Version Control Flow, Agile and Waterfall methodology of software development cycle

Class 11

05/4/2025

Acknowledgement

The series of the IT & Japanese language course is Supported by AOTS and OEC.

AOTS

Ministry of Economy, Trade and Industry

OVERSEAS EMPLOYMENT CORPORATION OEC

Overseas Employment Corporation

What you have Learnt Last Week

We were focused on following points.

- Usage of control and loop flow statement
- Performing Linear Algebra in Numpy
- Inspecting and Understanding Data
- Why Requirement Analysis is so important in the process?
- Review case studies that demonstrate successful requirement analysis practices
- Linear & Multi Linear Regression
- Support Vector Machine
- Cost Function

What you will Learn Today

We will focus on following points.

- Introduction of Gitflow, the version control system
- Difference between Agile and waterfall methodology in Japan
- Discuss software development life cycle and clean architecture
- Quiz
- Q&A Session

Introduction to Version Control

Understanding Version Control

What is Version Control?

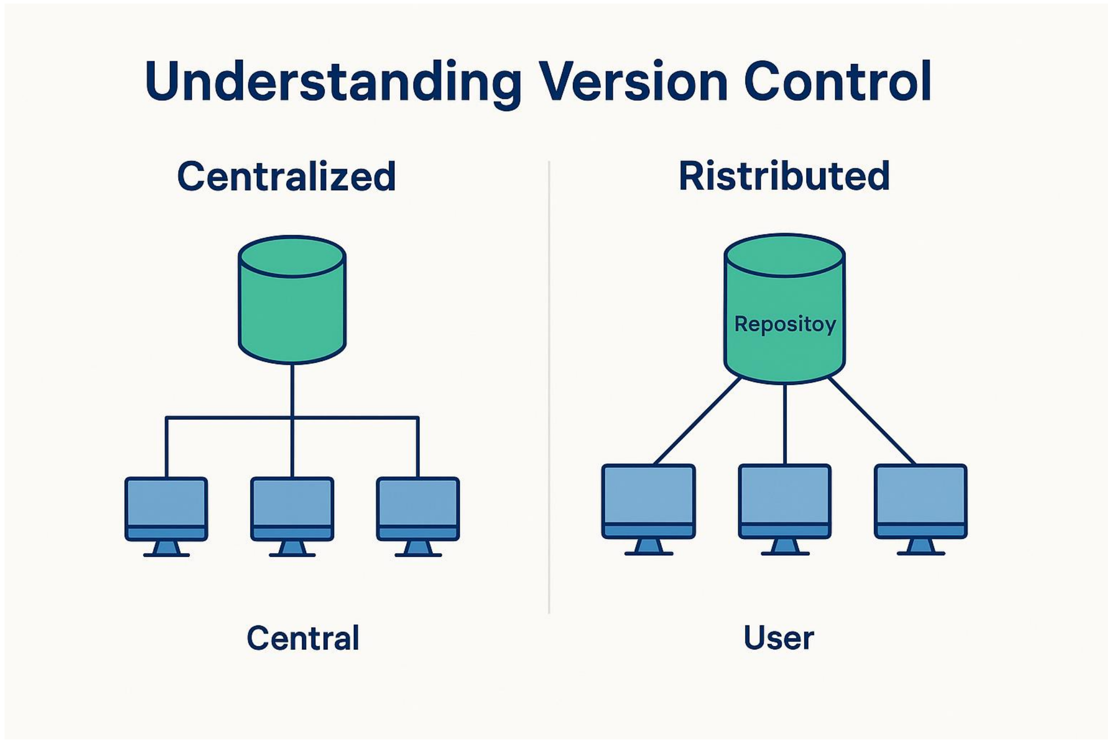
1. A system that records changes to files over time.

2. Helps teams collaborate and track changes efficiently.

Types of Version Control Systems

1. **Centralized (CVCS)** – Single central repository (e.g., SVN, Perforce).

2. Distributed (DVCS) – Every user has a complete copy (e.g., Git, Mercurial).



Benefits of Version Control: Tracks history of changes, Enables collaboration, Supports branching and merging.

