**Quality Assurance and Security in the Bookshelf Web Application**

**1. Testing Techniques Utilized**

To ensure the **quality and functionality** of the Bookshelf web application, the following testing methodologies are applied:

**1.1 Unit Testing**

* **Framework Used:** PHPUnit (for PHP) & Jest (for JavaScript)
* **Files:** FunctionsTest.php, bootstrap.test.js
* **Purpose:**
  + Validate individual functions and components.
  + Ensure correctness of **business logic**, including validation of data processing functions, API interactions, and calculation methods, without dependencies.
* **Examples:**
  + FunctionsTest.php: Tests core PHP functions.
  + bootstrap.test.js: Tests JavaScript utilities using Jest.

**1.2 Integration Testing**

* **Framework Used:** PHPUnit
* **File:** IntegrationTest.php
* **Purpose:**
  + Validate how different modules interact (e.g., database queries, authentication, and third-party API integrations such as payment gateways or external data sources).
  + Ensure the **customer order process** works correctly by verifying database queries interact properly with the **Azure-hosted SQL Server database**.
* **Example:**
  + Tests insertion of a customer and order in the **SQL Server database**.

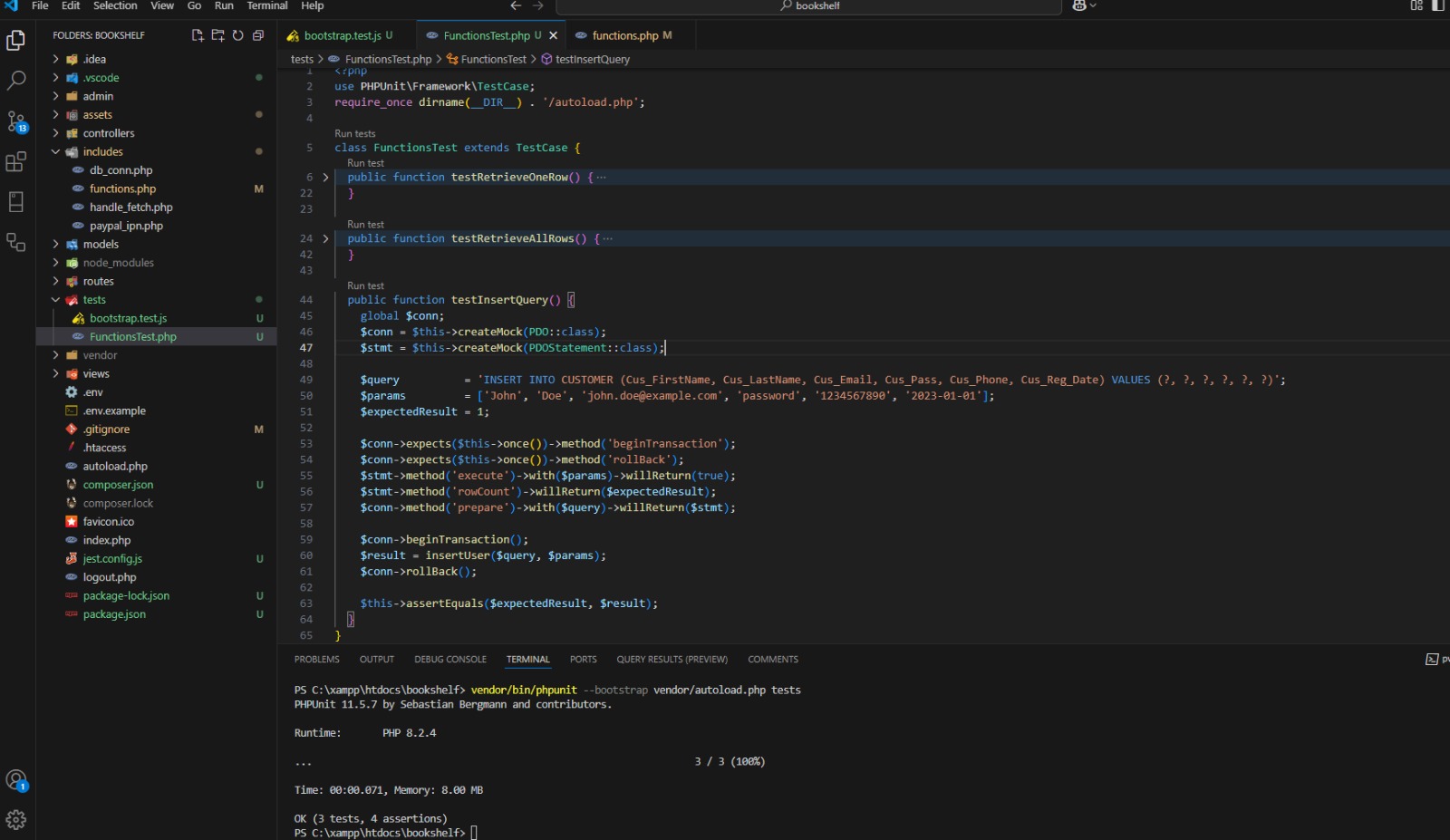
**1.3 End-to-End (E2E) Testing**

* **Framework:** Jest (for frontend testing)
* **File:** jest.config.js
* **Purpose:**
  + Simulate user interactions with the web application.
  + Ensure the **frontend and backend work seamlessly**.

**2. Testing Pyramid Implementation**

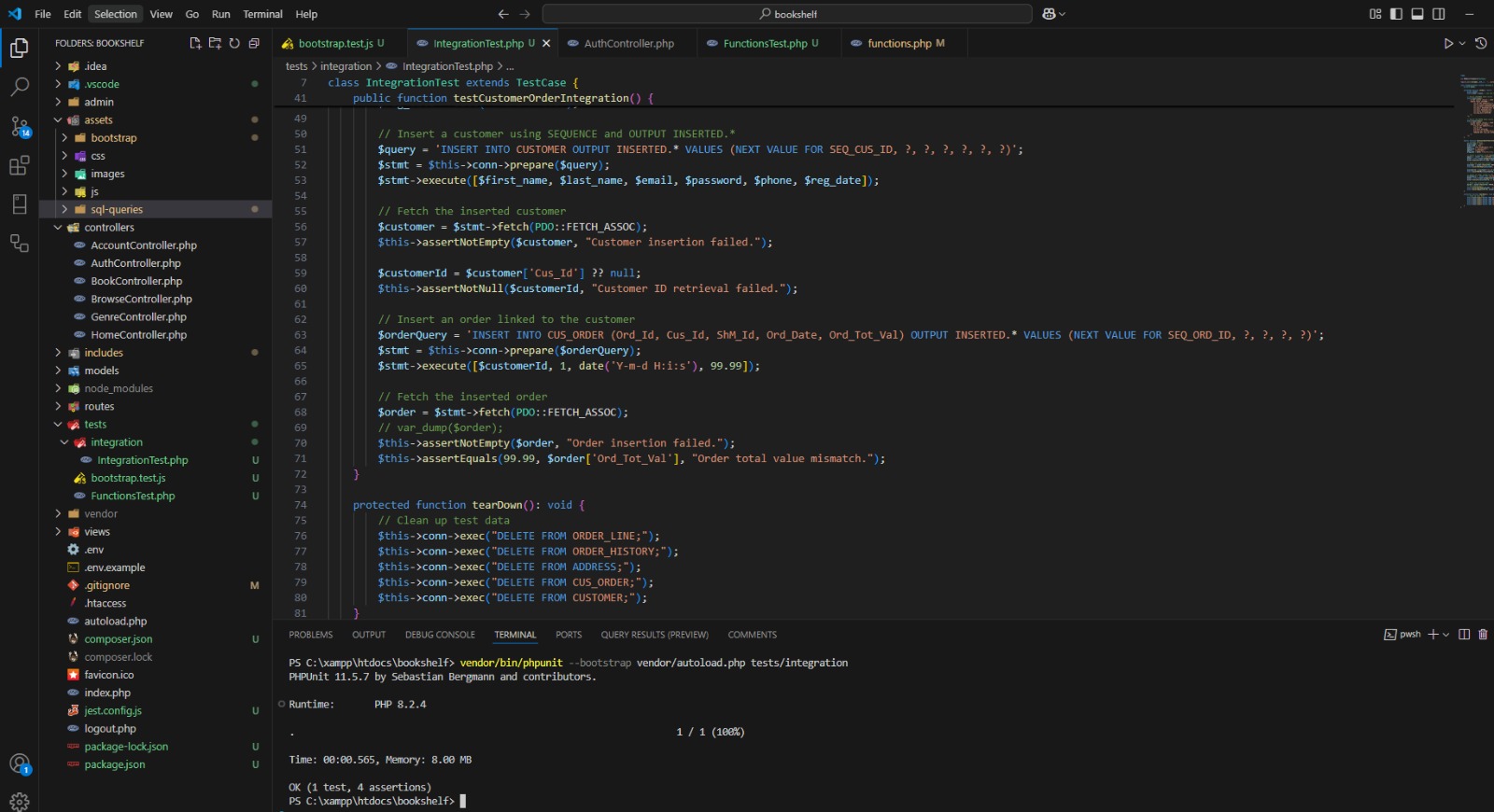
The **Testing Pyramid** follows a layered approach:

**2.1 Unit Tests (Bottom Layer – Fastest & Most)**

****

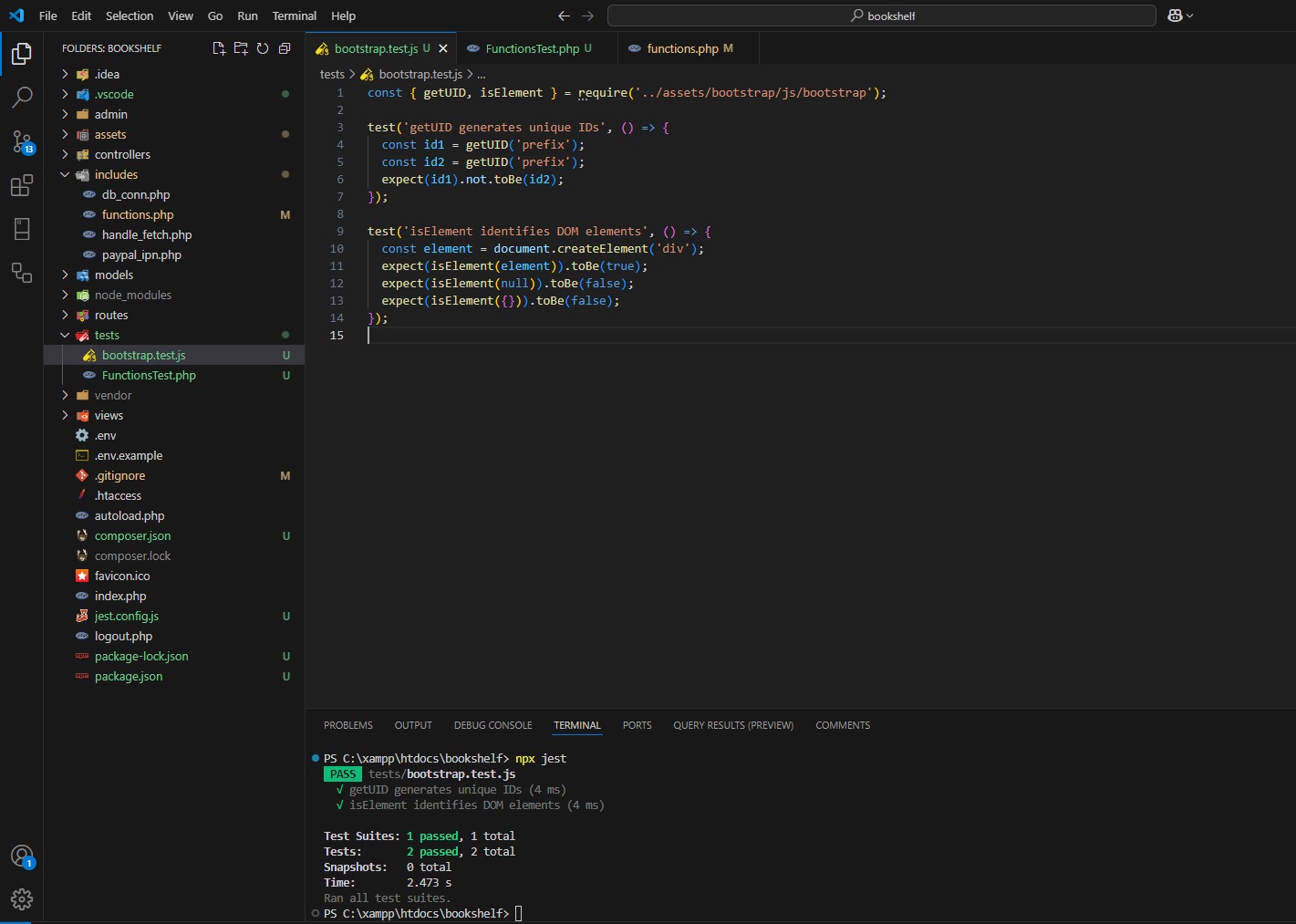
* **PHP:** FunctionsTest.php
* **JavaScript:** bootstrap.test.js
* **Execution Speed:** Fastest
* **Tools:** PHPUnit, Jest
* **Purpose:**
  + Verify individual methods in isolation.

