Day 6 – Group Project: Full-Stack App Deployment on AWS

I. Recap of Core AWS Services Used in Full-Stack App Deployment

Layer	AWS Services
Frontend	Amazon S3, CloudFront
Backend	EC2 / Lambda, API Gateway
Database	RDS (MySQL/PostgreSQL), DynamoDB
DevOps & CI/CD	CodePipeline, CodeBuild, CodeDeploy
Authentication	IAM, Cognito (optional)
Monitoring	CloudWatch
Storage	S3, EBS
Networking	VPC, Subnets, Internet Gateway, Security Groups

II. Team Formation & Project Plan

Team Setup:

- Teams of 3–5 members
- Assign roles:
  - DevOps Engineer
  - Backend Developer
  - Frontend Developer

  - Database Manager

Project Options:

1. **To-Do Application** – With user login, task creation, status update
2. **Feedback Form Web App** – Form data stored in DynamoDB/RDS
3. **Student Record Portal** – CRUD operations with API Gateway + Lambda

---

III. Step-by-Step Deployment Workflow

🔹 1. Frontend Setup (React/HTML App)

- Host frontend on S3 as a static website:
  - Upload `index.html`, JS/CSS files
  - Enable Static Website Hosting
  - Set Bucket Policy for public access
  - Optional: Use CloudFront for CDN

🔹 2. Backend API Setup

Option 1: Lambda + API Gateway

- Write Lambda function (Node.js or Python)
- Create REST API in API Gateway
- Connect Lambda with each HTTP method
- Deploy API & get endpoint

Option 2: Node.js/Express App on EC2

- Launch EC2 instance

- SSH & install Node.js
- Run server and expose via Security Group

🔹 3. Database Layer

Option A: DynamoDB

- Create table (e.g., `Tasks`, `Users`)
- Use Lambda to interact via AWS SDK

Option B: Amazon RDS

- Launch RDS instance (MySQL/PostgreSQL)
- Connect backend to DB using connection string

🔹 4. CI/CD (Optional)

- Set up CodePipeline with GitHub repo
- Add CodeBuild project for build/test
- Add deployment step using S3 or EC2

IV. Case Studies: Real-world Use Cases

1. Netflix on AWS

- Uses Amazon EC2 for compute
- Uses S3 for content storage
- Uses CloudFront for global delivery

2. Airbnb

- Uses EC2 for backend
- RDS and S3 for data management
- Uses IAM and VPC for secure networking

3. Zomato Clone (Educational)

- React + Node.js hosted via EC2/S3
- API Gateway + Lambda for orders
- DynamoDB for fast, serverless database

V. Common Troubleshooting Issues

Problem	Solution
Bucket gives 403 error	Add proper bucket policy, enable public access
EC2 not accessible	Open port 80/443 in Security Group
Lambda not triggered	Check API Gateway configuration & permissions
RDS connection fails	Add correct inbound rules, whitelist IP
API Gateway CORS error	Enable CORS in API Gateway
GitHub not deploying in pipeline	Check source connection and IAM permissions

---

VI. Q&A / Live Debugging

- Live debugging of issues faced during the group project
- Explanation of key pain points in AWS deployments
- Walkthrough of best practices

VII. Suggested Architecture Diagram

```
[ Client (Browser) ]
          |
      [ CloudFront ]
          |
       [ S3 - Frontend ]
          |
    [ API Gateway ] <--> [ Lambda Function / EC2 ]
          |
      [ DynamoDB / RDS ]
```