

**SPRINT-I**

**Programming and Application**

Stateless Claims Management System

Contents

[**Objective for Numiqers: Construct the foundational backend logic for a Claims Management System without the use of persistent storage, focusing on in-memory data structures.** 3](#_Toc160534966)

[**Define Basic Entities:** Outline entities such as Claim, Policyholder, Policy, etc., using appropriate data structures. 3](#_Toc160534967)

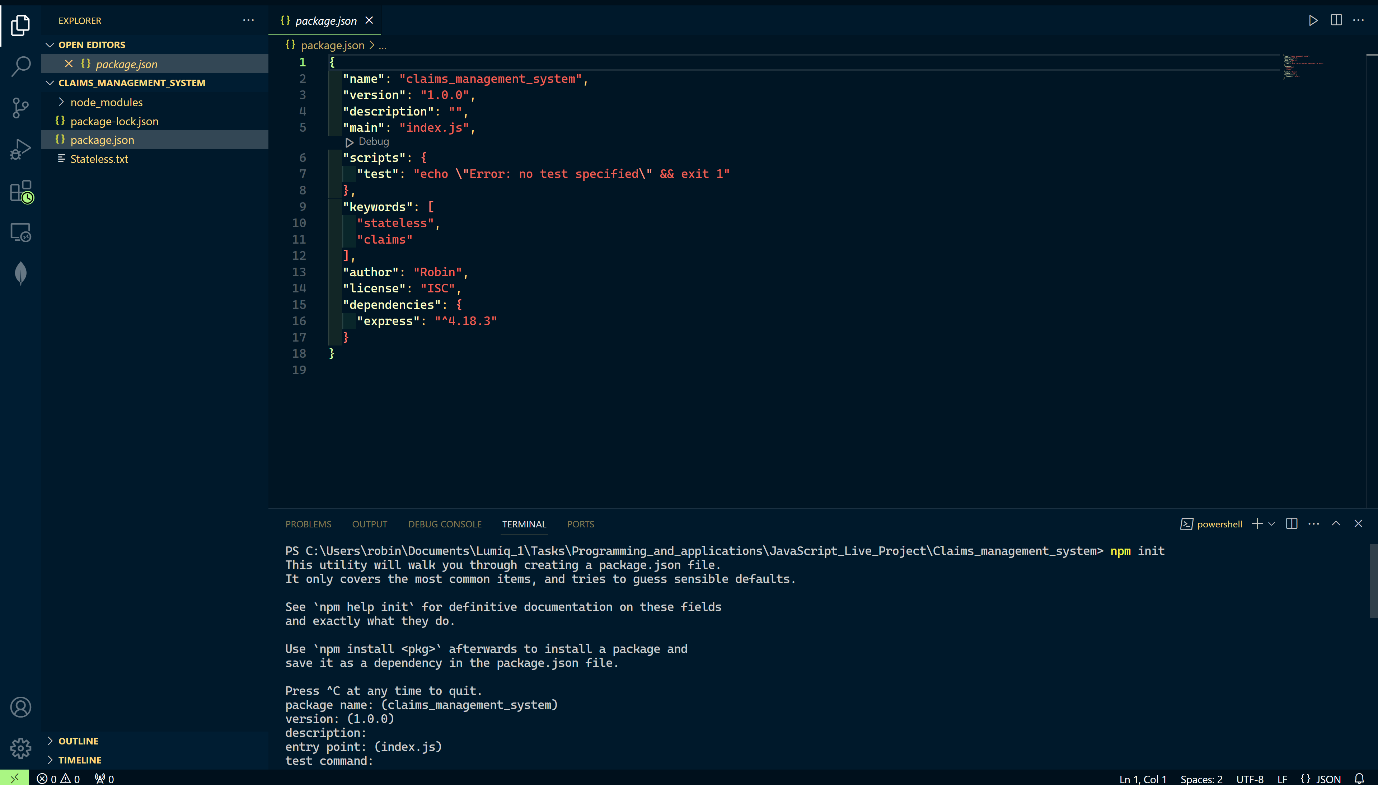
[Development 3](#_Toc160534968)