**2.2 Integration Tests (Middle Layer)**

****

* **File:** IntegrationTest.php
* **Execution Speed:** Moderate
* **Tools:** PHPUnit
* **Purpose:**
  + Validate database interactions.
  + Ensure API endpoints work correctly.

**2.3 End-to-End (E2E) Tests (Top Layer – Slowest & Fewest)**

****

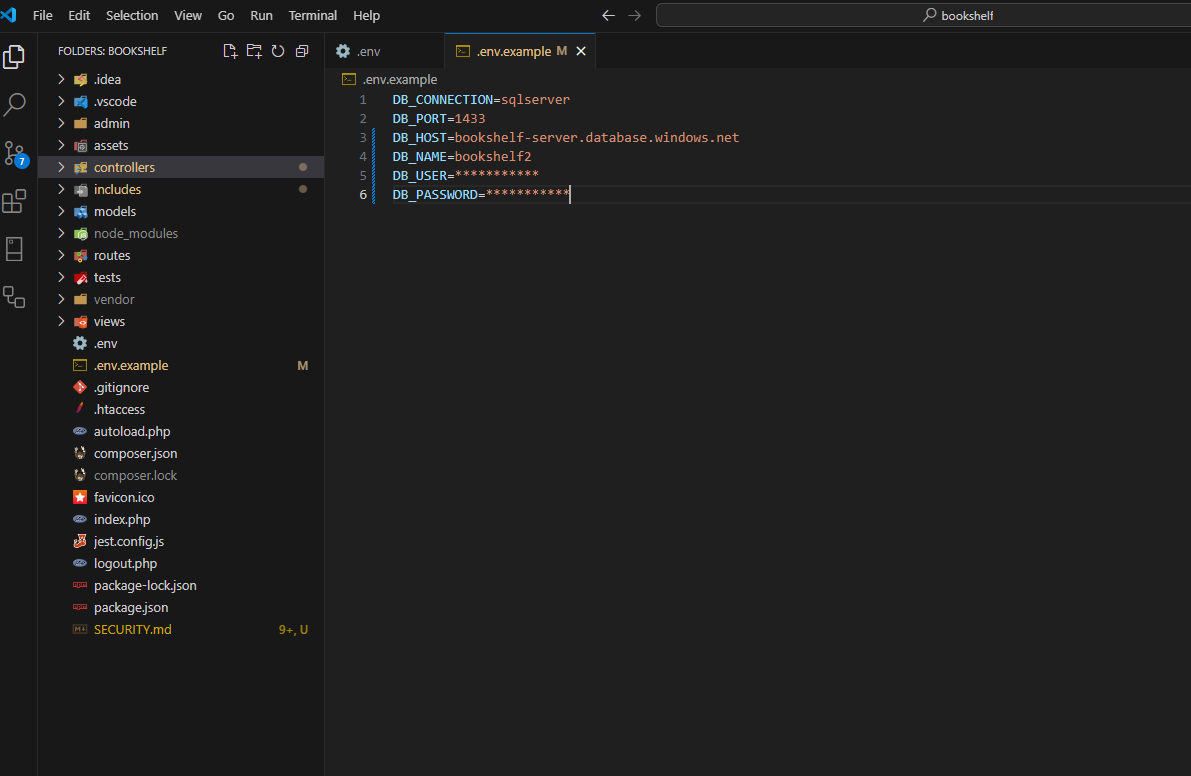
* **File:** Uses Jest and JavaScript DOM testing.
* **Execution Speed:** Slowest
* **Tools:** Jest, Puppeteer (if applicable).
* **Purpose:**
  + Simulate real user interactions.

**3. Security Development Techniques**

Security is integrated throughout the development lifecycle using various best practices, including adherence to industry standards such as OWASP guidelines and secure coding frameworks.

