**Training Document**

**Topic:** Difference’s BetweenWaterfall Model and Agile Model

Difference’s Between Cloud Services

**Date:** 05/02/2025

**Name:** Leelavathi Chokkapu

**EID:** PCIS465

**Difference Between Waterfall Model and Agile Model**

**(1)Development Approach**

* **Waterfall Model:** Follows a linear and sequential approach, where each phase (Requirement → Design → Development → Testing → Deployment) is completed before moving to the next.
* **Agile Model:** Uses an iterative and flexible approach, where the project is divided into small cycles (sprints) that deliver working software at regular intervals.

**(2)Flexibility**

* **Waterfall Model:** **Not flexible**; once a phase is completed, it's difficult to go back and make changes.
* **Agile Model:** **Highly flexible**; allows continuous feedback and modifications throughout development.

**(3)Customer Involvement**

* **Waterfall Model:** Customers are involved only at the beginning (requirements phase) and at the end (final delivery).
* **Agile Model:** Customers are involved throughout the development process, providing feedback after every sprint.

**(4)Project Size and Complexity**

* **Waterfall Model:** Suitable for small, well-defined projects with clear requirements.
* **Agile Model:** Best for large, complex projects where requirements may change frequently.

**(5)Testing Approach**

* **Waterfall Model:** Testing happens only after development is complete.
* **Agile Model:** Testing is continuous throughout the development cycle.

**(6️)Delivery Time**

* **Waterfall Model:** The final product is delivered at the end of the development cycle (can take months or years).
* **Agile Model:** Frequent deliveries after every sprint (usually 2-4 weeks).

**(7️)Risk Management**

* **Waterfall Model:** High risk, as issues are detected late in the project.
* **Agile Model:** Low risk, as continuous feedback helps in early issue detection and resolution.

**(8️)Documentation**

* **Waterfall Model:** Requires detailed documentation before starting development.
* **Agile Model:** Focuses on working software over extensive documentation.

**(9)Team Collaboration**

* **Waterfall Model:** Teams work independently on each phase, with minimal interaction between developers, testers, and stakeholders.
* **Agile Model:** Encourages high collaboration, where developers, testers, and stakeholders work together continuously.

**(10)Cost Efficiency**

* **Waterfall Model:** Higher cost for changes since modifications require going back to earlier phases.
* **Agile Model:** More cost-effective, as issues are identified and resolved early in development.

(11)**Real-Time Examples**

**Waterfall Model (Construction a Bridge):** Think about the construction of a bridge or a building. Once the design is complete, we follow the construction steps step by step (foundation, structure, finishing, etc.). we can't go back and change the foundation once it's done, so the project is rigid and follows a strict order. Similarly, in the Waterfall Model, we complete each phase before moving to the next.

**Agile Model (Developing a Mobile App):**Think about creating a mobile app like Instagram or WhatsApp. we start by building a small, basic version of the app like the login page and release it after the first sprint. Then, we improve the app in future sprints by adding more features like chat, photo sharing, etc. we get feedback after each update, and based on that, we keep improving. This is the Agile Model approach.

**differences between SaaS** **(Software as a Service), PaaS** **(Platform as a Service), and IaaS((Infrastructure as a Service):**

**(1)Definition:-**

* **SaaS:** SaaS is a cloud computing model where you use software over the internet without worrying about installation, maintenance, or updates. It's like renting an app instead of owning it.
* **PaaS**: It provides a platform for developers to build, test, and deploy applications without managing the underlying infrastructure.
* **IaaS:** IaaS is a cloud computing model where you rent IT infrastructure like servers, storage, networking, and operating systems on a pay-as-you-go basis. Instead of buying and maintaining physical hardware, you get virtual resources from a cloud Vendor’s.

**(2️)Control:-**

* **SaaS:** only control **how** we use the software. Everything else is managed by the Vendor.
* **PaaS:** You control **the app you create** and how it's built but not the infrastructure behind it.
* **IaaS:** control **everything** except the physical hardware – from storage to networking to the virtual servers.

**(3️)Who Uses it:-**

* **SaaS:** **End-users** like individuals or teams who just need to use a software tool (e.g., Gmail, Zoom).
* **PaaS:** **Developers** or businesses who want to create apps without dealing with servers (e.g., Google App Engine, Heroku).
* **IaaS:** **IT teams** or large businesses who need full control over their infrastructure and systems (e.g., AWS, Microsoft Azure).

**(4️)Customization Level:-**

* **SaaS:** Limited customization – you use it as it is. Think of it as renting an apartment where you can't change the structure.
* **PaaS:** Moderate customization – you can build your app on the platform, but the environment is set.
* **IaaS:** High customization – you build exactly what you need, as if you own the land and can design anything.

**(5️)Setup Time:-**

* **SaaS:** You can use it **right away** with no setup required.
* **PaaS:** You can start building your app quickly, but some setup for the platform is needed.
* **IaaS:** Takes more time for setup because you need to configure and manage the infrastructure.

**(6️)Examples in Real Time:**

* **SaaS:** **Netflix** – Watch movies online without downloading.
  + **Google Drive** – Store and edit files from any device.
  + **Zoom** – Join meetings without setting up servers.
* **PaaS:** Google App Engine – Developers build apps without worrying about servers, databases, or hosting. They just write the code and deploy.
* **IaaS:** Amazon EC2 (AWS) – Companies rent virtual machines to host websites, store data, or run applications. They have full control over the OS and software.

**(7️)Maintenance Responsibility:-**

* **SaaS:** The provider handles **everything** (updates, security, etc.), and you only use the software.
* **PaaS:** The provider handles the **platform** (servers, OS, etc.), but you handle the apps you build.
* **IaaS:** You handle **everything** – servers, networks, and storage – while the provider only handles the hardware.

**(8️)Payment Structure:-**

* **SaaS:** You pay a **fixed fee** for access to the software, usually on a subscription basis.
* **PaaS:** You pay based on **resources used** (e.g., processing power, storage).
* **IaaS:** You pay based on the **infrastructure you use** – like renting a virtual server and paying for how much you use.

**(9️)deal Use Case:-**

* **SaaS:** For businesses or individuals who need a **ready-to-go solution**. Example: Using **Microsoft Office** for document creation.
* **PaaS:** Developers who want to **build and deploy apps** quickly without worrying about the infrastructure. Example: Building a web app on **Heroku**.
* **IaaS:** For businesses that need **customized, scalable infrastructure** for hosting apps, data, or services. Example: Running a complex application on **AWS EC2**.

**(10)Scalability:-**

* **SaaS:** **Limited scalability**. You can change plans or users, but the software is set.
* **PaaS:** **Moderate scalability**. You can scale your app as it grows but the platform remains the same.
* **IaaS:** **Highly scalable**. You can add more servers, storage, or networking as needed to meet demand.