**📌 Overview**

This project demonstrates the deployment of a fully serverless web application using **Amazon Web Services (AWS)**. The application includes a static frontend hosted on **S3 + CloudFront**, dynamic backend APIs using **API Gateway + AWS Lambda**, and persistent data storage using **DynamoDB**.

**🧱 Architecture Components**

**🔸 1. Amazon S3 (Frontend Hosting)**

* Stores static files: HTML, CSS, JS, images.
* Public read access (or restricted via CloudFront).
* Acts as the **origin** for CloudFront.

**🔸 2. Amazon CloudFront (CDN)**

* Distributes frontend globally with low latency.
* Caches static content.
* Secures access to S3 via **Origin Access Control (OAC)** or OAI.

**🔸 3. Amazon API Gateway (REST API Layer)**

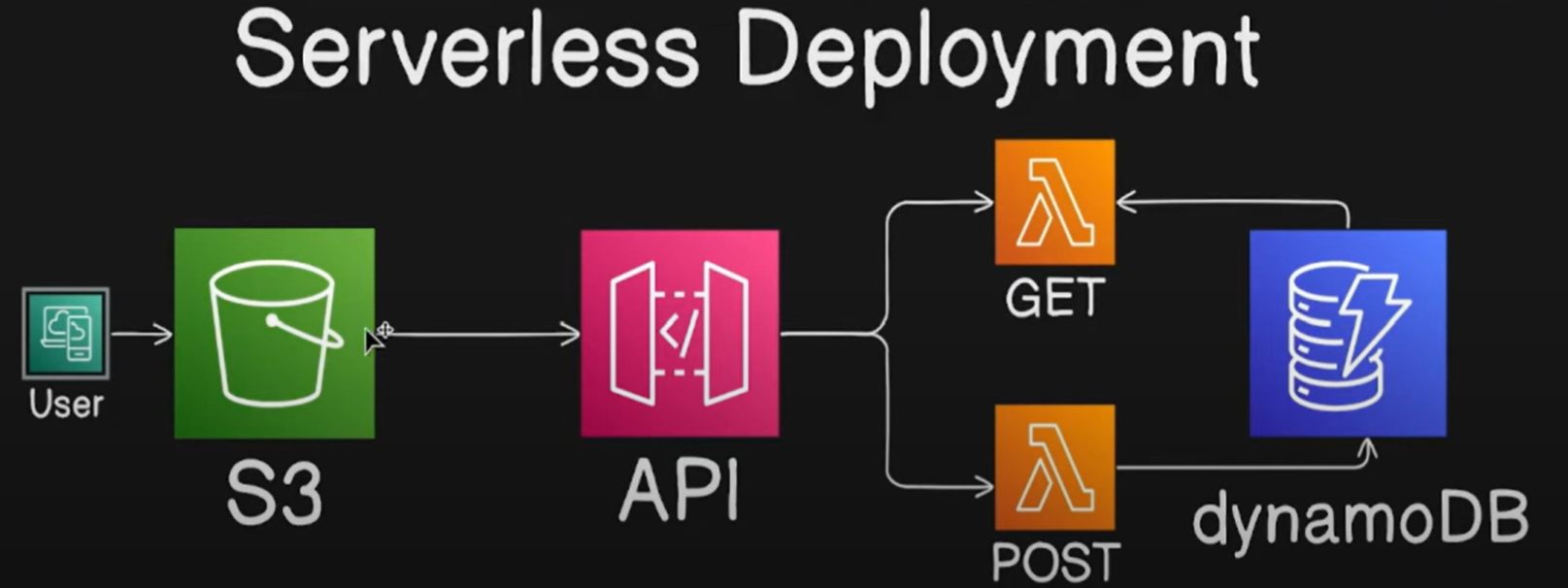
* Exposes RESTful endpoints to the frontend.
* Triggers Lambda functions on HTTP requests.
* Handles CORS and throttling.

**🔸 4. AWS Lambda (Business Logic)**

* Runs backend code (Node.js, Python, etc.) in response to API calls.
* Stateless and scalable on demand.
* Performs logic and interacts with DynamoDB.

**🔸 5. Amazon DynamoDB (Database)**

* NoSQL database storing dynamic data.
* Scalable, fast, and integrated with Lambda.
* Ideal for serverless apps due to on-demand pricing.



## 🛠️ Deployment Checklist

* **Create S3 bucket** and enable static hosting
* **Configure CloudFront**, set S3 as the origin
* **Build REST API** with API Gateway
* **Develop Lambda functions**, attach IAM roles
* **Design DynamoDB table(s)** with primary keys
* **Enable CORS** on API Gateway for frontend requests
* **Assign IAM permissions** for Lambda → DynamoDB
* **Deploy and test**, monitor via CloudWatch