Season 2

Episode : 1

Microservices vs Monolith - How to Build a project.

Software Development Life Cycle

→ In Industry, waterfall model approach is used.

1) what is waterfall model?

→ It is a methodology that is used in software & Product develop

→ It is a linear, sequential approach where each phase of the Project is completed before the next phase begins.

eg:

Requirements
↳ Design
↳ Development
↳ Testing
↳ Deployment
↳ Maintainance.

2) what are the things included in requirements?

→ what the project is about
→ what are the features we are going to build
→ How we are going to build
→ what are the different scenarios
→ who will be the audience
→ what will be the tech stack & so on.

→ Project Manager collects all these requirements

⇒ Product Manager + Designer → Mock Model.

2nd phase : Design Phase

→ Senior Engineer + Tech Lead defines the Design/architecture of the project (whether it is Monolith (or) Microservices)

→ Deciding the tech stack, how will they communicate

→ High Level Design (HLD) & Low level Design (LLD) are build in this phase.

3rd phase : Development

→ Developers are involved in developing this design & writing unit test cases

4th phase : Testing

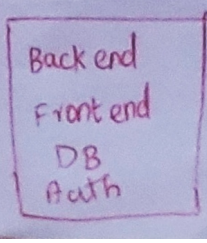
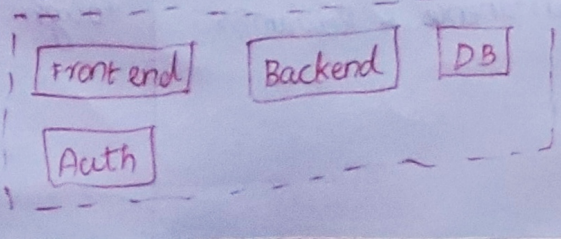
→ Testers test the project developed by the developers

5th phase : Deployment :

→ Devops manages the server & developers deploy the project after successfull testing phase.

→ It depends on company. if it is startup most of the works are done by developers only.

3) what is the difference between Monolithic & Microservices

Monolith	Microservices
→ The entire app is build as a single project. It has backend, Frontend, DB connection, Authentication, Emails, analytics, notification.	→ It consists of multiple small services, which are connected together where each services have their own responsibility such as one microservice for frontend, One for backend & so on.
eg: 	eg: 

Parameters	Monolith	Microservices
Development Speed	Slow	fast
code Repo	Single big repo	Multiple code repo. One repo for each Services
Scalability	As project grows. Scalability becomes tough	Each microservices are scaled independently
Deployment	Single deployment but whenever there is a change in code even its a single line we have to deploy the entire file once again	seperate deployment for each services.
Tech stack	should follow the same tech stack throughout the entire project	Each services can have their own tech stack. Suppose one service can built using React, Other can use Angular.
Infra cost	low	high
Complexity	tough	easy when project is larger.
Fault Isolation	Even a single bug breaks the entire app	it affects only that particular service.
Testing	Easier testing	Testing is done independently for each services.
ownership	handled by single team	Each services are handled by diff team
Maintanence & Rewamps	tough	easy