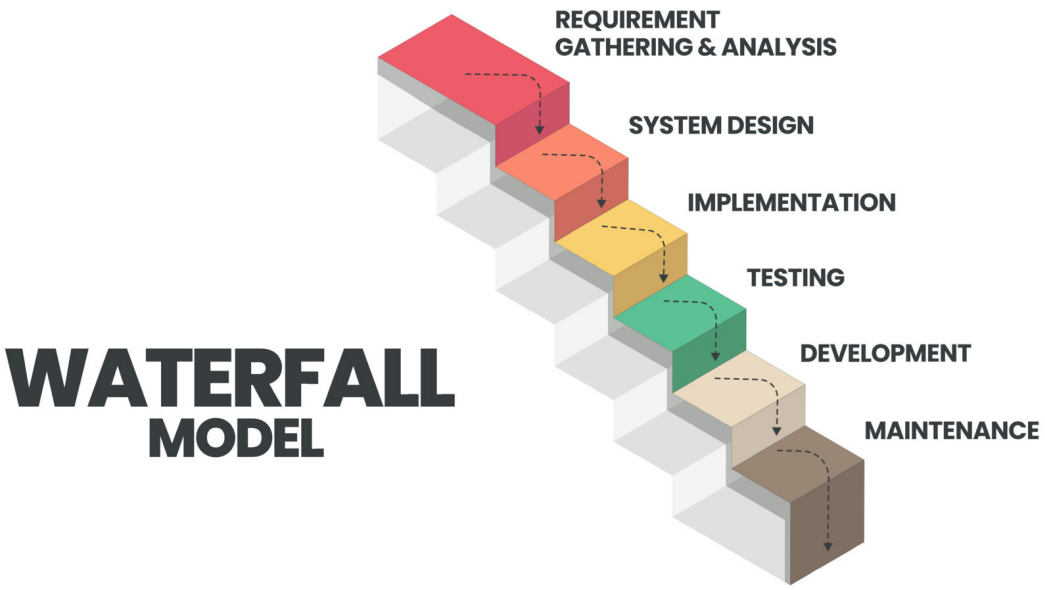
DHRUV SHETTY
121A1101 BATCH: D2

EXPERIMENT 1

Aim: Application of at least two traditional process models.

Waterfall Model



The Waterfall Model is a traditional software development methodology characterized by a linear and sequential approach to project management. While it may not be the most suitable model for all types of projects, it can still be adapted for the "Course Recommendation System" project.

1. Requirements Gathering:

- Define the scope and objectives of your course recommendation system.
- Identify the key stakeholders and gather their requirements.

- Create a detailed requirements document that outlines the system's features, user roles, and constraints.

2. System Design:

- Develop a high-level architecture for your course recommendation system.
- Create detailed design specifications for each component of the system, including the database structure, user interfaces, and algorithms.
- Review and validate the design with stakeholders to ensure it aligns with their requirements.

3. Implementation:

- Begin the actual development of the system based on the approved design.
- Code the various modules and components, adhering to coding standards and best practices.
- Conduct regular code reviews and testing to catch and fix any defects early in the process.

4. Testing:

- Develop a comprehensive testing plan that covers unit testing, integration testing, system testing, and user acceptance testing.
- Execute the testing plan, identify and document any defects or issues, and work on their resolution.
- Ensure that the system meets the specified requirements and is free from critical defects.

