TEST DOCUMENT

Testing Scope:

1. **Authentication and Authorization**: Verify that only authenticated users can access authorized functionalities. Test login, logout, registration, password reset, and session management.
2. **Menu Management**: Test CRUD operations for menu items, including adding, updating, deleting, and viewing items. Ensure proper validation and error handling.
3. **Order Management**: Validate the ordering process, including adding items to the cart, modifying quantities, and placing orders. Test order confirmation, cancellation, and tracking.
4. **Feedback Management**: Test the functionality for users to provide feedback to sellers and view their past feedback. Verify that sellers can access and respond to feedback.
5. **Payment Integration**: Validate the integration with Stripe payment gateway for secure and seamless transactions. Test payment processing, order status updates, and error handling.
6. **Security**: Conduct security testing to identify vulnerabilities such as SQL injection, cross-site scripting (XSS), and insecure authentication mechanisms. Ensure data confidentiality, integrity, and availability.
7. **Performance**: Measure the application's performance under various conditions, including peak loads and stress testing. Test response times, scalability, and resource utilization.
8. **Integration Testing**: Validate interactions between different modules, APIs, and third-party services. Test data synchronization, error handling, and data consistency.
9. **Usability**: Evaluate the application's usability and intuitiveness. Test user workflows, error messages, and help/documentation features.
10. **Data Management**: Verify data integrity, validation rules, and data encryption mechanisms. Test database operations, data backup, and recovery procedures.

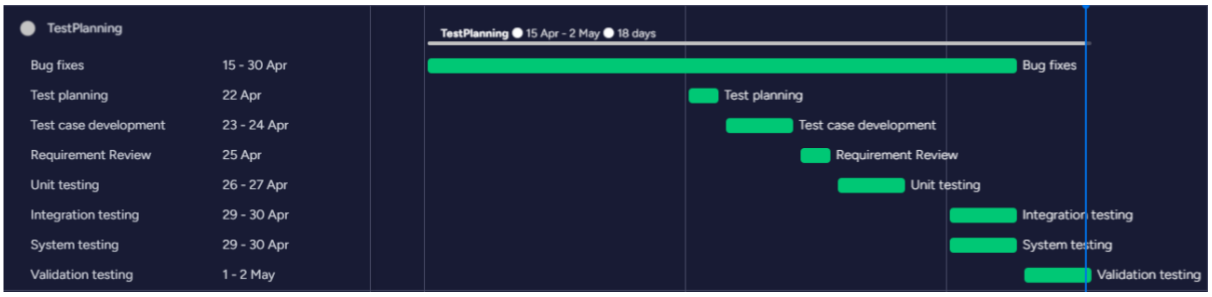
Testing Approach:

1. **Test Planning:** Begin by reviewing the project requirements, user stories, and acceptance criteria to define the scope of testing. Identify key functionalities and use cases that need to be tested.
2. **Test Case Development:** Create comprehensive test cases covering all identified functionalities. These test cases should include steps to replicate user actions, expected outcomes, and acceptance criteria.
3. **Test Execution:** Execute the test cases manually or using automated testing tools. Test authentication mechanisms, CRUD operations, user functionalities, and payment integration thoroughly.
4. **Requirement Review and Error Uncovering**: Begin by reviewing the content model, interface model, and design model of the WebApp to uncover any errors or inconsistencies. Ensure that all use cases are accommodated in the interface model.
5. **User Interface Testing**: Conduct thorough testing of the user interface to identify errors in presentation and navigation mechanics. Verify that the interface is intuitive and user-friendly.
6. **Unit Testing**: Perform unit testing for each functional component of the WebApp to ensure that individual components work as expected.
7. **Navigation Testing**: Test the navigation throughout the architecture of the WebApp to uncover any navigation errors or inconsistencies.
8. **Compatibility Testing**: Implement the WebApp in various environmental configurations and test its compatibility with each configuration. Ensure that the WebApp functions correctly across different browsers, devices, and operating systems.
9. **Security Testing**: Conduct security tests to identify and exploit vulnerabilities in the WebApp or its environment. Implement measures to address any security vulnerabilities uncovered during testing.
10. **Performance Testing**: Evaluate the performance of the WebApp by conducting performance tests. Measure factors such as response times, scalability, and resource utilization.
11. **User Testing**: Engage a controlled and monitored population of end users to test the WebApp. Evaluate their interactions with the system for content errors, navigation issues, usability concerns, compatibility issues, and overall reliability and performance.
12. **Regression Testing:** After implementing any changes or updates, perform regression testing to ensure that existing functionalities have not been affected.
13. **Bug Reporting and Fix Verification:** Document any defects found during testing and communicate them to the development team for resolution. Verify fixes before closing the reported issues.

