Team Requirement Document
Group 5- Team 3

Node Manager, Load Balancer and Api-gateway

Team members
Shukan Patel(2022201052)
Yash Chandaliya(2022202010)
Rutvik Jakhaniya(2022202015)

1. Node Manager

Description

This module contains information about nodes running. It stores the information about all nodes in nodes DB and when load balancer asks for a node details to deploy a service it returns the details of active nodes. It also creates a new node on the request of the load balancer.

Components

- Node Manager - Node manager manages the data about active nodes and provides the details load balancer whenever needed.
- Node DB - Stores the data provided by node manager and provides the data to node manager when asked for it.

Functional Requirements

- Providing the details of active nodes to load balancer.
- Create a node(Make node active).
- Maintaining the node registry.
- Make nodes free when no service is running on the node.

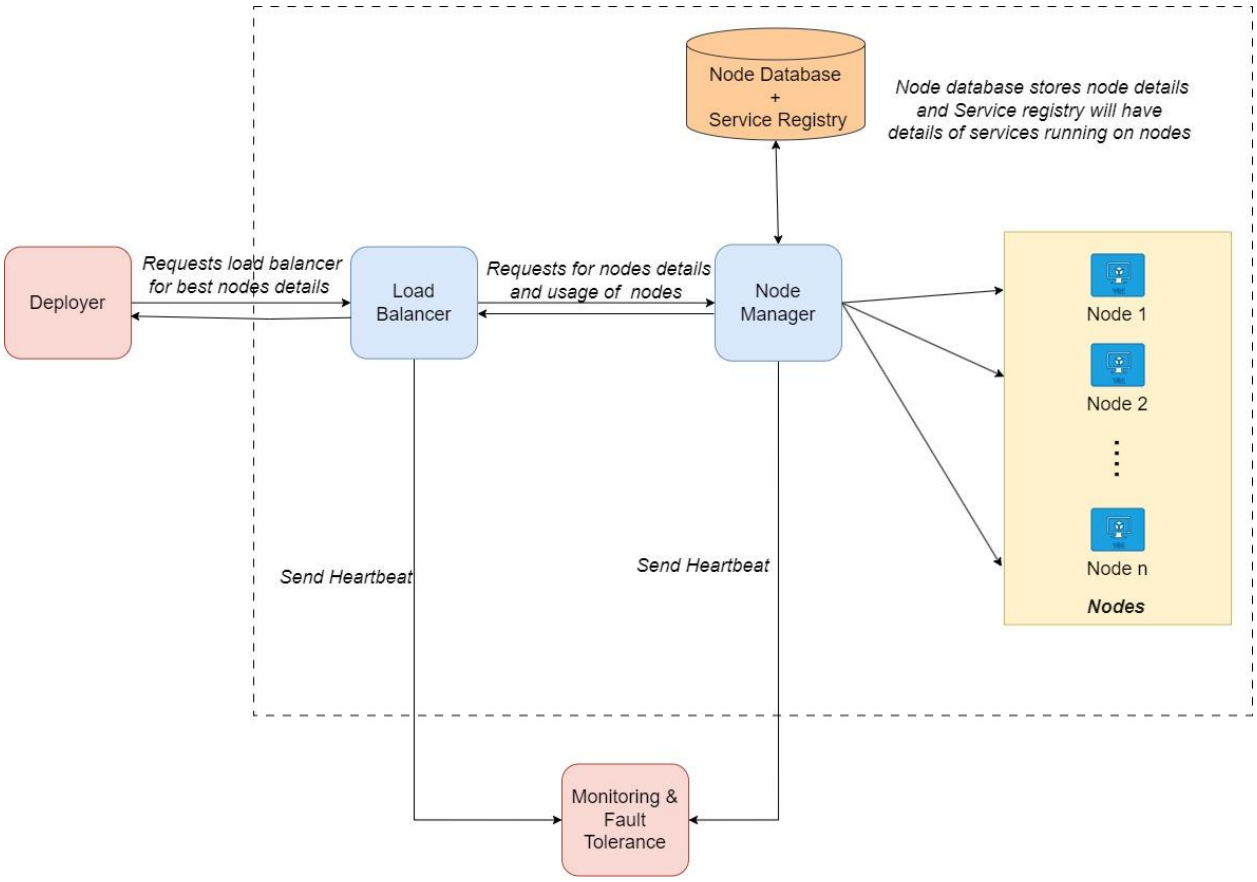
Interaction between components

- **Node DB - Node Manager :** Node Manager stores the data about nodes in Node DB and retrieves the data from Node DB whenever needed.
- **Node - Node Manager :** Nodes send their status to the node manager.

