Student’s Name: Jimmy Aubert

Instructor: Harlan Brewer

Course: D778

Date: 07/24/2025

#Project (Software engineer)

##Part I: Software Development Methodology of the E-commerce Web Application Software Development Proposal

###A. Agile and Waterfall Methodologies: A Comparison

####1. Activities, Phases, and Deliverables

Methodology of Waterfall a method that is consecutive and linear, with each step ending before the next one starts.

Activities and Phases:

Requirement analysis: List all of the needs, such as SRS for product browsing and user accounts.

System Design: Produce technical blueprints (UI wireframes, architecture).

Implementation: Write code for the program.

Unit, integration, and system testing should be carried out.

Deployment: Open the software.

Maintenance: Resolve issues after deployment.

Deliverables include the deployed application, source code, design documentation, test results, and SRS.

Short cycles, or sprints, are used in the iterative Agile Methodology (Scrum) to promote flexibility and teamwork.

Activities and Phases:

Plan your sprints by choosing user stories for two to four weeks.

Development: Add features (such secure payments) one at a time.

Monitor progress and address obstacles at daily standups.

Sprint Review: Present features to interested parties.

Retrospective: Determine areas for process improvement.

Product backlogs, functional software updates, and sprint reports are examples of deliverables.

2. Possibilities and Drawbacks

Waterfall's advantages

Clear structures are appropriate for certain needs, such as payment integration (Pressman, 2014).

Thorough documentation facilitates safe payment compliance.

A predictable schedule helps keep set budgets in place.

Waterfall's shortcomings unable to adapt to new payment methods, for example.

Costly fixes for features like order tracking are at risk due to late testing.

Insufficient stakeholder input causes UI/UX alignment to be delayed.

Agile (Scrum) Advantages:

Schwaber and Sutherland (2020) demonstrate that flexible sprints respond to alternative demands, which may include product browsing. Early feedback satisfies the stakeholders

### B. The Agile Scrum Justification

Scrum is advised for the e-commerce web application as it aligns with important project components:

The SRS covers a wide range of criteria, such as secure payments and user accounts. Iterative backlog refinement in Scrum enables prioritizing and change adaption, including the addition of new product categories (Schwaber and Sutherland 9).

\*\*Complexity\*\*: The project calls for intricate connections, such as logistics APIs and payment gateways. With regular testing to identify problems early, Scrum's incremental methodology divides development into manageable jobs.

The two- to four-week sprints of Scrum meet the requirement for quick market launch by allowing features like product browsing for user testing to be delivered early.

\*\*Budget\*\*: Scrum gives priority to high-value features, keeping costs under control and permitting scope modifications if necessary.

\*\*Risks\*\* - Unlike waterfall testing, which delays testing, iterative testing and stakeholder evaluations reduce the likelihood of misaligned features or unreported flaws.

Stakeholder expectations are that the application will fulfill business and consumer requirements for a smooth shopping experience via regular sprint reviews.

This dynamic e-commerce project is a perfect fit for Scrum because of its flexibility and emphasis on incremental delivery (Schwaber and Sutherland 10).

### C. Applying Scrum to SRS Requirements

\*\*1. Section 3.1 presents the functional requirements. \*\*FR1: User Account Management\*\* (Users must create accounts, log in, and maintain their profiles):

Sprint Planning: Sort the user stories in the list by how important they are for identification.   
Create a safe OAuth-based login system that works with a database in one sprint.   
Sprint Review: Show off features and make changes based on feedback from stakeholders on how well they work.

Product Browsing (FR2): Browse by category, search, or filter:

Create user stories for the browsing and search functions as part of the \*\*Sprint Planning\*\* phase.

Construct front-end user interfaces and back-end search algorithms.

Verify search performance and filter usability with stakeholders during the \*\*Sprint Review\*\* phase.

\*\*FR3: Secure Payments\*\* (Secure Payment Processing):

Make plans to integrate with a payment gateway such as Stripe.

PCI-compliant payment processing should be implemented and tested.

Verify the user experience and transaction security using a \*\*Sprint Review\*\*.