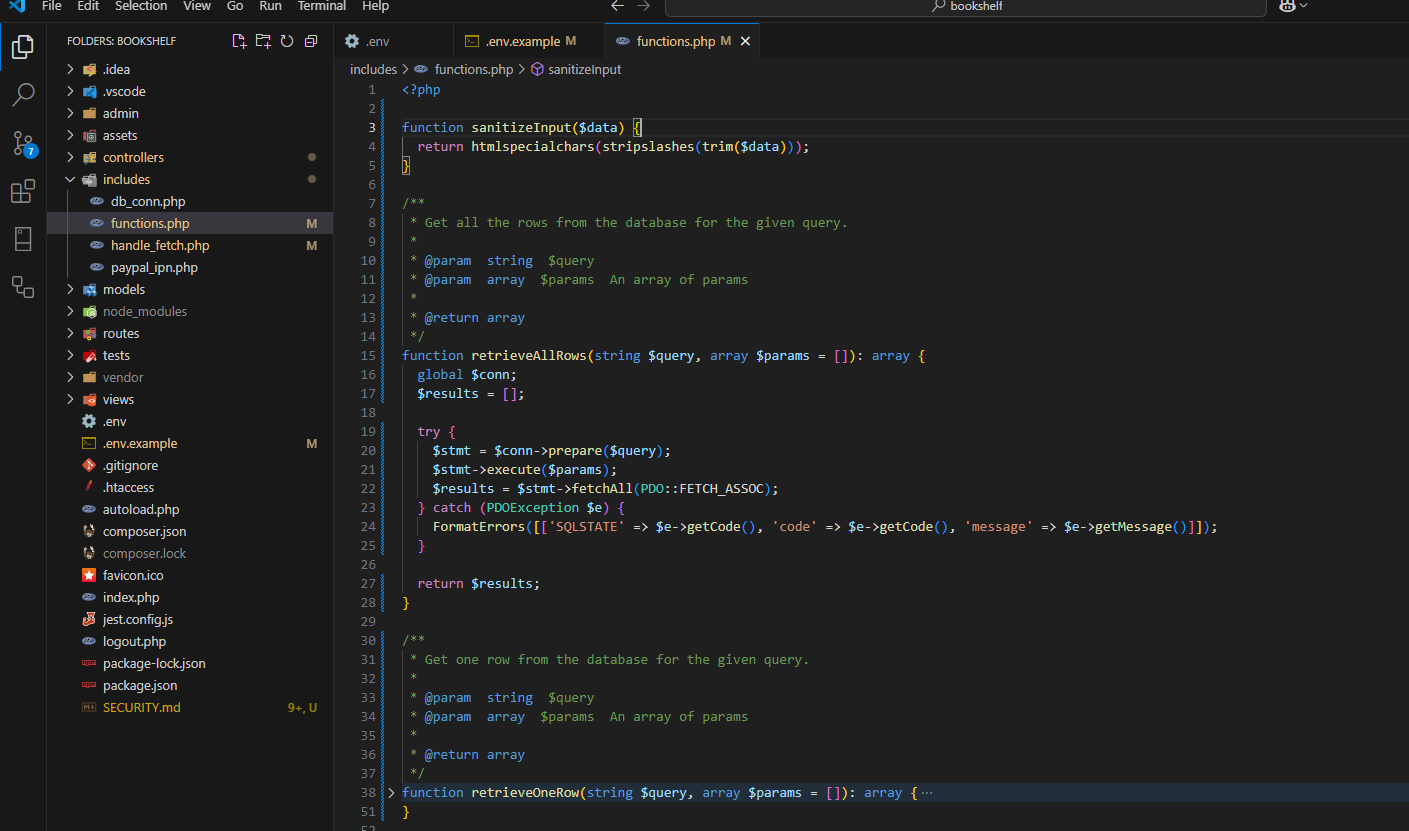
**3.1 Secure Database Connection**

* **Configuration File:** .env
* **Techniques:**
  + Uses **environment variables** for sensitive credentials instead of hardcoding.
  + Secure database credentials:
  + DB\_CONNECTION=sqlserver
  + DB\_HOST=bookshelf-server.database.windows.net
  + DB\_USER=bookshelf-server-admin
  + DB\_PASSWORD=\*\*\*\*\*\*\*\*\*\*\*
  + SQL Server with **Azure-hosted database** for security.



