Deployment Strategies

1- Single Server Deployment

- All back-end services are hosted on one server.
- Simple and cost-efficient option for small apps.
- Host app on a single server provided by a cloud service.

Pros	Cons
Easy to set-up	Limited scalability
Low initial cost	Single point of failure

Use Cases			
Android	**iOS**	**Flutter**	**React-Native**
Game with minimal user interaction	App to manage personal budgets	Basic app connecting to single back-end	Light weight apps with basic functionalities

2- Load Balancing

- Distributes large incoming traffic on multiple servers.
- Prevents overwhelming a single server.
- Spreads requests along several servers.

Pros	Cons
Increased Relaibilty	More complex to setup
Better performance across heavy loads	High Operational cost

Use Cases	
Android and iOS	**Flutter and React-Native**
Instagram and snapchat can handle millions of requests.	Cross-Platform app can scale and balance numerous requests through APIs.

3- Microservices architecture

- Breaks down backend into smaller independent services.
- Allows you to develop, deploy each microservice independently.
- For example, In an E-commerce app you can develop separate service for user authentication, product listing and payment processing.

Pros	Cons
Greater flexibility and scalability	More complicated architecture
Easier to manage large teams	High Latency due to interservice communication

Use Cases	
Android and iOS	**Flutter and React-Native**
Large-scale applications like amazon and ebay	Use RESTful APIs to manage complex and distinct functionality apps

4- Serverless Architecture

- Allows building and running apps without managing servers.
- Use automated cloud scaling services.

Pros	Cons
Cost-efficient for variable workloads	Cold start latency
Automatic Scaling	Limited control over the environment

Use Cases	
Android and iOS	**Flutter and React-Native**
Apps fetching data on demand or requiring sporadic data processing	Easily integrate with serverless backend

5- Continuous Deployment

- Automated testing and deployment of the code changes.
- Allows releasing updates more frequently and reliably.
- Allowing pushing changes at any time.

Pros	Cons
Can launch app faster in market	Requires robust testing practices
Immediate feedback on changes	Can introduce bugs into production

Use Cases	
Android and iOS	**Flutter and React-Native**
News apps require frequent updates. Apps like Spotify that requires regular feature updates.	Flutter: Integration with CI/CD tools like Jenkins and GitHub actions React-Native: Agile development with immediate feedback