APP SECURITY

The 4 aspects (Authentication , Authorization , Auditing and Administration) collectively contribute to robust app security, safeguarding against unauthorized access, maintaining accountability, and enabling efficient management.

Authentication: 

Purpose: Authentication is the process of verifying the identity of a user or system attempting to access an application.

How It Works: Users provide credentials (such as username and password) to prove their identity. The system compares these credentials with stored data to grant or deny access.

Importance: Proper authentication ensures that only legitimate users gain access to the application.

Authorization: 

Purpose: Authorization determines what actions a user is allowed to perform within the application after successful authentication.

How It Works: Based on the user's identity and role, authorization defines access privileges to specific resources, features, or functionalities.

Example: A regular user may have limited access, while an administrator has broader permissions.

Importance: Authorization prevents unauthorized actions and maintains data security.

Auditing: 

Purpose: Auditing tracks user activity within the application.

How It Works: The system records details such as login times, data sent/received, IP addresses, and accessed services.

Use Cases:

Analyzing user trends.

Auditing user activity for compliance.

Accurate billing based on usage.

Importance: Auditing provides insights, accountability, and helps detect anomalies or security breaches.

Administration: 

Purpose: Administration involves managing the application, including user accounts, permissions, and system settings.

Responsibilities:

User account creation, modification, and deletion.

Role assignment (e.g., user, admin).

Configuring security policies.

Importance: Effective administration ensures smooth operation, security, and efficient resource utilization.

In this lab, you'll dive into the fundamental concepts of securing web applications by understanding how to authenticate users and control their access to different parts of your system (Authorization).

We will build the backend of a Web App that will be used for a fictitious CCA club in Temasek Polytechnic

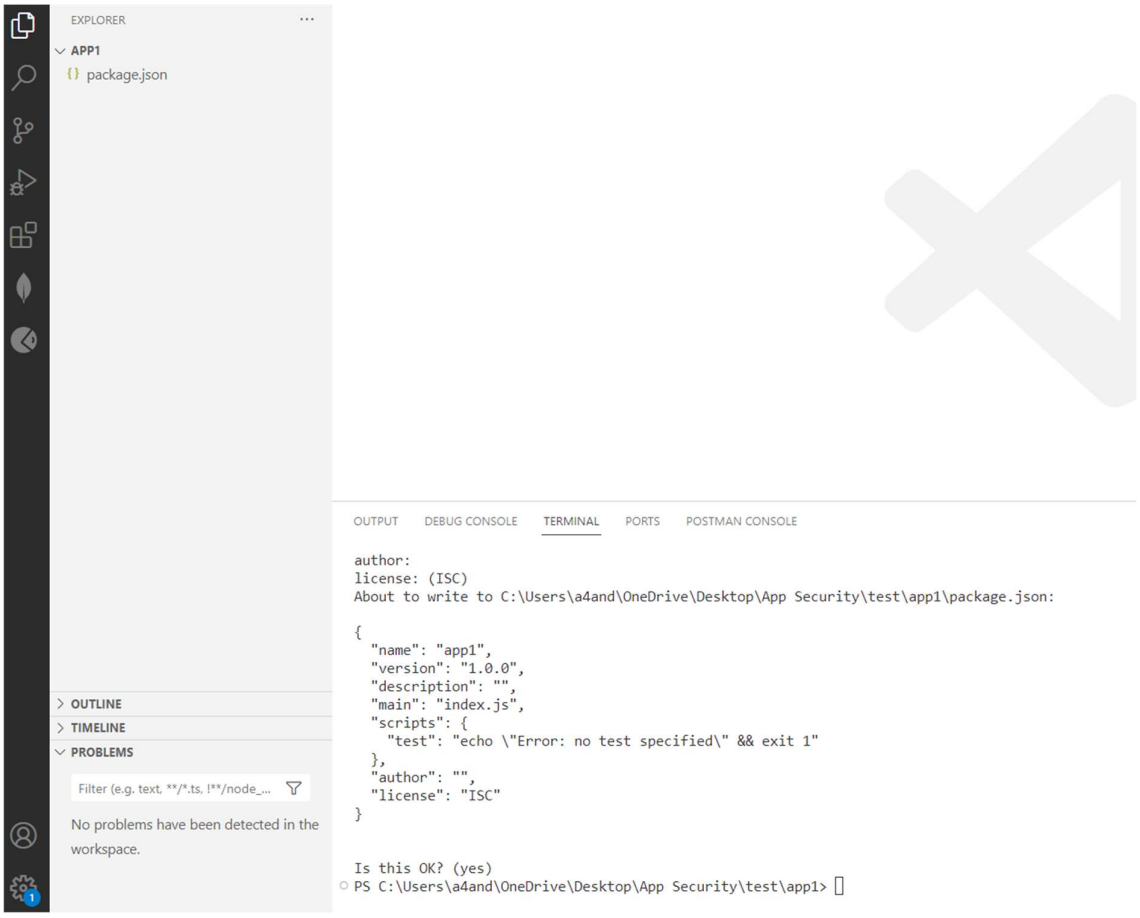
Building our Node.js Web Server for connection to CloudAtlas MongoDB – Part 1