Introduction to Git

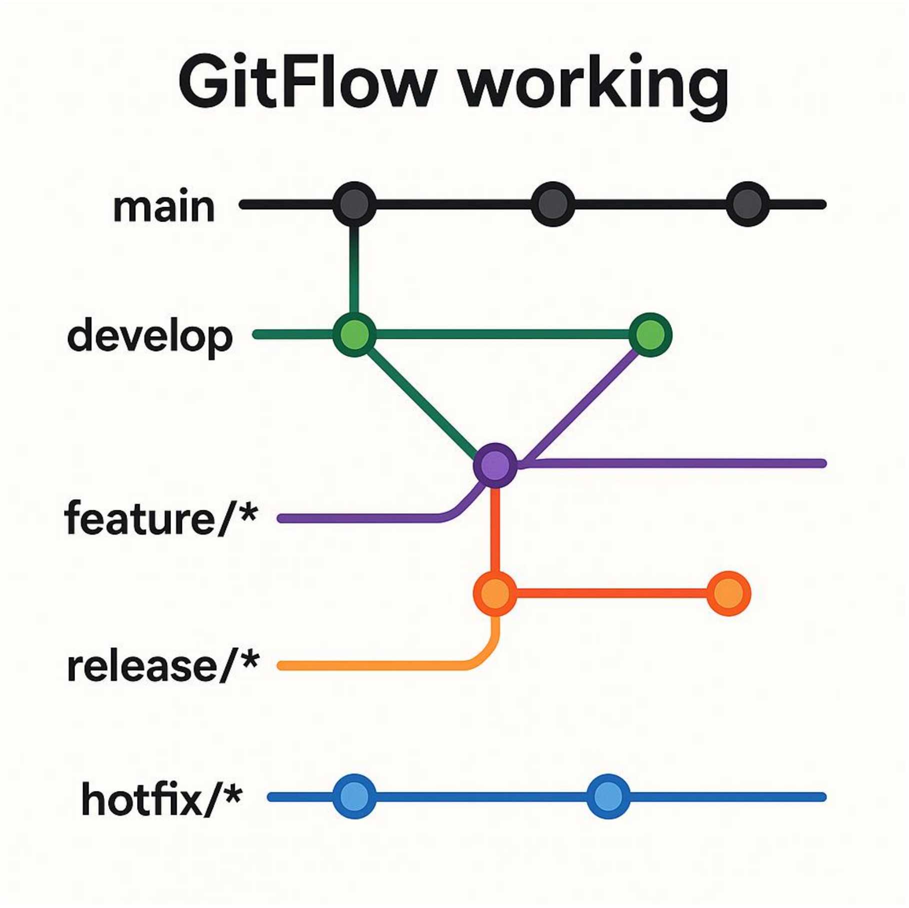
A distributed version control system used for tracking source code changes.

Key Features of Git:

1. Distributed system
2. Branching and merging
3. Fast performance
4. Secure and reliable

Basic Git Commands:

1. **git init** – Initialize the repository
2. **git clone** – Copy of an existing repo
3. **git commit –m "message"** – Save changes
4. **git push / git pull** – Sync with remote repo
5. **git branch / git merge** – Manage branches



What is Gitflow?

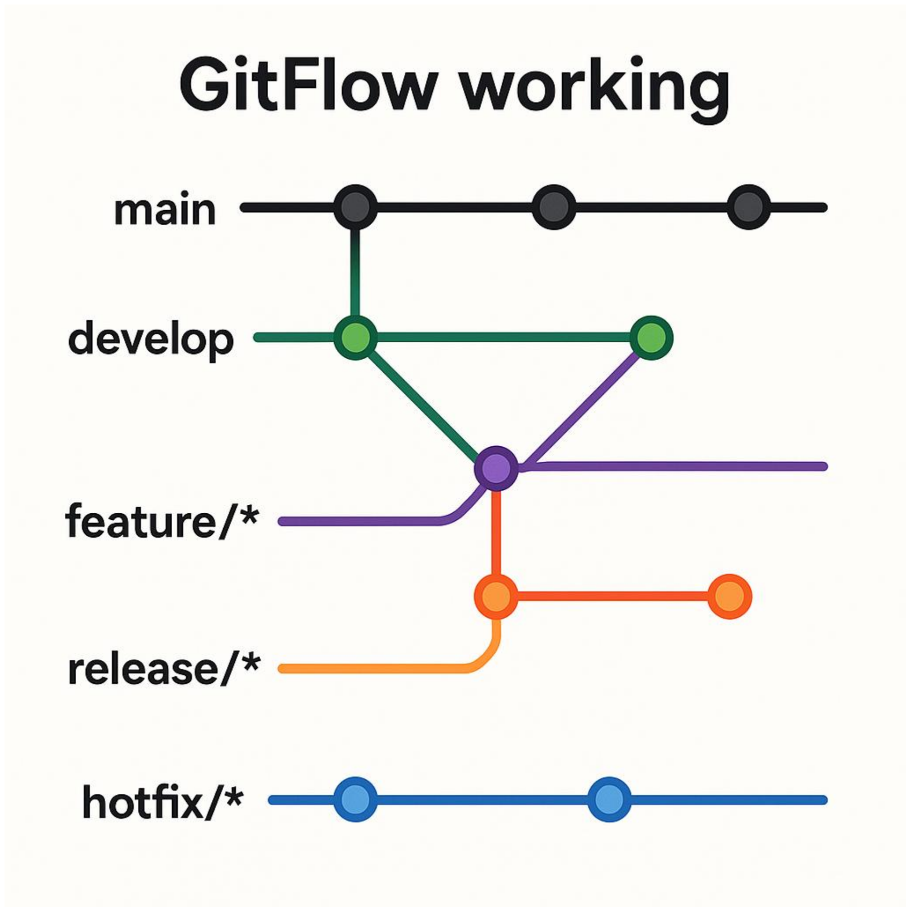
A branching model for Git that organizes development workflow.