2. Non-Functional Needs (Section 3.2) The performance of \*\*NFR1: Handle 1,000 concurrent users, <2s response):

It is recommended that performance testing be included into every sprint.

\*\*Development\*\*: Make caching and database queries more efficient.

To confirm response times, do load testing as part of the \*\*Sprint Review\*\*.

In order to accommodate traffic surges during sales, NFR2: Scalability:

Plan the integration of cloud infrastructure, such as AWS, using \*\*Sprint Planning\*\*.

\*\*Development\*\*: In a sprint, put auto-scaling functionality into place.

\*\*Retrospective\*\*: Evaluate scalability and make plans for enhancements.

The GDPR-compliant data protection, or \*\*NFR3: Security\*\*:

Make encryption and safe storage your top priorities when it comes to sprint planning.

The integration of data encryption with HTTPS is a development.

Verify compliance with auditors and stakeholders via a \*\*Sprint Review\*\*.

Section 4: Requirements for External Interfaces \*\*- \*\*EIR1: Integration of Payment Gateways\*\* (Incorporate third-party payment APIs):

\*\*Sprint Planning\*\*: Establish user stories for integrating APIs (like PayPal).

\*\*Development\*\* - Construct and evaluate API connection.

Check the success rates of transactions using the \*\*Sprint Review\*\*.

Support for mobile browsers is provided by \*\*EIR2: Mobile Responsiveness\*\*:

Assemble duties related to responsive user interface design in advance.

For a user interface that is responsive to mobile devices, use Bootstrap.

Test the user interface (UI) on mobile devices and take comments into consideration.

Integration of logistics APIs (EIR3: Order Tracking API):

Prioritize logistics API integration in your sprint planning.

\*\*Progress\*\*: Construct and evaluate the tracking feature.

Confirm functionality with stakeholders during the \*\*Sprint Review\*\* phase.

## Section II: Project Administration and Tracking

### D. Important Metrics to Track Development

In order to ensure that features like product browsing are delivered on schedule, \*\*1. Metrics and Project Factors\*\* - \*\*Sprint Velocity\*\* (Timeline): Tracks story points achieved every sprint.

A defect's density (Quality/Scope): Monitors flaws for each functionality (like payments), guaranteeing SRS adherence.

\*\*Budget Burn Rate\*\* (Budget): Tracks weekly expenditures and manages expenses for resources such as cloud services.

\*\*Healthy Backlog\*\* (Scope): Maintains a record of finished user stories to make sure they fit the project scope.

2. Making Use of Metrics

Project health is tracked by these measures taken together. Burn rate regulates expenses, backlog health synchronizes scope with objectives, velocity guarantees timetable adherence, and defect density preserves quality. Sprint reviews use these metrics to modify priorities, guaranteeing that the project is completed on schedule and under budget while meeting SRS objectives (Schwaber and Sutherland 12).

### E. Strategy for Stakeholder Communication

Asynchronous Collaboration: Utilize Slack for real-time updates and Jira for backlog management to enable stakeholders to monitor progress at any time.

The biweekly sprint reports, which include velocity, accomplished stories (such secure payments), and forthcoming goals, including budget and defect updates, are sent via email.

Maintain a Jira dashboard that shows velocity, backlog health, and feature completion in order to monitor alignment with SRS targets, such mobile responsiveness.

### F. Managing Risk  
#### 1. Anticipated Risks

- \*\*Scope Creep\*\*: Extra feature requests (e.g., analytics) may delay timeline and raise costs.

- \*\*Integration Delays\*\*: Third-party API issues (e.g., payment) may impact schedule/scope.

#### 2. Proactive Measures

- \*\*Scope Creep\*\*: Use a change control board to assess new requests’ budget/timeline impact.

- \*\*Integration Delays\*\*: Test APIs in sprint 0, allocate buffer sprints, and liaise with vendors.

Works Cited

Pressman, Roger S. \*Software Engineering: A Practitioner’s Approach\*. 8th ed., McGraw-Hill Education, 2014.

Schwaber, Ken, and Jeff Sutherland. \*The Scrum Guide\*. Scrum.org, Nov. 2020, [www.scrumguides.org/scrum-guide.html](http://www.scrumguides.org/scrum-guide.html).