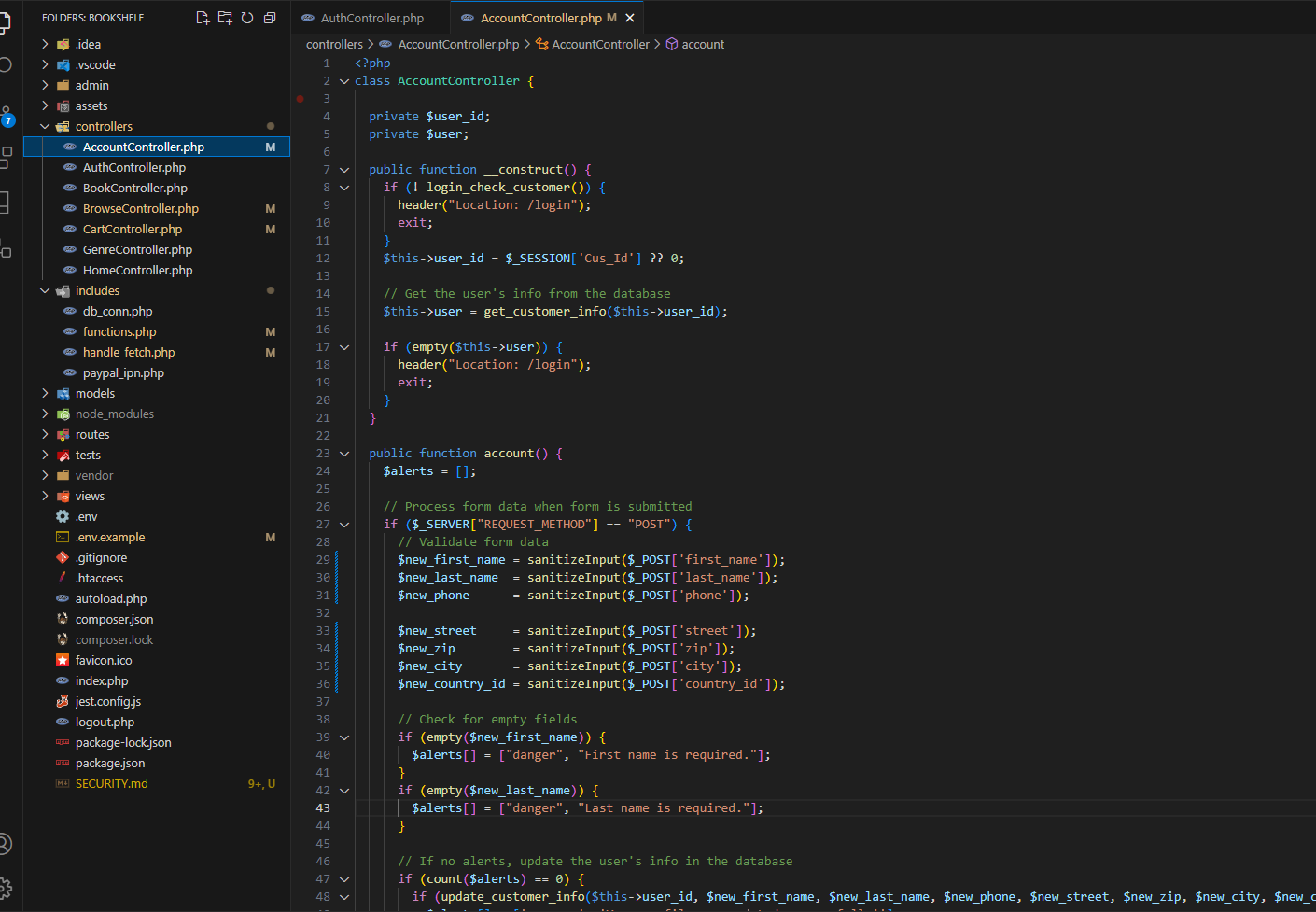
**3.2 Input Validation & Sanitization**

* **File:** functions.php
* **Purpose:**
  + Prevents **SQL Injection** and **XSS** attacks.
* **Techniques:**
  + **Prepared Statements:** Used in database queries.
  + **Sanitization Functions:** Clean user input before processing.



**3.3 Authentication & Session Security**

* **File:** HomeController.php
* **Security Measures:**
  + **User Authentication Checks**
  + **Session Management**
  + Uses **password hashing** for user credentials.



**3.4 Secure Development Practices**

* **Use of HTTPS**
  + The project files are hosted on a secure managed server, and the database is on **Azure**, ensuring secure HTTPS encryption.