# **Objective for Numiqers: Construct the foundational backend logic for a Claims Management System without the use of persistent storage, focusing on in-memory data structures.**

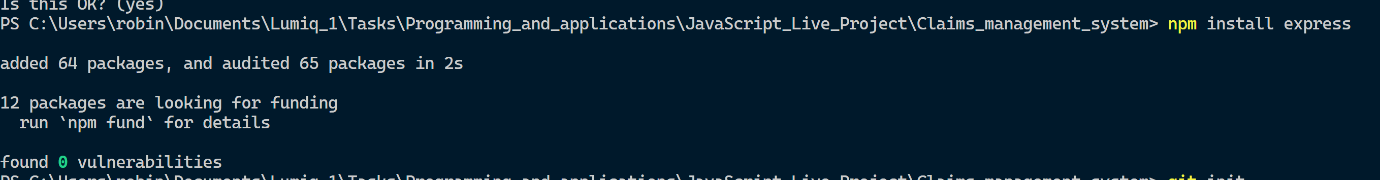
## **Define Basic Entities:** Outline entities such as Claim, Policyholder, Policy, etc., using appropriate data structures.

### Development

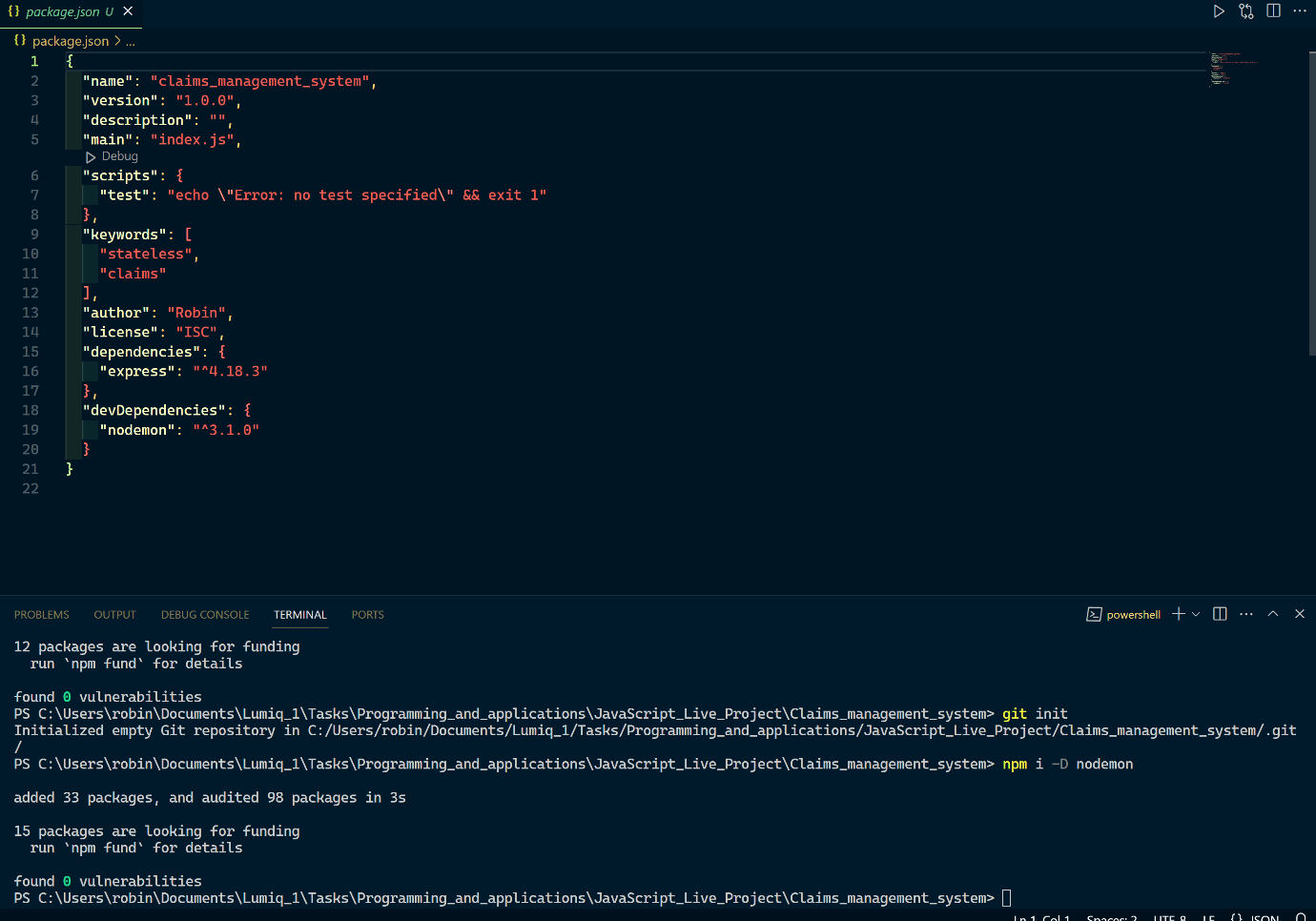
1. First I’ll initialize the terminal with the node environment for necessary system files using ‘npm init’.