Definition & Purpose of Gitflow

- A branching model for Git that organizes development workflow.
- Helps teams work on features, releases, and hotfixes efficiently.

Why Use Gitflow?

- Structured and scalable workflow.
- Ensures stable production releases.
- Enables parallel feature development.

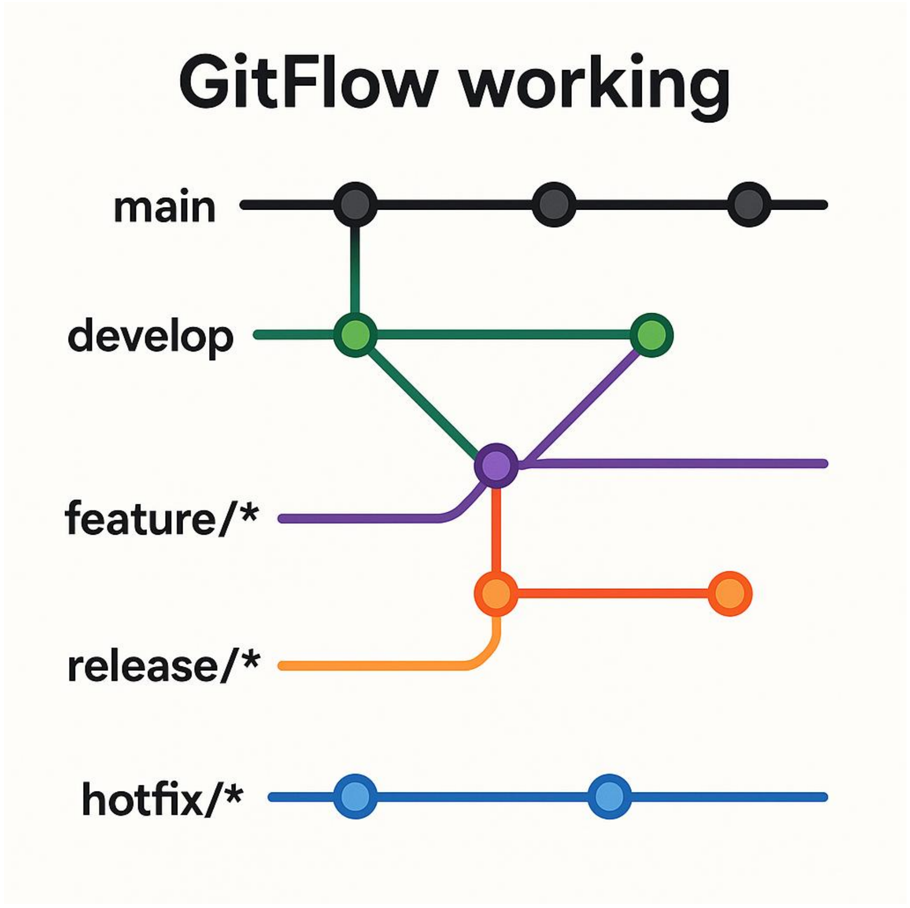


Gitflow Workflow

It is branching model for Git that defines a strict workflow for managing features, releases, and hotfixes in a more organized way

Gitflow Branching Strategy

- **Master:** Stable production ready branch
- **Develop:** Main branch for integration
- **Staging:** For testing before production
- **Feature:** New feature for development
- **Release:** Prepares for production
- **Hotfix:** Fixes critical bugs in production

