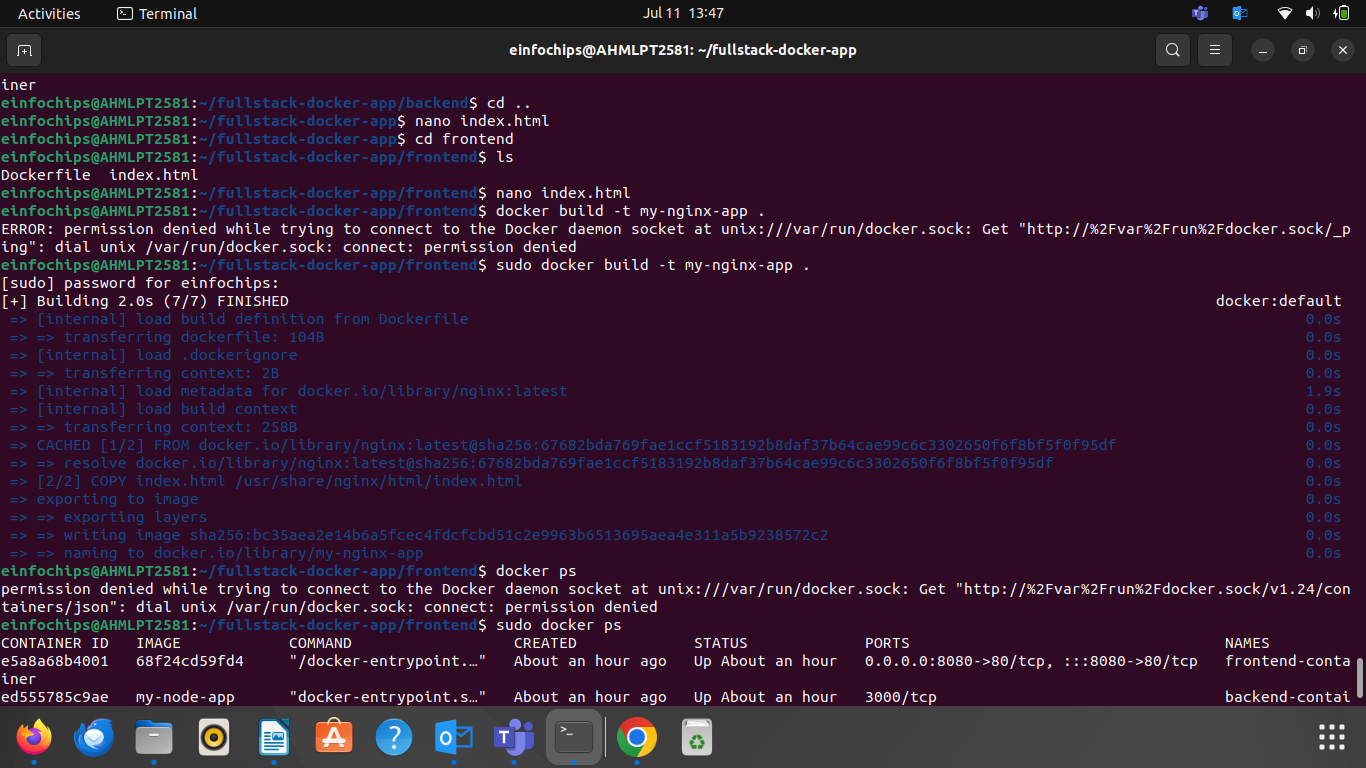


### **Part 6: Final Integration and Testing**

**Objective:** Ensure all components are working together and verify the full-stack application.

**Steps:**

1. **Access the Frontend:**
2. **Verify Full Integration:**
3. **Rebuild and Run the Updated Frontend Container:**
4. **Final Verification:**



**Part 7: Cleaning Up**

**Objective:** Remove all created containers, images, networks, and volumes to clean up your environment.

**Steps:**

1. **Stop and Remove the Containers:**
2. **Remove the Images:**
3. **Remove the Network and Volume:**