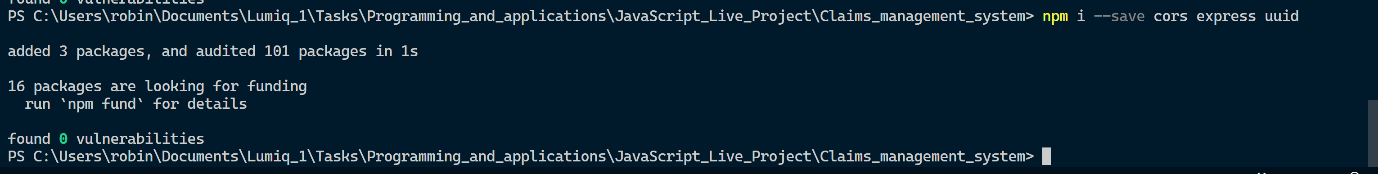
1. After that I’ll be installing the ‘express’ in order to create our server using ‘npm install express’.



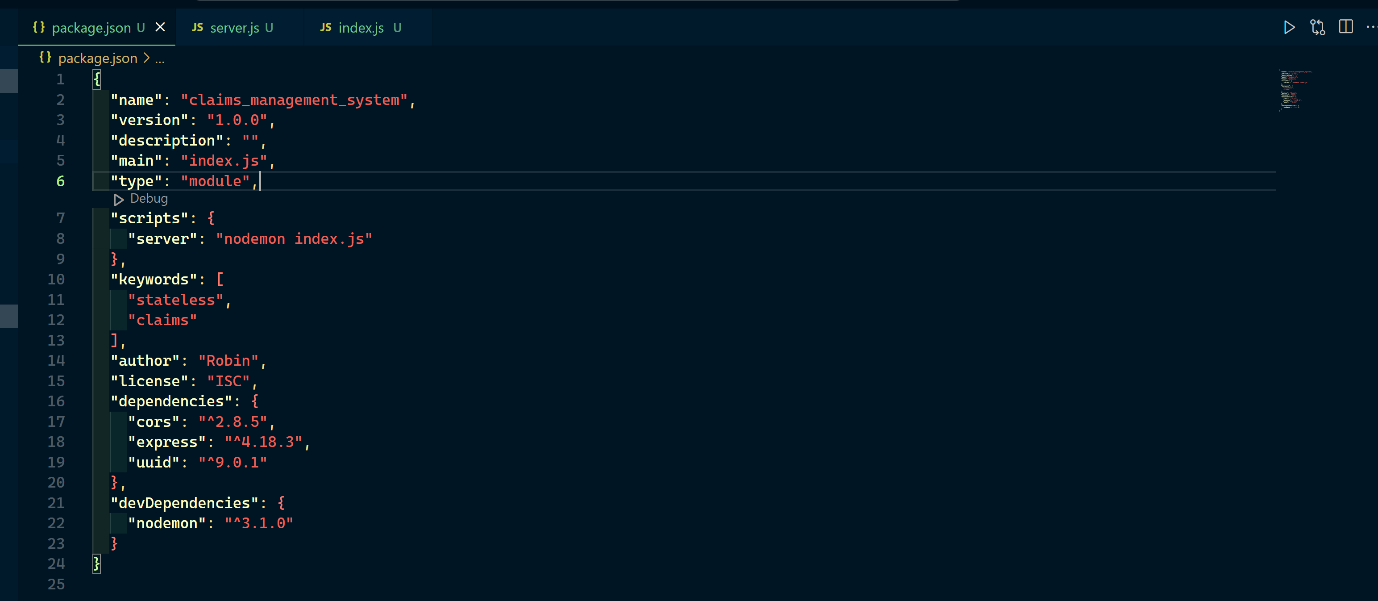
1. Further I’ll be installing nodemon, so that I don’t have to restart the server after every changes I make. Nodemon will restart the server by itself.
2. We can see the nodemon package in devDependencies now.



