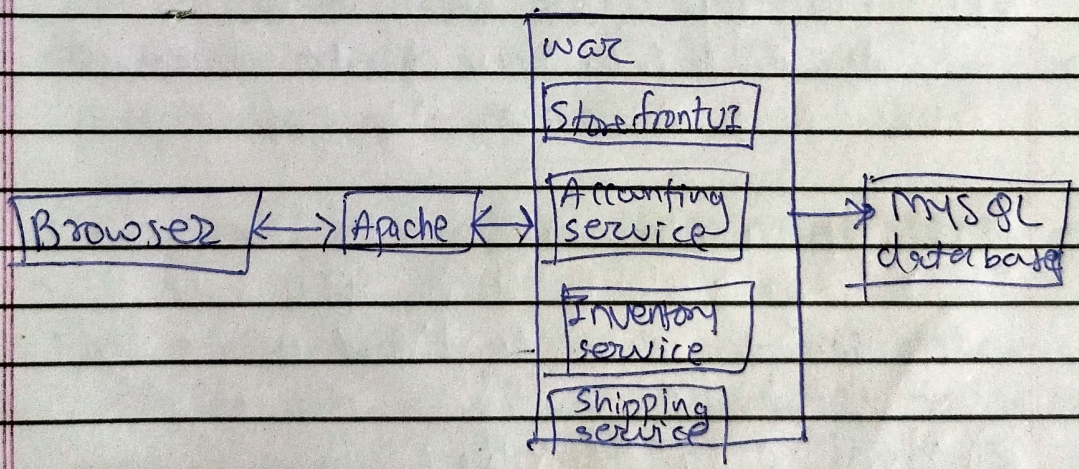
Unit-2 Microservices Architecture & Cloud Native Development

Page No.
Date

Q] Explain Monolithic application. Explain adv & disadv.

→

- Monolithic architecture is like a large container in which all the software components of an application are integrated into one large package.
- If all functionalities of project exists in single codebase, then the application is known as monolithic application.
- mono refers to single codebase containing all the required functionalities.
- If you build everything and deploy it on the server, that's called a monolithic architecture



Advantages

1) Easier to develop
2) Easier to test
3) Easier to deploy
4) Less complex & lower overhead.

Page No.
Date

Disadvantages

1) Becomes too large
2) Difficult to ~~Redeploy~~ Redeploy.
4) Hard for New Developers.
5) ~~Inefficient~~ Scaling.
6) Poor reliability.
7) Technology adoption is hard.

Page No.
Date

Q] Explain microservices architecture.

→ The objective of microservices is to completely decouple the app component such that they can be created, deployed scaled, maintained independently.

- Since all the code in monolithic is deployed together on the same base, adding or enhancing feature becomes a complicated process.

- Microservices allow large application to split into smaller pieces such that they can be developed or deployed individu

- In microservices the architecture talk to each other using lightweight applicati programming interface.

Microservices are implemented through

1) Communication → It contains many different seperate modules running individually which needs to communicate at right times hence communication is important.

Two ways → Synchronous → With all.
(Asynchronous) → Seperate communicat

2) Integration →

3) Deployment.

Advantages

1) Ider Independent Deployment →
You can update or deploy single service without redeploying all whole application.

2) Faster Deployment →
• multiple teams can work parallel on different services without interfering with each other.

3) Scalability →
• Scale only the service that need more resources

4) Fault isolation →
• If one service fails the rest of application can continue working.

5) Technology flexibility →
• Each microservices can use the most suitable programming language, database, or framework.

6) Better maintainability →
• Smaller modules are easy to maintain & understand.

7] Continous Delivery & faster Releases:

- Easier to adopt DevOps and CI/CD pipelines

Disadvantage

1) High complexity

2) More difficult Testing →

3) Data consistency issues → Each microservice have its own database making it harder to maintain.

4) Network overhead & latency → services communicates Of network which is slower.

5) Increase resource consumption →

6) Inter service communication needs → unlike monolithic app, devops team must ensure microservices can talk to another.

Q] Explain characteristics of micro service architecture.

→ 1) Decoupling → services within app are primarily decoupled. So the app can be easily build, alter, updat, deploy & scaled.

2) componentization → Microservices are treated as independent component that can be easily replaced & upgraded.

3) Business capabilities → Microservices are simple & focus on single microservices capabilities

4) Autonomy → Development team can work independently of each other, thus increasing speed & agility for the business

5) Continuous delivery → Allows frequent releases of software through systematic automation of software creation, testing, approval & deployment

6) Responsibility → microservice dont treat app like project & instead. They treat this as product with responsibility of their own.

7] Agility → Any new feature can be quickly developed.

Q) Compare monolithic & microservices.

features	Monolithic	Microservices
1) Definition	One container where all code is located.	App is divided into chunks.
2) Code.	All code in one big file/project	Each service has its own code.
3) Deployment	Must deploy whole app even for a small change.	Only particular change in targeted service.
4) failure	failure in one part can cause app to stop	If one service crashes others can still work.
5) Speed	Slow	Faster
6) Technology	One programming lang. for everything	Can use multiple programming as need
7) Database	One common DB	Each service have its own db.
8) Maintainance	Becomes difficult when the app grows big	Easier because each service is small & simple
9) Complexity	Easier to deploy & manage	Complex.