5. Deployment:

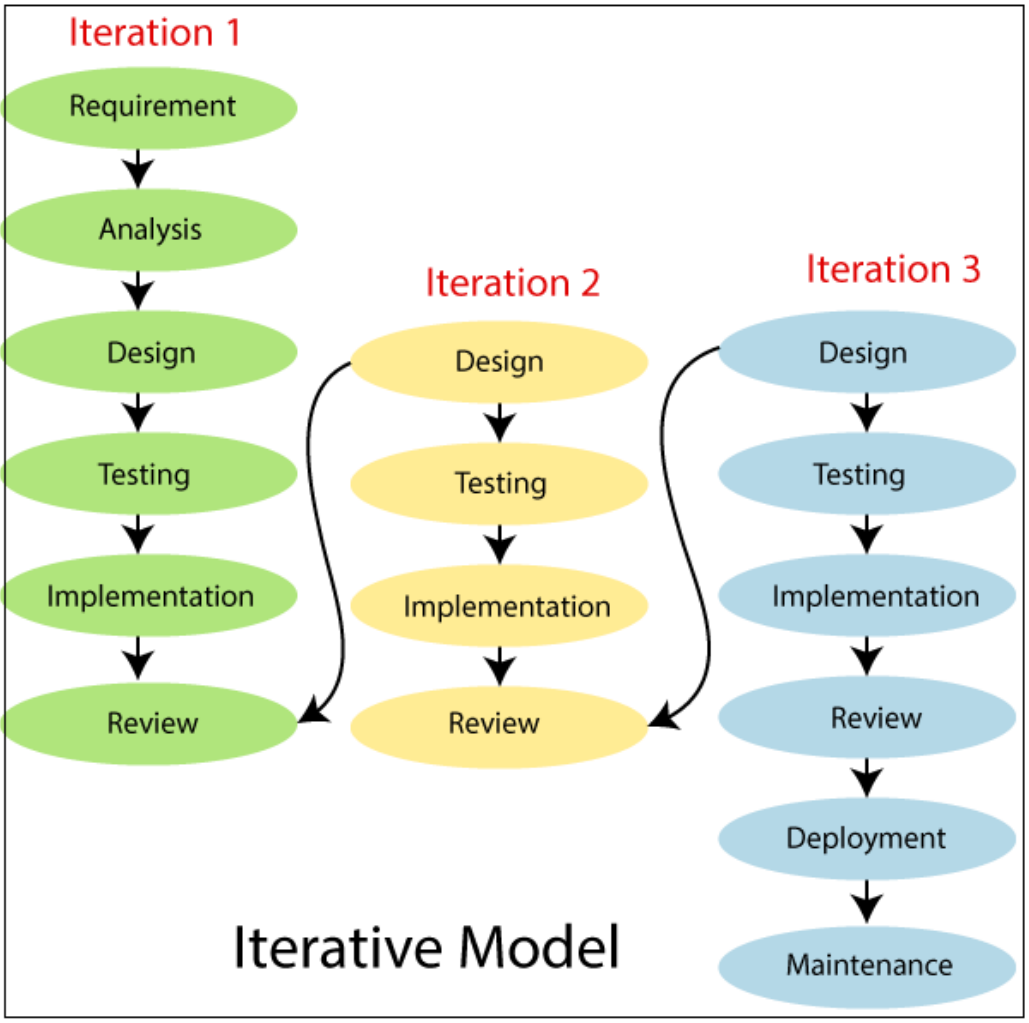
- Prepare for the deployment of your course recommendation system in a production environment.

- Create user documentation and training materials if necessary.
- Deploy the system and monitor its performance during the initial phase.

6. Maintenance and Support:

- Provide ongoing maintenance and support to address any issues that arise in the production environment.
- Enhance the system by adding new features or making improvements based on user feedback and changing requirements.

Iterative Model



In the Iterative model, the iterative process starts with a simple implementation of a small set of the software requirements and iteratively enhances the evolving versions until the complete system is implemented and ready to be deployed. Creating an iterative model for a Course Recommendation System involves several stages of development and refinement.

Iteration 1: Data Collection and Preprocessing

1. Data Gathering: Start by collecting data about courses, including course descriptions, prerequisites, ratings, and user profiles (if available). You can scrape data from educational websites, use APIs, or partner with educational institutions.
2. Data Cleaning: Clean and preprocess the data to handle missing values, outliers, and inconsistencies.

Iteration 2: Basic Recommendation System

3. Content-Based Filtering: Implement a basic recommendation system using content-based filtering. Calculate course similarities based on attributes like course descriptions, prerequisites, and categories.
4. User-Item Matrix: Create a user-item interaction matrix to capture user preferences and behavior.
5. Recommendation Engine: Build a simple recommendation engine that suggests courses based on user profiles and the content-based filtering approach.

Iteration 3: Collaborative Filtering

6. Collaborative Filtering: Incorporate collaborative filtering techniques (user-based or item-based) to improve recommendations. Use user-item interaction data to identify user behavior patterns.

Iteration 4: Model Evaluation and Metrics

7. Evaluation Metrics: Define evaluation metrics like precision, Lines of Code (LOC), Effort and Mean Average Precision (MAP) to measure the performance of your recommendation system. Split your data into training and testing sets.
8. Model Evaluation: Evaluate your recommendation models using the defined metrics and iterate on your algorithms to improve performance.

Iteration 5: Personalization and User Feedback

10. User Feedback: Implement mechanisms to gather user feedback, such as ratings and reviews, to further enhance the recommendation system's accuracy.
11. Personalization: Incorporate personalization features to tailor recommendations to individual user preferences and behavior.

Iteration 6: Scalability and Deployment

12. Scalability: Optimize your recommendation system for scalability, as more data and users are added. Consider using distributed computing or cloud-based solutions if necessary.
13. Deployment: Deploy your recommendation system as a web application or integrate it into an existing platform. Ensure it's user-friendly and accessible.

Iteration 7: Monitoring and Maintenance

14. Monitoring and Maintenance: Set up monitoring tools to track system performance and user engagement. Regularly update your recommendation models and retrain them with new data.