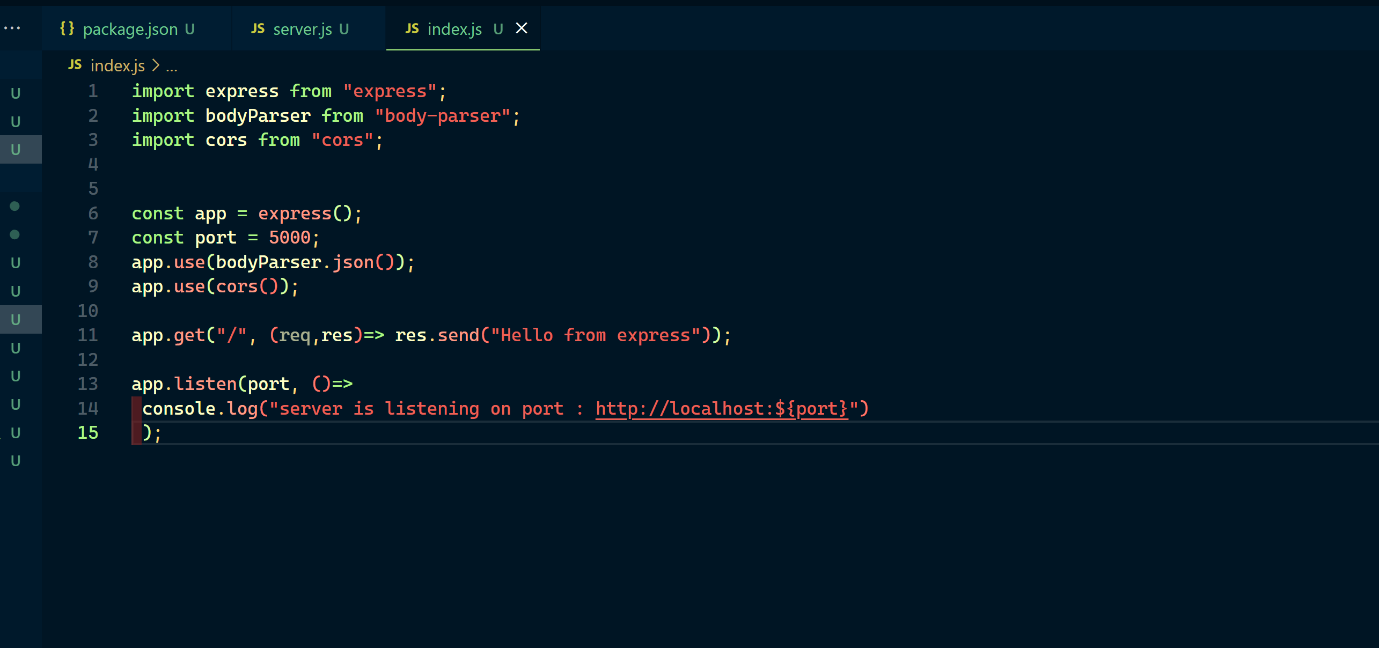
1. I’ll be using installing the CORS for cross-domain requests and security, then uuid for unique id generation for each of my entries.



