**EXPERIMENT-7**

**AIM: Software Development scenario using FDD**

**theory :**

Feature-Driven Development (FDD) is an iterative and incremental software development methodology that focuses on building software features incrementally. It involves breaking down the software development process into manageable features, each with its own design, implementation, and testing phases.

**Here's a simplified outline of how FDD can be applied in software development, including the coding aspect:**

1.Develop an Overall Model: Start by creating an overall model of the system. Identify the major features and their dependencies. This model will serve as a roadmap for the development process.

2.Feature List Creation: Break down the system into smaller, manageable features. These features should be prioritized based on business value and dependencies.

3.Plan by Feature: For each feature, create a plan that includes design, coding, and testing. This involves creating feature-specific task lists, identifying the resources needed, and estimating the time required.

4.Design by Feature: Focus on designing the architecture and details for each feature. This can involve creating class diagrams, sequence diagrams, and other design artifacts specific to the feature.

5.Build by Feature: Coding takes place based on the design for each feature. Developers work on implementing the features based on the requirements and designs outlined earlier.

6.Feature Inspection: After the initial implementation, conduct inspections to ensure that the feature meets the requirements and quality standards. This involves reviewing the code, functionality, and any other relevant aspects.

7.Build and Test Incrementally: As features are completed, integrate them into the overall system and conduct testing. This incremental approach helps in identifying and resolving issues early.

8.Repeat and Refine: Continue this process iteratively, working through each feature until the entire system is implemented.

**When it comes specifically to coding within FDD:**

1.Collaborative Coding: Encourage collaboration among developers working on different features. They should follow coding standards and practices defined by the team.

2.Code Ownership: Assign ownership of specific modules or features to developers. This helps maintain accountability and ensures better knowledge sharing within the team.

3.Regular Integration: Ensure continuous integration of code from different features to maintain a cohesive codebase. This involves using version control systems and automated build tools.

4.Testing and Quality Assurance: Incorporate testing throughout the development process. Developers should write unit tests to verify the functionality of their code, while dedicated QA engineers can perform broader tests on integrated features.

5.Communication and Reporting: Regularly communicate progress, challenges, and any impediments during feature development. Reporting mechanisms should be in place to track feature completion and overall project status.

**Conclusion:**

FDD emphasizes an iterative and feature-centric approach to software development, which can result in a more predictable and manageable development process.