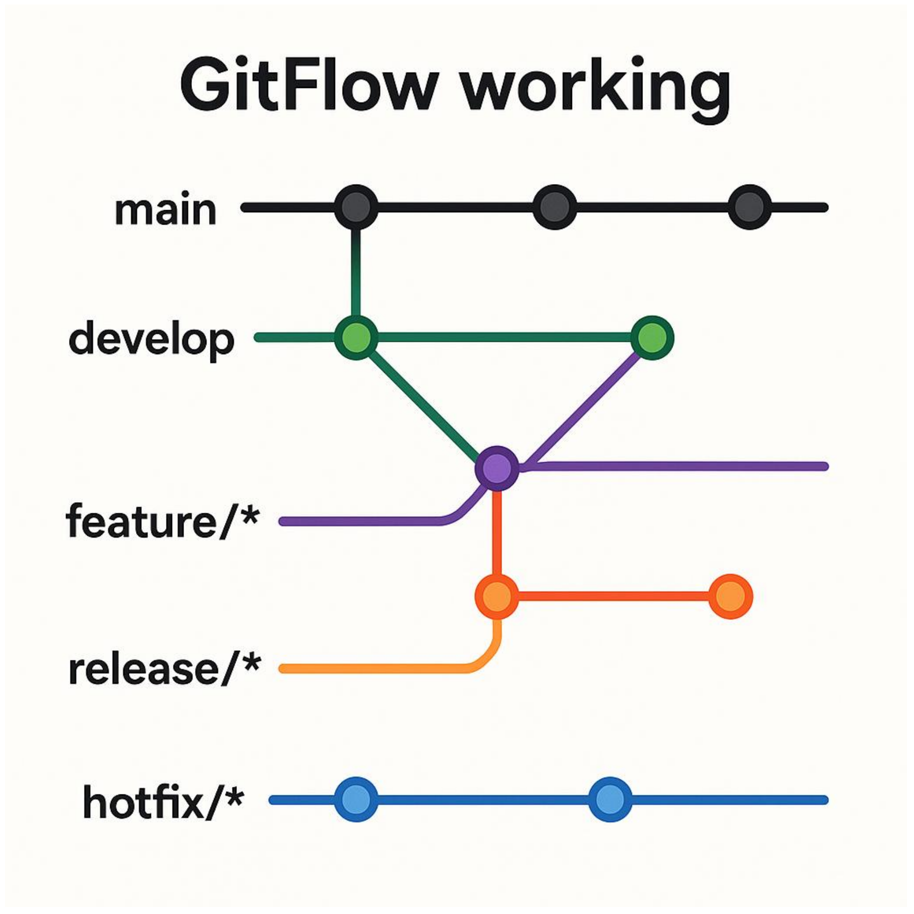


Steps of Gitflow Workflow

Steps include in Gitflow

Step-by-Step Process

- Create a feature branch (git flow feature start feature/notification)
- Merge into develop after completion
- Create a release branch for testing (git flow release start version)
- Merge release into Master and develop
- Fix urgent bugs using hotfix



Advantages & Challenges of Gitflow

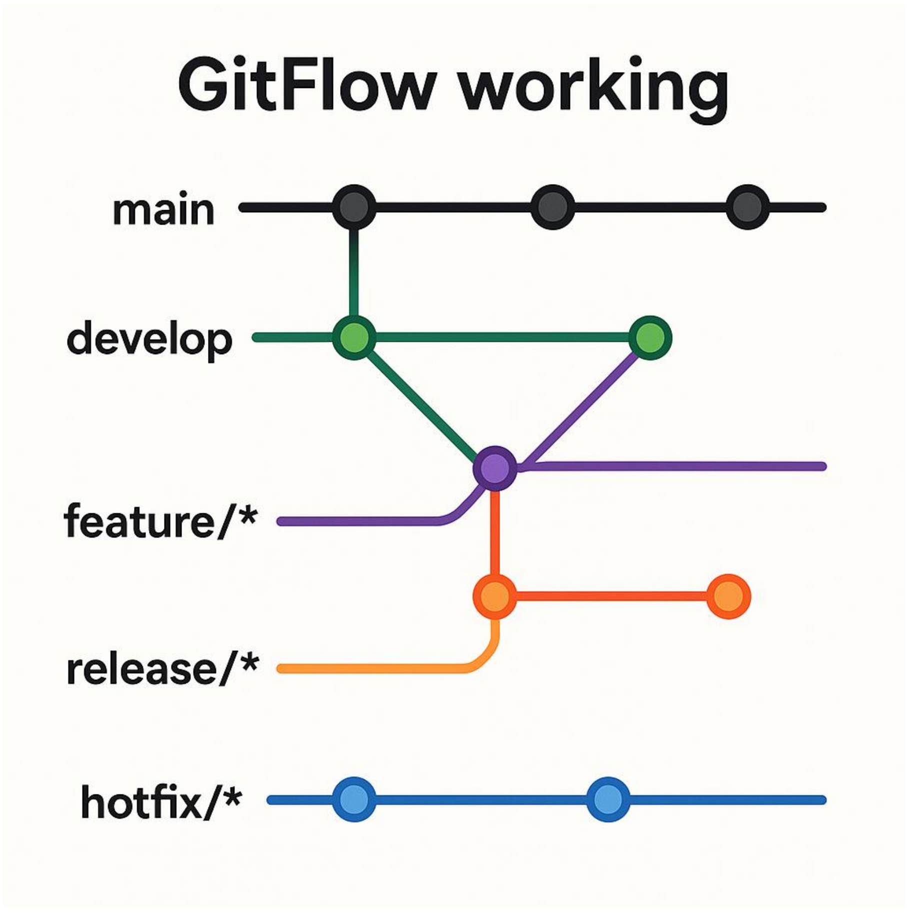
Advantages & Challenges of Gitflow

Advantages of Gitflow

- Organized development process.
- Allows parallel development.
- Better collaboration & version control.

