IAS Project (Group No: 5)

Deployer

Submitted By:

PAWAN PATIDAR (2020201031)
RONIT RAY (2020201024)
RAM PRATAP BACHINA (20173098)

Deployer

Introduction :

Deployment is generally defined as publishing an application or resource to a Web server, making it available for use.

Scope :

This is an application deployment platform which contains various independent services deployed using Microservices architecture such as handling dynamic scheduling requests, maintenance of job queue and database management services. The requests are managed by the scheduling manager via the application manager. The scheduler then reroutes the selected jobs to the deployer which assigns the necessary resources required to run them. It also employs a load balancer to distribute the requests to different nodes. A server health check module monitors the server heartbeat throughout the server lifecycle.

Intended Use:

The deployer will deploy applications/algorithms defined using the constraints specified by the application/communications manager. The scheduler will assign services/jobs to the deployer.

Assumptions and Dependencies:

At least one server node is assumed to be up, and its heartbeat status should be healthy.

Scheduler can schedule one job from its queue to the deployment manager.

Functional Overview:

The application works as:

- Scheduler module accepts the information from the application manager which validates the information of the user and now the scheduler will handle its scheduling information given by the user.
- The scheduler will assign the job to the deployer.
- Heartbeat monitor will determine current server status and if status is fine will proceed.
- Load balancer component will assign to the appropriate server node.
- Deployer will execute the specific algorithm for the application.

List of sub-systems:

Following subsystems are involved:

a) Heartbeat monitor manager
b) Load balancer
c) Node manager/server lifecycle monitor

List of Services/Technologies used:

1.) Kafka
2.) MongoDB
3.) Flask
4.) Python Framework

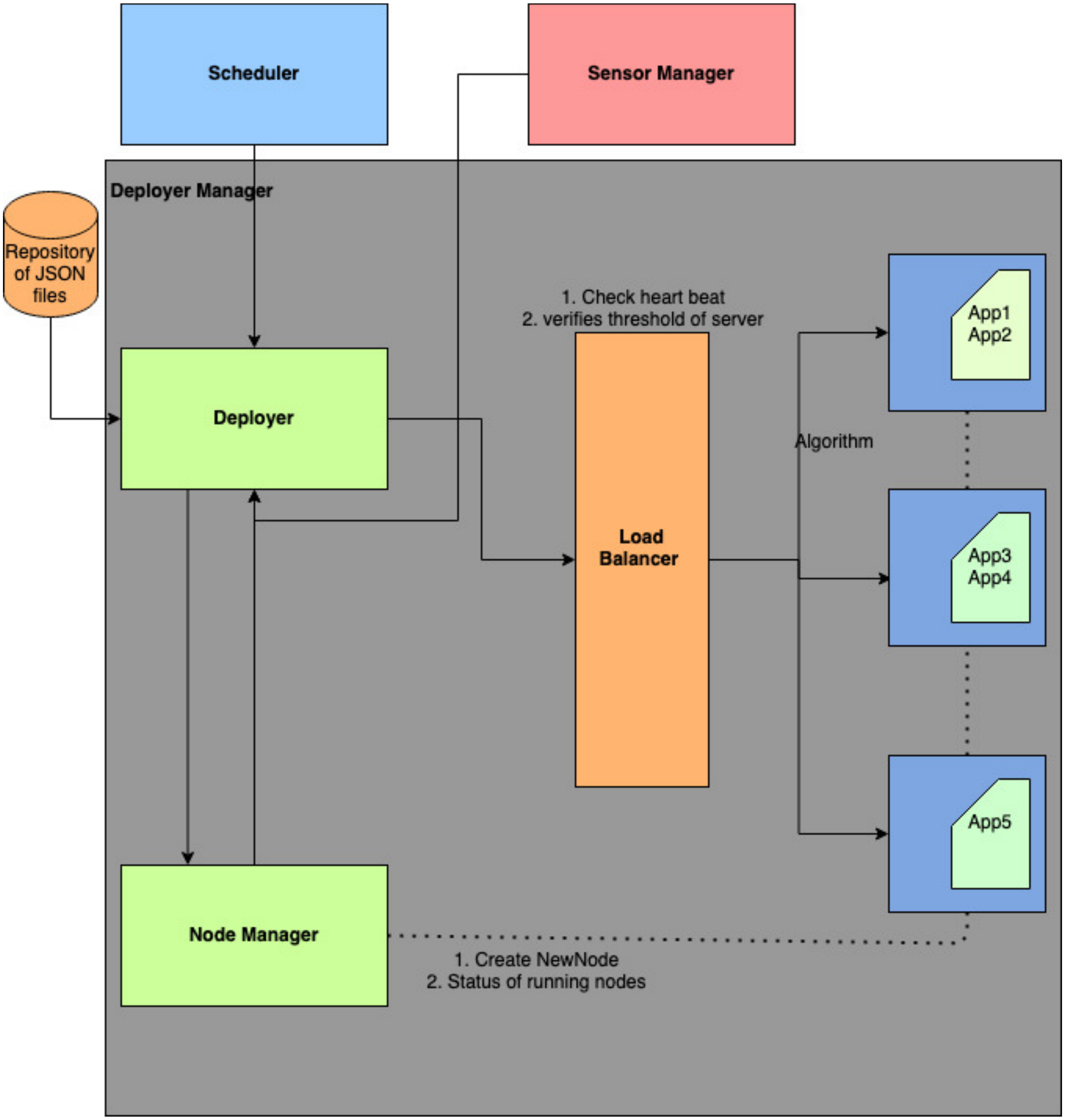
Interactions with other Modules:

1) Deployer communicates with load balancer using Kafka for assigning appropriate server node.
2) During execution server nodes interacts with communication manager as per the algorithm.
3) Deployer then communicates with notification and action manager and forwards appropriate actions.

Test Cases

1) Check if the server goes offline.
2) Integration check with load manager
3) Server connection check to get values from each node for a heartbeat monitoring service.
4) Test case to validate load on server freeness.
5) Integration check application and sensor id binding
6) Any node server goes down or needs to be unallocated.

Block Diagram


Scheduler
Sensor Manager
Deployer Manager
Repository of JSON files
1. Check heart beat
2. verifies threshold of server
App1
App2
Deployer
Algorithm
Load Balancer
App3
App4
App5
Node Manager
1. Create NewNode
2. Status of running nodes