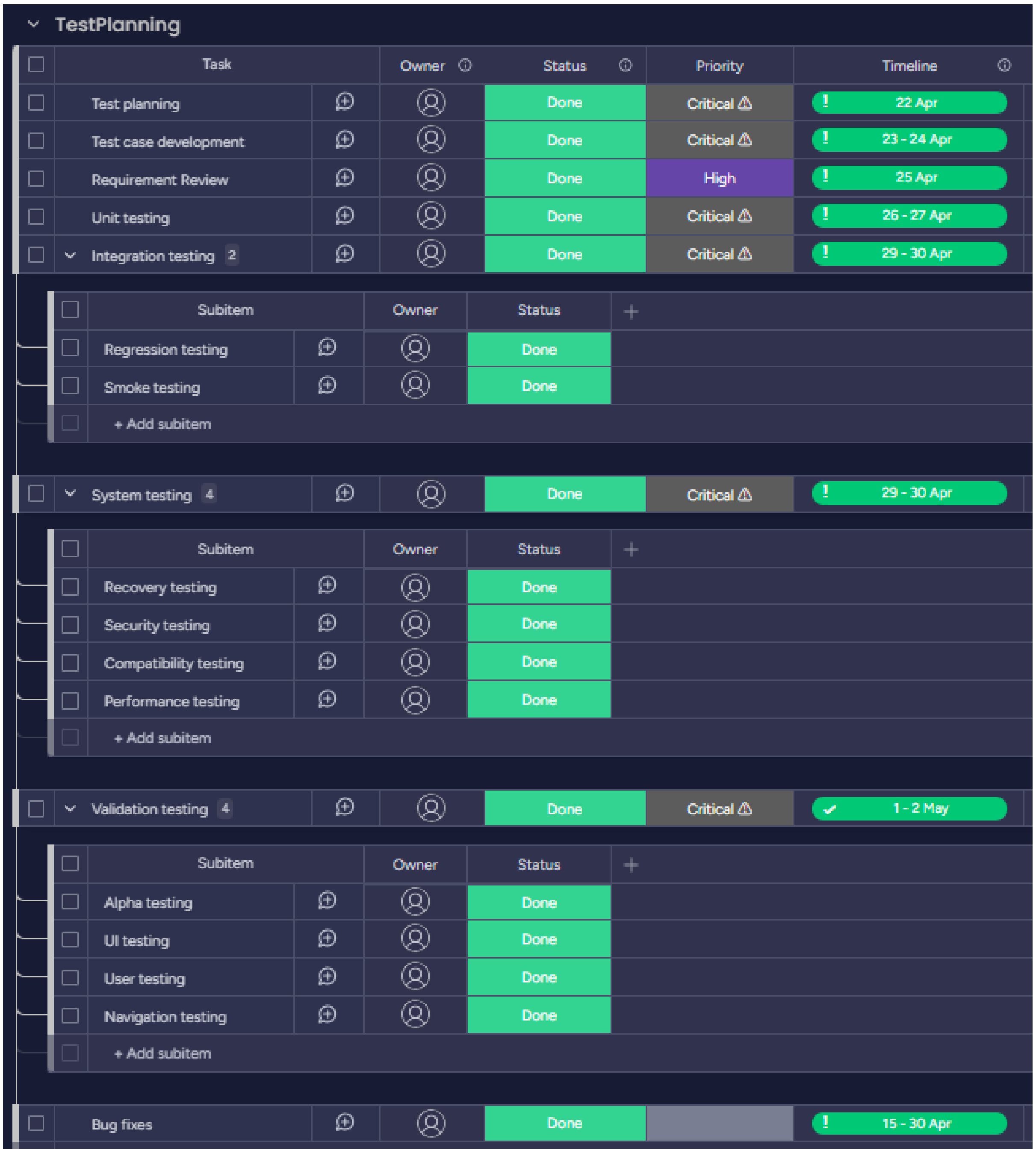
Testing Resources:

- **Testers**: Skilled QA engineers with expertise in manual and automated testing methodologies.
- **Testing Tools**: Utilize tools like *Postman* to test the API calls to the backend. Utilized *CodeQL* for static code analysis, enabling the detection of security vulnerabilities and potential risks within the codebase.
- **Devices and Platforms**: Access to a variety of devices, browsers, and operating systems to cover a wide range of user environments.
- **Test Data**: Prepare realistic test data sets to simulate different user scenarios and edge cases.
- **Test Environment**: Set up dedicated testing environments that replicate the production environment to ensure accurate testing.

Testing Schedule:



TestPlanning	
Bug fixes	15 - 30 Apr
Test planning	22 Apr
Test case development	23 - 24 Apr
Requirement Review	25 Apr
Unit testing	26 - 27 Apr
Integration testing	29 - 30 Apr
System testing	29 - 30 Apr
Validation testing	1 - 2 May



TestPlanning

Task	Owner	Status	Priority	Timeline
Test planning		Done	Critical ⚠	! 22 Apr
Test case development		Done	Critical ⚠	! 23 - 24 Apr
Requirement Review		Done	High	! 25 Apr
Unit testing		Done	Critical ⚠	! 26 - 27 Apr
Integration testing 2		Done	Critical ⚠	! 29 - 30 Apr

Subitem	Owner	Status
Regression testing		Done
Smoke testing		Done
+ Add subitem		

Task	Owner	Status	Priority	Timeline
System testing 4		Done	Critical ⚠	! 29 - 30 Apr

Subitem	Owner	Status
Recovery testing		Done
Security testing		Done
Compatibility testing		Done
Performance testing		Done
+ Add subitem		

Task	Owner	Status	Priority	Timeline
Validation testing 4		Done	Critical ⚠	✓ 1 - 2 May

Subitem	Owner	Status
Alpha testing		Done
UI testing		Done
User testing		Done
Navigation testing		Done
+ Add subitem		

Task	Owner	Status	Priority	Timeline
Bug fixes		Done		! 15 - 30 Apr