Challenges of Gitflow

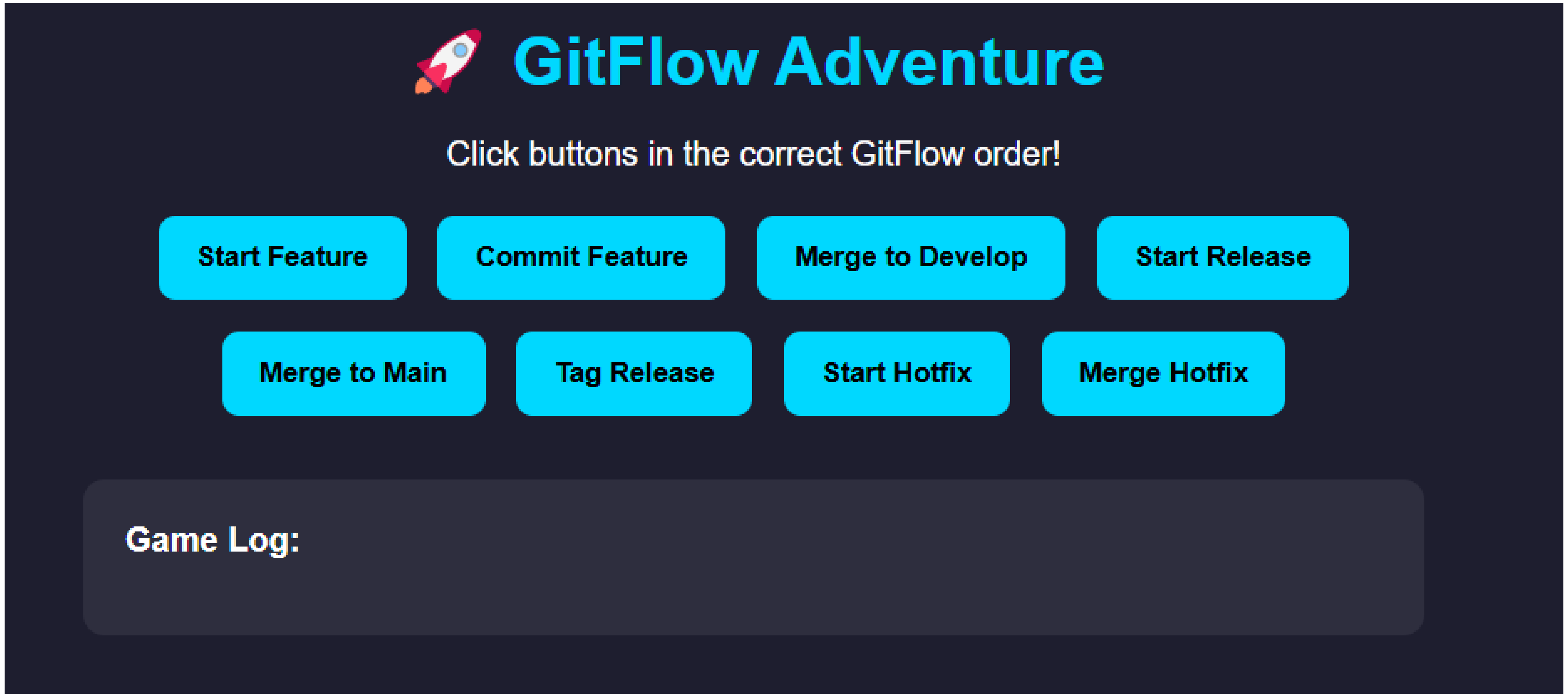
- Can be complex for small teams.
- Requires discipline in managing branches.
- Might slow down rapid release cycles.



Task-1

Gitflow Adventure

Let's learn the gitflow by clicking the buttons in correct order


🚀 GitFlow Adventure

Click buttons in the correct GitFlow order!

Introduction to Agile Methodology

It is a flexible, iterative approach to software development.

What is Agile?

