**Test-Driven Development (TDD)**

Approach:

1. Write test cases first.

2. Write code to pass the tests.

3. Reactor code as needed.

Benefits:

- Early bug detection.

- Improved code quality.

- Guides design decisions.

Suitability:

- Agile environments.

- Projects with clear requirements.

**Behaviour-Driven Development (BDD)**

Approach:

1. Define behaviour using plain language.

2. Translate behaviour into testable specifications.

3. Write code to fulfil specifications.

Benefits:

- Collaboration between developers and non-technical stakeholders.

- Focus on business value.

- Enhanced communication.

Suitability:

- Complex projects with evolving requirements.

- Client-facing applications.

**Feature-Driven Development (FDD)**

Approach:

1. Develop features iteratively.

2. Emphasize domain object modelling.

3. Build features based on client priorities.

Benefits:

- Clear project progress tracking.

- Scalability for large projects.

- Strong emphasis on design.

Suitability:

- Large-scale enterprise projects.

- Teams with domain expertise.