Introduction

DevOps:

Evolution:

**SDLC**

1. Planning
2. Analysis
3. Design
4. Implementation
5. Testing and Integration
6. Maintenance

**Waterfall model**

Requirements are clear and fixed.

Product definition is stable.

**Agile**

Requirements change frequently

Development needs to be fast

**DevOps approach**

Requirements change frequently

Development needs to be agile

Operations needs to be agile

**Facebook Issue**

New features released in 2011 made server meltdown across all data centers around the globe.

Dark Launching technique:

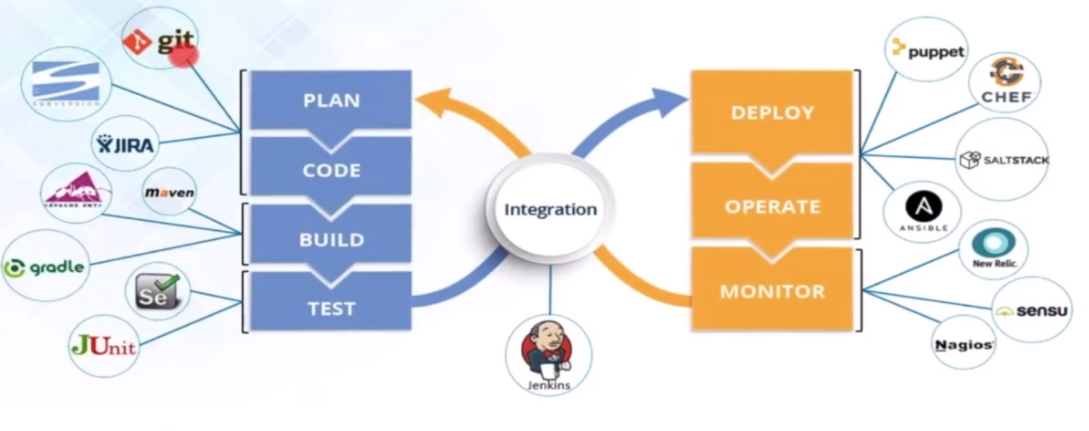
New features will be deployed on a smaller and specific user base.

Continuously monitored and feedback are developed and tested.

Once the features are stable they are deployed in multiple releases

**DevOps Life cycle**

1. Continuous Development
2. Continuous Testing
3. Continuous Integration
4. Continuous Deployment
5. Continuous Monitoring



Git – version control

Jira – Bug Tracking

Maven - Build

Gradle - Build

Selenium - Testing

Jenkins – Integration

Puppet – Operate

Chef - Automation

**Continuous Development**

Planning and coding

Code maintained by version control system

**Continuous Testing**

Build and Test

Lets developer know where the bugs are when built with previous code

**Continuous Integration**

Automated builds and automated tests

Helps detect errors quickly and locate quickly.

Suppose an error after a certain commit. U can check the error with the previous commit.

Jenkins

1. After every commit an auto build is triggered and automatically deployed on the test server.
2. If the results show there is a bug then the developers are informed

Continuous Deployment

Keeping code deployable at any point of time.

QA servers

Staging servers

Production servers.

**Continuous Deployment and Monitoring**

1. Configuration management
2. Containerization

Take care of application deployment on all the servers.

Monitoring - checking the performance