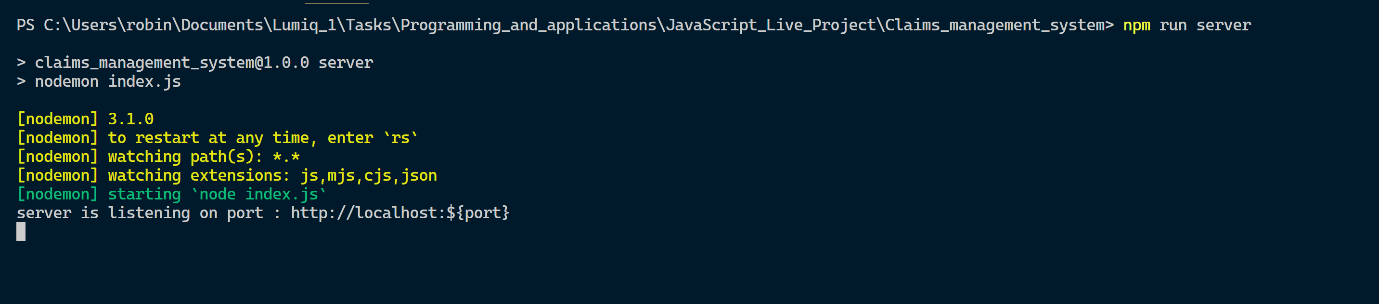
1. In the scripts under package.json I’ll update the nodemon for index.js file, which is our main file.



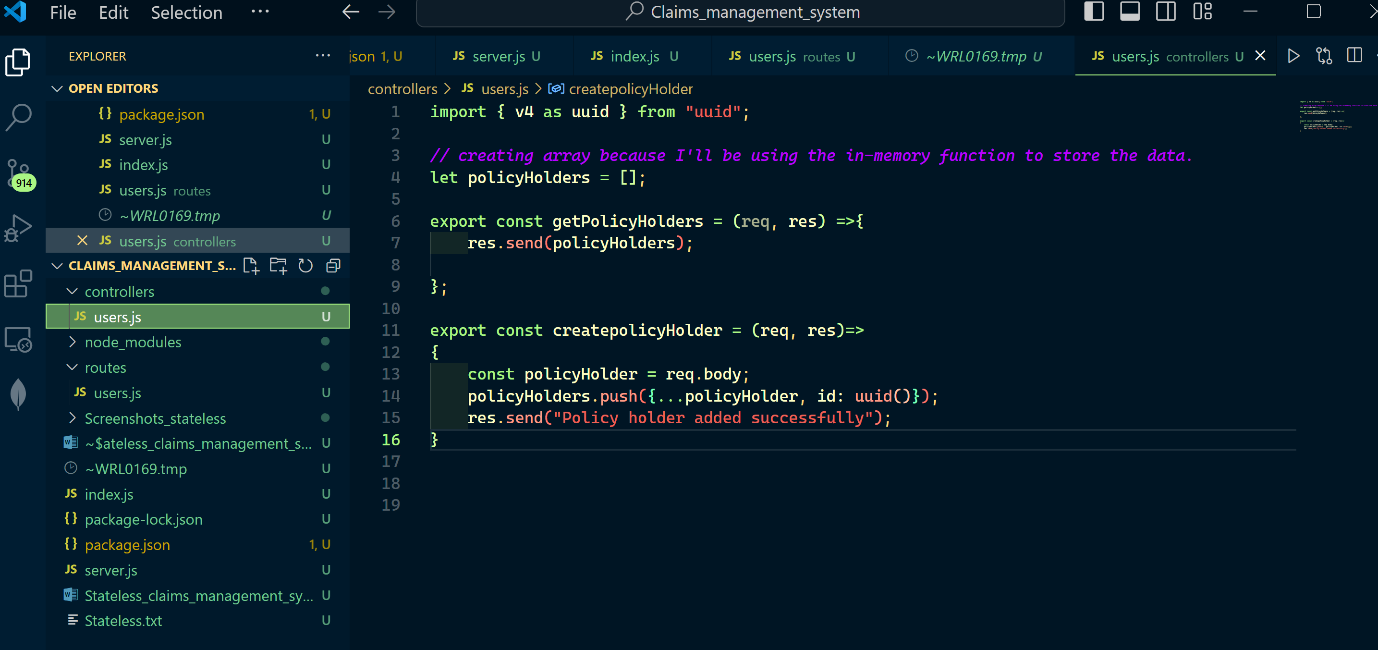
1. Using we’ll set up the server for listening on port 5000.