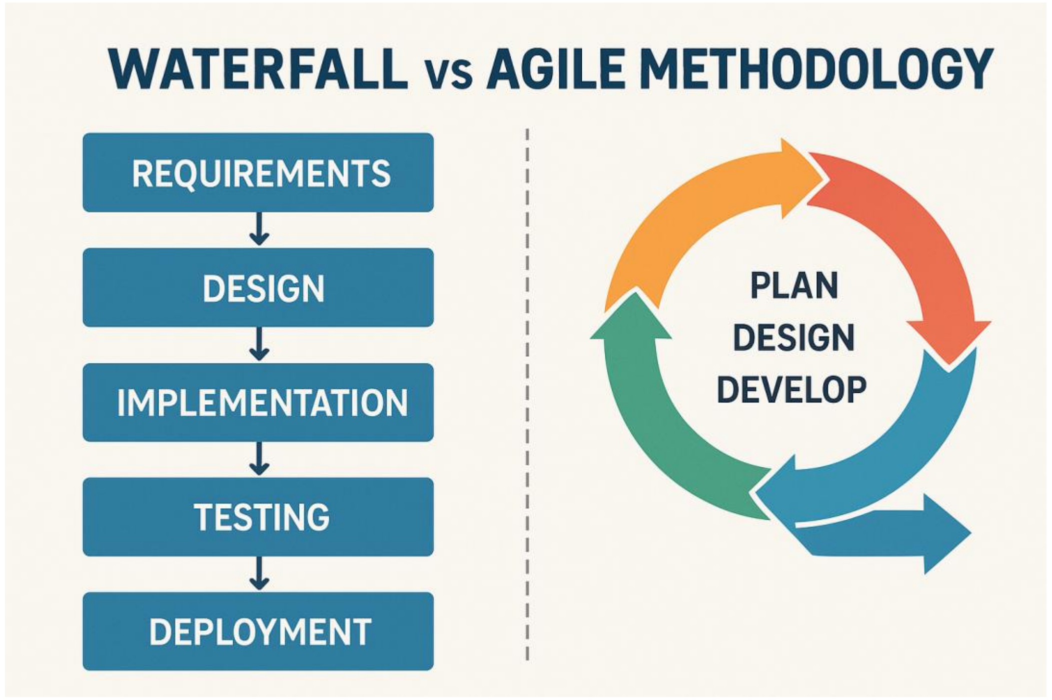
- A flexible, iterative approach to software development.
- Focuses on collaboration, customer feedback, and adaptability.

Core Principles of Agile (Agile Manifesto)

- Individuals & interactions over processes & tools.
- Working software over documentation.
- Customer collaboration over contract negotiation.
- Responding to change over following a plan.

Common Agile Frameworks

- **Scrum** – Sprints & daily stand-ups.
- **Kanban** – Visual task management.
- **SAFe** – Scaling Agile for enterprises.



Understanding Waterfall Methodology

Understanding of Waterfall Methodology

What is Waterfall?

- A linear, sequential software development approach.
- Each phase must be completed before moving to the next.

Phases of the Waterfall Model

- Requirements Gathering
- System Design
- Implementation (Coding)
- Testing
- Deployment
- Maintenece

Advantages and Limitations of Waterfall

✅ Structured and well-documented.
✅ Easier for fixed-scope projects.
⚠️ Less flexible to changes.
⚠️ Long development cycles.

Agile vs. Waterfall in Japan

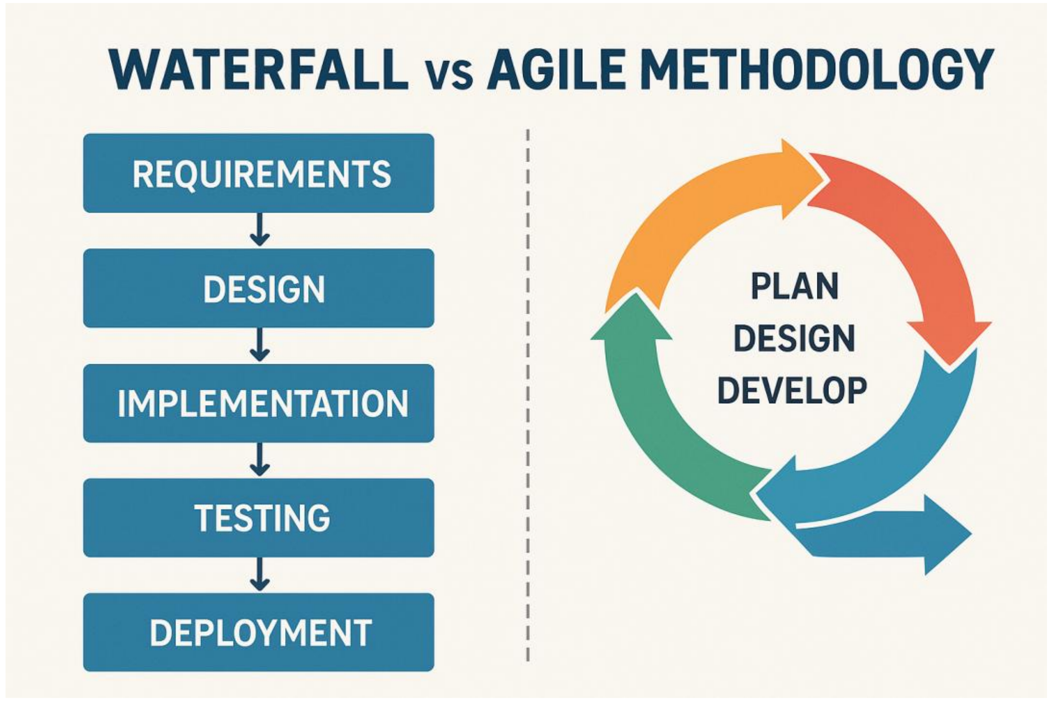
Adoption Trends & Cultural Influence

Adoption Trends in Japan

- Traditional companies favor Waterfall.
- Startups and modern IT firms lean towards Agile

Cultural Influence on Methodologies

- **Kaizen (Continuous Improvement):** Supports Agile practices.
- **Risk Avoidance Culture:** Prefers Waterfall for predictability.
- **Group Decision-Making:** Impacts Agile's iterative feedback loops.



