Design Thinking for Software Engineers

Lab-4

Online Learning Platform

Name: **Nivedh Sunil** **Roll No.:** **24BTCE111**

1. Aim of the Experiment

To analyse and compare the Agile and Waterfall software development models by applying them to the development of an Online Learning Platform and to study the project risks involved in designing, implementing and maintaining such a platform.

2. Scenario Description

The Online Learning Platform is a large-scale digital education system designed to deliver interactive courses, manage student learning and support instructors through an integrated web-based ecosystem. The platform enables real-time access to educational resources, assessments and performance analytics.

The system consists of:

- Web portal for students to access courses, assignments and progress tracking
- Instructor dashboard for course creation, grading and student management
- Mobile application for on-the-go learning and notifications
- Analytics and reporting module for tracking student performance and engagement

3. Identified Requirements

Functional Requirements

1. User registration and role-based login for students, instructors and administrators
2. Upload and manage courses, lectures and learning materials
3. Conduct online assessments, quizzes and assignment submissions
4. Provide real-time notifications for deadlines, announcements and updates
5. Generate progress reports and performance analytics for students and instructors

Non-Functional Requirements

1. Scalability to support a large number of concurrent users
2. High system availability and reliability for uninterrupted learning
3. Secure storage and transmission of user data
4. Fast system response and smooth performance during peak usage

4. Application of the Waterfall Model to the Scenario

Waterfall model Phase-wise Execution

- Requirement Analysis
  - All system requirements such as user management, course delivery, assessments, notifications and reporting are collected and documented at the beginning.
  - No major changes are allowed once requirements are finalized.
- System Design
  - Complete architecture including database structure, web platform, mobile application and analytics modules is designed upfront.
  - User interface and system workflows are finalized before development begins.
- Implementation
  - Coding of all modules (student portal, instructor dashboard, assessment system and reporting tools) is done strictly according to the design documents.
- Testing
  - System testing is carried out after full development is completed.
  - Integration issues between modules such as course delivery and assessment systems are identified at this stage.
- Deployment and Maintenance
  - The platform is deployed for all users after successful testing.
  - Maintenance focuses on bug fixes and performance improvements.

Observations Using the Waterfall Model

- Changes such as adding new learning features or interactive tools are difficult to incorporate after development begins.
- Usability and performance issues are often discovered late in the process.
- Cost and development time increase when defects are detected in later stages.

5. Application of the Agile Model to the Scenario

Agile software development-based Execution

- Product Backlog Creation
  - The initial backlog includes core features such as user registration, course management and basic content delivery.
  - Advanced features like AI-based personalized learning recommendations and advanced analytics are added progressively.
- Sprint-wise Development
  - Sprint 1: User authentication system and basic student course access
  - Sprint 2: Course upload and instructor dashboard features
  - Sprint 3: Online assessments, quizzes and assignment submission

    - Sprint 4: Notifications, performance tracking and analytics dashboard
    - Sprint 5: Mobile optimization and advanced personalization features
- Continuous Feedback
    - Students and instructors review features at the end of each sprint.
    - Requirements are refined based on usability feedback and learning outcomes.
- Continuous Testing
    - Testing is integrated into every sprint.
    - Performance, security and usability issues are detected and fixed early.

Observations Using Agile

- Requirement changes are incorporated smoothly during development.
- A functional version of the platform is delivered early and improved incrementally.
- Project risks are reduced due to iterative testing and continuous stakeholder feedback.

6. Risk Analysis

Risk Identified	Impact in Waterfall Model	Impact in Agile Model
Requirement Changes	High	Low
System Integration Failure	Detected Late	Detected Early
Scalability Issues	Post Deployment	During Iterations
User Acceptance Risk	High	Low
Cost Overrun	High	Low

7. Comparative Analysis

Parameter	Waterfall Model	Agile Model
Requirement Flexibility	Low	High
Change Handling	Difficult	Easy
Risk Detection	Late	Early
Stakeholder Involvement	Minimal	Continuous
Suitability for Scenario	Less Suitable	Highly Suitable

8. Result

In the Online Learning Platform scenario, the Agile software development model is clearly the better choice. The platform depends on evolving requirements, frequent feature upgrades and constant feedback from students and instructors. Agile supports iterative releases, early risk detection and rapid adaptation to changing educational needs, which are critical for keeping the system relevant and usable.

The Waterfall model would be better only if the Online Learning Platform had clear, fixed requirements from the beginning and no major changes were expected. It works best when everything is planned in detail upfront and the project follows a strict, step-by-step process.

9. Conclusion

This experiment concludes that for an Online Learning Platform involving dynamic requirements, interactive user engagement and ongoing enhancement, the Agile development model provides superior flexibility, lower project risk and higher overall system quality compared to the Waterfall model.