First, we'll create a directory for our server. Navigate to a suitable directory and run the following code in your terminal:

```
mkdir app1
```

After creating our directory, we'll navigate to this directory and initialize npm. The npm init command is used to initialize a new or existing npm package. When you run this command, it creates a package.json file for your project. You can press “enter” to accept the default values.

```
npm init
```

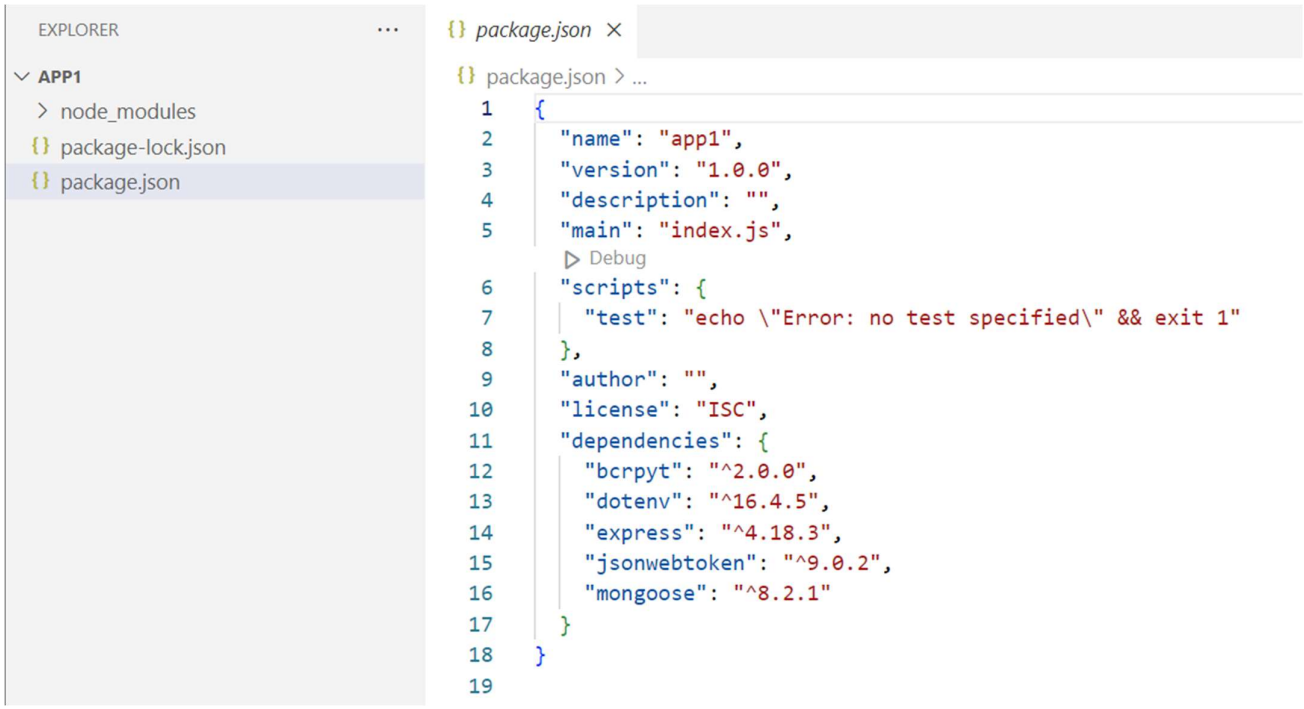


Installing the Required Node Packages

For this project, we'll use the following dependencies and packages:

- dotenv: This package loads environmental variables from an env file into Node's process.env object.
- bcrypt: This is used to hash our passwords and other sensitive information before sending them to the database to protect us against a breach of our database.
- jsonwebtoken: This provides a means of representing claims transferred between two parties, ensuring that the information transferred has not been tampered with by an unauthorized third party.
- Express.js: This makes building APIs and server-side applications with Node.js effortless by providing us with useful features such as routing, implementing middleware, and so on.
- Mongoose: Helps us connect with our database and provides features such as schema validation, managing relationships between data, etc.

```
npm i jsonwebtoken@8.5.1 mongoose@6.6.5 bcrypt@5.1.0
express@4.18.2 dotenv@16.0.3
```


EXPLORER
APP1
node_modules
package-lock.json
package.json
package.json
package.json > ...
Debug
{
"name": "app1",
"version": "1.0.0",
"description": "",
"main": "index.js",
"scripts": {
"test": "echo \"Error: no test specified\" && exit 1"
},
"author": "",
"license": "ISC",
"dependencies": {
"bcrpyt": "^2.0.0",
"dotenv": "^16.4.5",
"express": "^4.18.3",
"jsonwebtoken": "^9.0.2",
"mongoose": "^8.2.1"
}
}