Let's go step-by-step and understand the **Architecture Flow of deploying a Node.js App on AWS EKS** with minute details and concepts.

Step-by-Step Detailed Flow:

---

1. User Requests www.myapp.com

- **What happens?**
  - A user types www.myapp.com in a browser.
  - This domain is mapped in **Route53 (AWS DNS Service)**.
  - Route53 resolves the domain to an **Application Load Balancer (ALB)** public IP.
- **Concepts:**
  - **Route53 (optional):** Used for domain management.
  - **ALB:** Acts as a reverse proxy to route traffic into your Kubernetes cluster.

---

2. Application Load Balancer (ALB) forwards traffic to Kubernetes Service (LoadBalancer Type)

- **What happens?**
  - ALB listens on HTTP/HTTPS ports.
  - Based on configured **Ingress Rules** (like path /api, host www.myapp.com), ALB forwards traffic to a **Kubernetes Service**.
- **Concepts:**
  - **Ingress Controller (optional, like ALB Ingress Controller):** Creates ALB and manages routing based on Ingress resources.
  - **Kubernetes Service (Type: LoadBalancer):** Exposes Kubernetes Pods to the external world.
  - This Service is backed by AWS LoadBalancer IP.

---

3. Kubernetes Service Routes to Node.js Pods running on EKS Worker Nodes

- **What happens?**

    - The Kubernetes **Service** distributes incoming requests to healthy Pods.
    - It uses **selectors/labels** to identify the Pods to forward requests to.
    - Example: Service selects all Pods with label app: nodejs-app.
- **Concepts:**
    - **Service Discovery (Kube-DNS/CoreDNS):** Resolves service names inside the cluster.
    - **Kube-proxy:** Manages routing rules on worker nodes to forward traffic to Pods.
    - **Load Balancing across Pods:** Evenly distributes requests among Pods.

---

4. Pods are Managed by Deployment (Ensures Desired Replicas)

- **What happens?**
    - Pods are not created manually; they are controlled by a **Deployment** object.
    - Deployment ensures there are always, say, 3 running replicas.
    - If a Pod crashes, Deployment automatically spins up a new one.
- **Concepts:**
    - **Deployment:** Declarative way to manage Pods.
    - **ReplicaSets:** Under the hood, Deployment uses ReplicaSets to ensure Pod count.
    - **Self-Healing:** Kubernetes maintains "desired state".

---

5. Pods use ConfigMaps & Secrets for Configurations

- **What happens?**
    - The Node.js app might need configurations like DB Host, API keys, etc.
    - These are not hardcoded inside the container.
    - Instead, they are mounted as environment variables or files from **ConfigMaps (non-sensitive)** and **Secrets (sensitive data)**.
- **Concepts:**

- **ConfigMap:** Key-value pairs for app configurations.
- **Secret:** Encrypted storage for sensitive data.
- Pods consume these at runtime without needing to rebuild images.

---

6. If App Requires Data Persistence → EBS Volumes via PVCs

- **What happens?**
  - If your Node.js app needs to store persistent data (like file uploads), you use volumes.
  - Kubernetes **PersistentVolume (PV)** backed by AWS EBS is provisioned.
  - Pods request this storage via **PersistentVolumeClaim (PVC)**.
  - PVC gets bound to a PV dynamically.
- **Concepts:**
  - **PV/PVC abstraction:** Decouples storage provisioning from Pods.
  - **StorageClass:** Defines storage types (EBS, EFS, performance specs).
  - **Stateful Apps:** Necessary for apps requiring disk persistence across Pod restarts.

---

7. CloudWatch Captures Logs & Metrics for Monitoring

- **What happens?**
  - Logs from Node.js containers (stdout/stderr) are captured.
  - Metrics like CPU, Memory, Network are monitored.
  - You can use **CloudWatch Container Insights** for detailed observability.
- **Concepts:**
  - **CloudWatch Logs Agent/Fluentd:** For shipping logs.
  - **Metrics Server & Prometheus (optional):** For Kubernetes resource metrics.
  - **ALB Access Logs:** Traffic logs at LoadBalancer level.
  - CloudWatch helps with alerting, dashboards, and automated responses.

End-to-End Flow Recap:

Flow	What's Happening	Concepts
1. User Request	User hits www.myapp.com	Route53 DNS
2. ALB	Routes request based on Ingress rules	ALB, Ingress Controller
3. Kubernetes Service	Routes traffic to Pods	LoadBalancer Service, kube-proxy
4. Deployment & Pods	Ensures Pods are always running	Deployment, ReplicaSet, Pods
5. Configurations	Injects configs/secrets into Pods	ConfigMap, Secret
6. Persistent Storage	Attaches storage if needed	PVC, PV, EBS, StorageClass
7. Monitoring	Captures logs & metrics	CloudWatch Logs, Metrics