[Note]

✅ **Agile** – Faster delivery, better adaptability.
✅ **Waterfall** – Clear documentation, predictable results.

Case Studies – Agile vs. Waterfall in Japan

Real-World Implementation

Agile Case Study (Japanese Tech Company)

- A Tokyo-based startup adopted Scrum for rapid software releases.
- Achieved a 30% faster development cycle.
- Improved team collaboration and client satisfaction.

Waterfall Case Study (Manufacturing Software)

- A large automotive company used Waterfall for ERP system development.
- Ensured stability and compliance with regulatory standards.
- Faced delays due to rigid phase transitions.

Choosing the Right Approach

Hybrid Models & Best Practices

Factors to Consider When Choosing Agile or Waterfall

- Project size & complexity.
- Industry regulations & compliance.
- Flexibility requirements.

Hybrid Models in Japan

- **Water-Scrum-Fall:** Uses Agile for development but maintains structured testing & deployment.
- **Lean-Agile:** Merges Lean Manufacturing principles with Agile methodologies.

Best Practices for Japanese Companies

- Align methodology with company culture.
- Train teams in Agile while retaining structured workflows.
- Combine Agile’s flexibility with Waterfall’s predictability.

Lean-Agile: The Fusion

When combined, Lean-Agile creates a powerful framework for modern organizations

Lean Principle	Agile Practice	Result
Eliminate waste	Minimize unnecessary features/tasks	Faster delivery, reduced cost
Deliver fast	Iterative development (sprints)	Frequent releases, fast feedback
Empower teams	Cross-functional, self-organizing teams	Better ownership and innovation
Optimize the whole	Agile release trains, value stream mapping	Aligned teams working toward outcomes
Continuous improvement	Retrospectives, metrics, Kaizen culture	Constant learning and adaptation

Introduction to Clean Architecture

A software architecture pattern that separates concerns and ensures independence from frameworks, databases, UI, or external agencies.

Key Principles:

- Separation of concerns
- **Dependency Rule** (Inner layers should not depend on outer layers)
- Independence from frameworks
- Testability

Benefits:

- Easy to test
- Easy to maintain and scale.
- Clear structure for teams

The inner layers are focused on **core business logic** and **application rules**

Layers of Clean Architecture

There are four main Layers

1. **Entities**
   - Business rules, domain objects
   - **Example:** user, order, product

2. **Use Cases**
   - Application-specific logic
   - **Example:** CreateOrder, LoginUser

3. **Interface Adapters**
   - Translate data between layers (Controllers, Gateways)
   - **Example:** OrderController, UserPresenter

4. **Frameworks & Drivers**
   - UI, DB, APIs (outermost layer)
   - **Example:** Reac_frontend, Postgre_SQL, REST API

Clean Architecture in SDLC

Fitting Clean Architecture into Software Development Life Cycle

SDLC Phases Integration

- **Requirements:** Define business rules (Entities)
- **Design:** Map to Use Cases and layers
- **Implementation:** Build with separation
- **Testing:** Test layers independently
- **Maintenance:** Easier updates without affecting core logic

Best Practices

- Start from the core (Entities → Outward)
- Use interfaces and DI (Dependency Injection)
- Keep frameworks out of business logic

The outer layer includes things like database access, web controllers, and API routes.

Real-World Examples & Use Cases

Real-World Examples & Use Cases

Example 1: E-commerce App

- **Entities:** Product, Cart
- **Use Cases:** Add to cart, Checkout
- **Testing:** Test layers independently
- **Frameworks:** Angular + Firebase

Example 2: Banking App

- **Entities:** Account, Transaction
- **Use Cases:** Transfer money, Generate report
- **Adapters:** CLI/GUI, Presenter
- **Frameworks:** Spring Boot + MySQL

Why It Works:

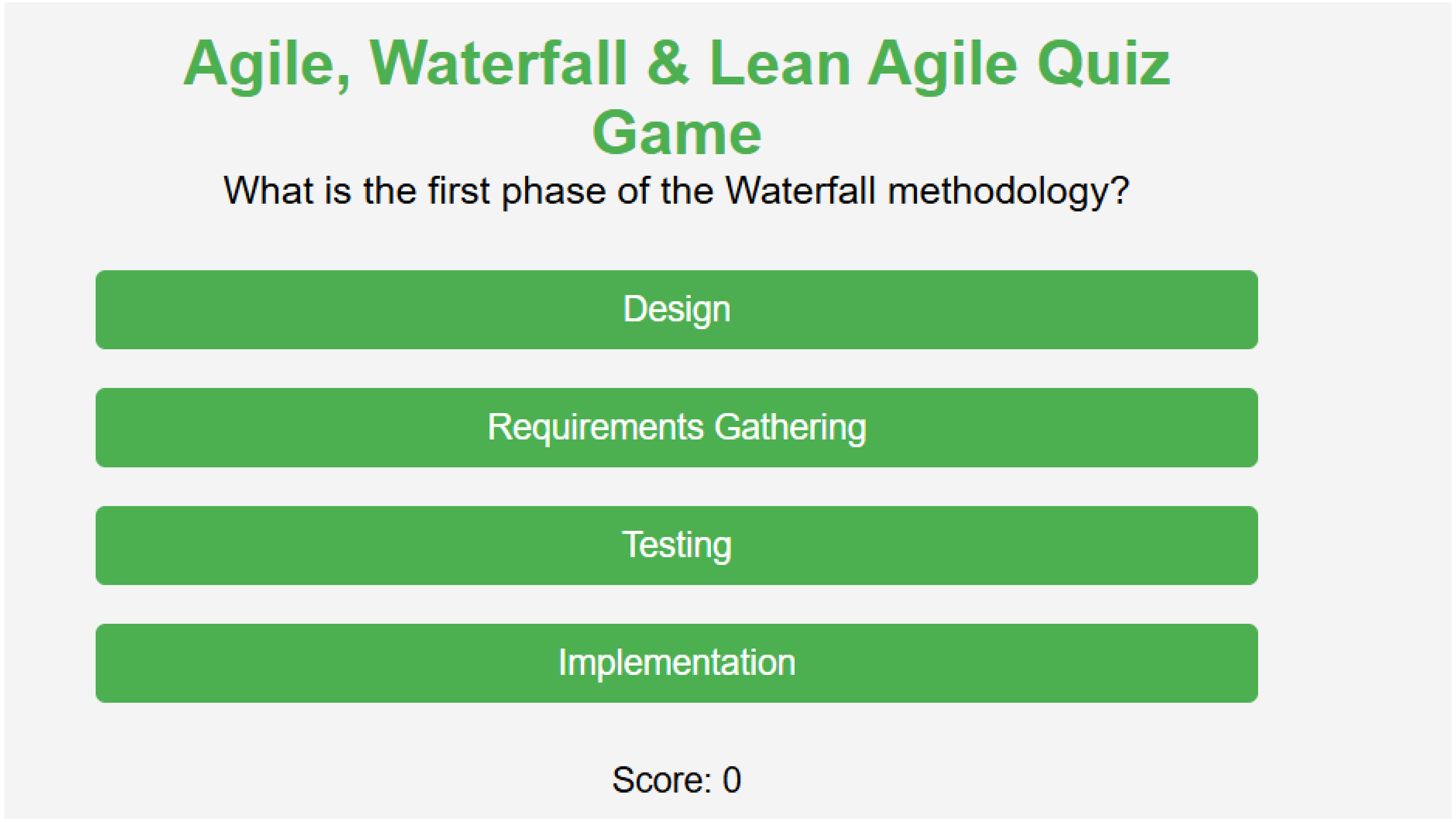
- Easier to replace frontend
- Better unit testing on core logic
- Codebase is flexible for change

Swapping the database without changing core logic fits clean architecture separation rule

Task-2

Agile, waterfall, lean agile Adventure

Let's learn the agile, waterfall, lean agile by playing the game



Agile, Waterfall & Lean Agile Quiz Game

What is the first phase of the Waterfall methodology?

Score: 0

Quiz Section

Quiz

Everyone student should click on submit button before time ends otherwise MCQs will not be submitted

[Guidelines of MCQs]

1. There are 20 MCQs
2. Time duration will be 10 minutes
3. This link will be share on 12:25pm (Pakistan time)
4. MCQs will start from 12:30pm (Pakistan time)
5. This is exact time and this will not change
6. Everyone student should click on submit button otherwise MCQs will not be submitted after time will finish
7. Every student should submit Github profile and LinkedIn post link for every class. It include in your performance

Assignment

Assignment should be submit before the next class

[Assignments Requirements]

1. Create a post of today's lecture and post on LinkedIn.
2. Make sure to tag @Plus W @Pak-Japan Centre and instructors LinkedIn profile
3. Upload your code of assignment and lecture on GitHub and share your GitHub profile in respective

   your region group WhatsApp group
4. If you have any query regarding assignment, please share on your region WhatsApp group.
5. Students who already done assignment, please support other students

Q&A Session

ありがとうございます。

Thank you.

شكريا

+W

For the World with Diverse Individualities