Tasks Accomplished Today

Date: 05-02-2025
Name:Suneetha.V

Today Tasks:

1. What are the Major differences between SaaS, IaaS, PaaS.
2. What are the major differences between WaterFall Model and Agile Model.

1. What are the Major differences between SaaS, IaaS, PaaS.

Feature	SaaS (Software as a Service)	PaaS (Platform as a Service)	IaaS (Infrastructure as a Service)
Definition	Fully managed software applications delivered via the internet.	Platform for developing, running, and managing applications without managing infrastructure.	Virtualized computing resources over the internet.
User Control	Very little (just the application).	Control over applications and data.	Full control over OS, applications, and data.
Provider's Responsibility	Manages everything (software, hardware, updates).	Manages the platform (OS, runtime, servers, storage).	Manages hardware and networking (servers, storage).
User's Responsibility	No management, only use the application.	Manages application development and data.	Manages operating system, applications, and data.
Examples	1.Google Workspace, Salesforce, Dropbox. **2. Zomato** is like a fully functional restaurant (software) where customers (users) just walk in, order, and eat. Everything from food preparation to serving is managed by the restaurant. The customer does not need to worry about	1.Heroku, Google App Engine, Microsoft Azure App Service. 2.**Google App Engine** is like renting out a semi-equipped kitchen where the restaurant owner (developer) only needs to bring their own recipes (applications). The kitchen	1.AWS EC2, Google Cloud Compute, Microsoft Azure Compute. **2.AWS** is like the restaurant's kitchen infrastructure (virtual servers, storage, etc.). The restaurant owner can rent the kitchen (infrastructure), set up the menu (software), and manage how the food is prepared, but they don't have to own the building or kitchen appliances. They just pay for the resources they use.

Tasks Accomplished Today

Date: 05-02-2025
Name:Suneetha.V

	managing any kitchen or staff; they just use the service.	infrastructure (servers, storage) and cooking tools (platform services) are provided, so the chef (developer) can focus on preparing the food (building the app) without worrying about maintaining the kitchen or buying the equipment.	
Use Case	End-users who need a fully functional app without installation or maintenance.	Developers who need to focus on building apps without managing the underlying platform.	IT teams and developers who need to create custom infrastructure and manage it.
Infrastructure Management	No management needed.	Minimal (only applications).	Full management required (OS, networking, storage).
Target Audience	Businesses and consumers using software apps.	Developers building applications.	IT administrators, system architects, and developers.
Customization Level	Low (can't modify software).	Moderate (can customize app configurations).	High (complete control over environment and resources).
Cost Structure	Subscription-based (per user, per month).	Subscription-based (depending on resources used, e.g., compute).	Pay-as-you-go or subscription, based on resource consumption (e.g., compute, storage).

2. **What are the major differences between WaterFall Model and Agile Model**

Feature	Waterfall Model	Agile Model
Development Approach	Linear and sequential. Each phase must be completed before moving to the next.	Iterative and incremental. Development happens in cycles (sprints).
Flexibility	Rigid; changes are difficult and costly to	Highly flexible; changes can be incorporated at any stage.

Tasks Accomplished Today

Date: 05-02-2025
Name:Suneetha.V

	implement once a phase is complete.	
Project Phases	Distinct and non-overlapping phases: Requirements → Design → Implementation → Testing → Maintenance.	Continuous cycle of planning, development, testing, and reviewing in sprints.
Customer Involvement	Customer is involved mostly at the beginning (during requirements) and at the end (during delivery).	Continuous customer feedback and involvement throughout the development process.
Planning	Detailed upfront planning; scope, timelines, and resources are fixed early on.	Adaptive planning; scope and priorities evolve with each sprint.
Progress Tracking	Progress is measured by completed phases and deliverables.	Progress is tracked through completed features or user stories at the end of each sprint.
Testing	Testing happens after development is complete (late in the process).	Testing is done continuously throughout each sprint, often integrated with development.
Risk Management	Risks are identified upfront and addressed later in the project (if needed).	Risks are managed incrementally by adapting after each sprint.
Project Delivery	Final product is delivered after all phases are complete.	Working product increments are delivered at the end of each sprint.
Documentation	Heavy documentation is required at each phase.	Documentation is lighter and often focuses on key points relevant to the team.
Ideal for	Projects with well-defined, unchanging requirements.	Projects with evolving requirements and a need for flexibility.
Team Collaboration	Less emphasis on collaboration; the team works in a sequential manner.	High collaboration with frequent meetings (e.g., daily standups, sprint reviews).
Time to Market	Longer, with a single release at the end of the project.	Faster, with incremental releases at the end of each sprint.

Tasks Accomplished Today

Date: 05-02-2025
Name:Suneetha.V

3. What are the major differences between **V-Model**, **Prototype Model**, **Spiral Model**, **Incremental Model**, and **Iterative Model.**

Feature	V-Model	Prototype Model	Spiral Model	Incremental Model	Iterative Model
Approach	Sequential and linear with parallel testing	Rapid creation of prototypes with user feedback.	Iterative with emphasis on risk management .	Developmen t in small, incremental releases.	Repeating development cycles with refinements.
Phases	Clear, distinct phases (development and testing in parallel).	Starts with building a prototype, then refining based on feedback.	Multiple phases of planning, risk analysis, engineering, and testing.	Develop features in stages and integrate them	Develop in cycles, revisiting requirements and design after each cycle.
Testing	Testing happens after the development phase, but is still part of the process.	Testing is part of the prototype feedback loop.	Testing is done after each cycle (focused on risk).	Testing happens at the end of each incremental release.	Testing happens within each iteration (often after each sprint)
Risk Management	Limited to the planning and analysis phase.	Early user feedback helps reduce the risk of misunderstand ing requirements.	Focus on risk management at each iteration.	Risks are addressed in each increment based on feedback.	Risks are addressed after each cycle with revisions.
User Involvement	Involved mostly at the start (requirements) and end (acceptance testing).	Involved continuously with feedback on prototypes.	Involved regularly in planning and reviews.	involved after each increment to review features.	Regular involvement during feedback cycles.
Flexibility	Rigid; changes are costly once the project progresses.	High; the product evolves with user feedback.	High flexibility, especially in risk management	Moderate flexibility; scope can evolve between increments.	High flexibility with continuous feedback and adjustments.

Tasks Accomplished Today

Date: 05-02-2025
Name:Suneetha.V

Progress Measurement	Progress is measured by completed phases (design, implementation, testing).	Progress measured by iterative feedback and evolving prototypes.	Progress measured by risk reduction and completed cycles.	Measured by the completion of increments or features.	Progress measured by completion of iterations and feedback.
Time to Market	Longer, as testing happens after development.	Faster, as a prototype is built early to showcase concepts.	Can be quick due to early risk analysis and incremental cycles.	Faster as functional parts of the system are released early.	Faster due to continuous feedback and the ability to release partial versions.
Documentation	Extensive documentation, especially for design and testing.	Light on documentation, focused on prototypes.	Documentation is focused on planning, risk management, and each phase.	Documentation focuses on each increment's features and integration.	Documentation updates incrementally with each iteration.
Ideal for	Projects with well-defined requirements and a focus on quality assurance.	Projects with unclear requirements or where early user feedback is critical.	Complex, high-risk projects requiring frequent reevaluation.	Large projects with well-defined features that can be split into smaller parts.	Projects where the requirements evolve over time and flexibility is key.