Q Explain microservices Best practices.

→ 1) One service = One Job.
- each microservice should do only one task.
- eg → in pizza app → separate services for orders, payments, delivery & inventory.

2) Own Database for each service
- Don't share on big database across all services.
- Each service should have its own data store & share data only via API's

3) Use asynchronous Communication.
- Services should send events and let others ~~react~~ react

4) ~~Fail fast~~ with circuit Breaker
- If service you depend is slow or down stop waiting & return a quick error.
- prevents one failure from ~~cas~~ crashing system.

5) API Gateway for all requests
- All client requests go through an API Gateway
- Handles authentication, logging & routing

6) Keep apis compatible

- Update API without breaking old client
- Update api by pretesting that it wont break existing users.

7) Dedicated Infrastructure.

- Host each service on separate, reliable infrastructure so one doesnt affect others

8) Independent release

- Each service should be deploy by its own without waiting for other services.

Q) Explain different deployment strategies

→

1) Recreate Deployment

- The old version of the application is completely shut down before deploying the new one.
- This results in downtime during the update
- This is usually performed during off-hour usually at night

2) Ramped (Rolling update)

- New version is released gradually by replacing instance one by one.
- Replace 1 server → wait → replace next

- Works well but may cause temporary mixed-version issues.

★ 3) Blue/Gree Deployment

1) Green → Old/live version
2) Blue → New version, tested & ready.

- After testing in Blue, you can switch all traffic from Green to Blue.
- If there's problem instantly switch back to green.
- Costly

★ 4) Canary deployment.

- The new version is released to a small percentage of users (10 %.)
- Rest of users will still use old version
- Performance & feedback are closly monitored
- If all set release the model gradually

Page No.
Date

Q] Explain Cloud Computing Service model.

→ Cloud computing is the delivery of computing services - including servers, storages, databases, networking, software over the internet (the cloud).
- You pay the amout only for services you use.

Cloud Computing Service models:

1) Infrastructure as a service (SAAS)
→ Think of it like renting a raw infrastructure (servers, storage, network) from the cloud engg
- The provider manages the hardware, network & virtualization.
- You manage the OS, app & data.
- You have full control over what you install & run.

2) Platform as a service
- Think of it like renting a ready platform where you just have to deal with Software.
- You will focus on coding & deployment
- Cloud engg will manage hardware, OS etc.

3) Software as a service (SaaS)
- This is ready made app through the internet
- The provider manages everything
- You just use software no need of installation.

eg → Google, Gmail etc...

Q] Explain Cloud Computing deployment model.

→

* 1] Public cloud
  - Services are provided over the public internet and accessible to anyone.
  - managed by third party cloud provider like Aws, Google Cloud, Azure.
  - Users pay on pay-per-use basis

pros: Low initial cost, no need for own hardware, quick setup
cons: Less control, security & privacy concern.

* 2] Private Cloud

  - Cloud infra is dedicated to only one person / organization
  - Can be managed internally or 3rd party vendor.
  - Offers more control & security

Pros → High security, full control over data

Cons → very expensive, skilled workers

* 3] Hybrid Cloud

  - Combination of public & private cloud
  - Sensitive data stored in private cloud & remaining in public cloud

pros → cost effective ; Keeps sensitive data secure.

cons → complex to manage

4] Community Cloud.

- shared by specific group of organization with common goals
- Cost for this is contributed by each organizat

pros → cost saving, easy collaboration.

cons → Security is risky, All members must agree on policies

Q] Explain the components of docker container.

→ 1) Image → The blueprint of our app which form the basic of containers

2) Containers → The instance of docker image & are what run the actual app

3) Docker Daemon :
- A background services that does all heavy lifting - builds images, run containers

4) Docker client
- It is the tool you use to send commands to dockers.
- Client communicate to docker daemon to perform the tasks

5) Docker hub
- Online registry to store Docker images
- You can pull public image from here or push your own.

6) Base image
- Starting point for creating image.
- Does not depend on other image.

7) Child images.
- Build on top of Base image with extra changes or software installed.

Page No.
Date

9] Diff between Automation vs orchestration.

Aspect	Automation	orchestration.
1) Definition	perform a single task or process automatically without human inputs	Coordinating & managing multiple automated tasks
2) Scope	Focuses on one specific task.	Covers multiple tasks & their sequence.
3) Complexity.	Less complex	More complex
4) Goal.	Increase speed, accuracy & efficiency of a single proces.	Ensure smooth execution.
5) Example.	Auto-deploying code to a server.	Deploying code, running tests, updating db. & notify in sequence
6) Tools	Jenkins	Kubernets.
7) Relations	Subset of orchestration	uses automation as building blocks