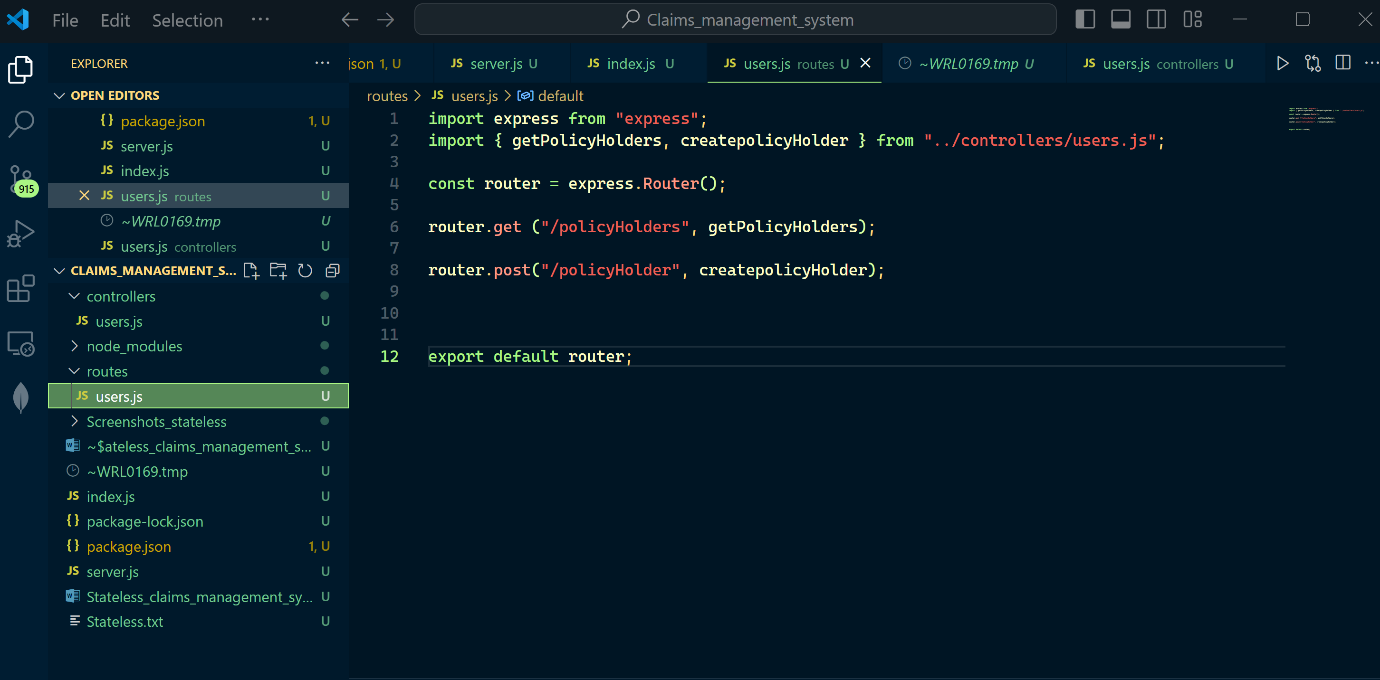
1. I’ll now test the server if it’s running or not.
2. As we can see the server is running perfectly on port 5000.



1. Now I’ll create folder like controller which will handle all our CRUD operations logics.
2. Another folder for routes will be created for all the route definition.
3. Under routes and controllers folder, I’ll create a file under the same name called ‘users.js’.
4. I am creating two logics for reading the policyHolders and creating the policyHolder under controllers folder in users.js file. Since it’s an in-memory operation, so I’ll be making an empty array for policyHolders to store all the data requested.



1. Now, I’ll set up the users.js under routes for all the route handling. Also the getPolicHolders and createpolicyHolder method is being imported from controller.



1. I’ll import userRoutes into index.js file and create a rule to send a message for other access endpoints which is not hosted by me.