Interaction with other modules

- Load Balancer - It provides information about active nodes to the load balancer and creates a new node if the load balancer asks for it.

- Monitoring and Fault tolerance - It sends heartbeats to monitoring and fault tolerance to tell that it is alive and working.



Node Manager and Load Balancer Diagram

Service Registry Node DB

Node DB will have details of nodes as follows:

- Node name
- Ip
- User name
- Password
- status(active/free)

Service Registry

Registry will be used to store several run-time related information for applications and platform modules as follows:

- App name
- Service name
- Node name

- Node ip
- Port
- Container Id
- Container Up time
- Container Name

Test Cases

- When Load Balancer requests the details of the active nodes, Node Manager sends the details of active nodes to Load Balancer.
  Input - Deployer requests a node to deploy the service and all nodes provided by node manager are at its full capacity.
  Output - Request the node manager to create a new node and provide its details.
- On request of new node creation from Load Balancer, Node Manager creates a new node.
  Input - Load Balancer requests to create a new node.
  Output - Node manager creates the new node and sends its details to load balancer.

2. Load Balancer

Description

This module is responsible for providing the best node to the deployer to deploy a service and also checks for container utilization. It asks for node details from the node manager and selects the best node(one with the lowest node).

Functional Requirements

- Select the best node amongst the node details provided by the node manager.
- If all nodes provided are at its full capacity then ask the node manager to create a new node and provide its details.
- Provide the selected node to the deployer to deploy a service.
- It will continuously check the health of containers, if it finds an overloaded container it should inform the deployer to create a new instance of that service.

Interaction with other modules

- Node manager - Request the node manager to get the details of active nodes.
- Deployer - Accept the request from the deployer to deploy a service and provide a node to the deployer to deploy the service.
- Deployer - Load balancer sends the details of service when it finds an overload on that container(service).
- Monitoring and Fault tolerance - It sends the heartbeat signal to monitoring and fault tolerance.

Test cases

- On requesting a node to deploy a service load balancer should return the node to deploy the service.
  Input - Deployer requests a node to deploy the service and all nodes provided by node manager are at its full capacity.
  Output - Request the node manager to create a new node and provide its details.
- Load balancer finds that service s1 running on node n1 and container c1 is overloaded. So it sends details of the service to the deployer to create one new instance of that service running.
  Input: LB finds c1 overloaded.
  Output: LB sends details of s1 to deployer.

3. API Gateway

Description

API gateway is responsible for routes the requests for the whole platform. One needs to use the service and don't know the domain and port of that service, that will make a request to api gateway and api gateway will route that request to particular service and send back the response.

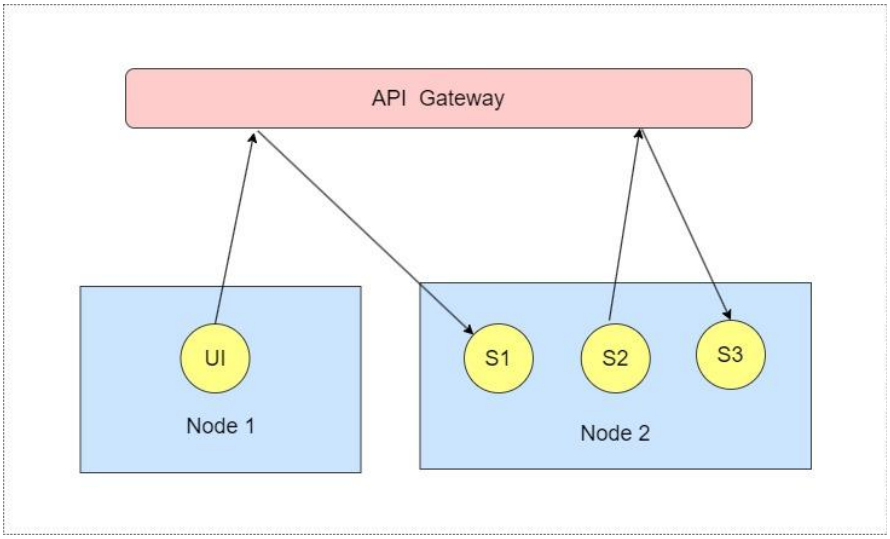
Functional Requirements

- Service can request a gateway for another service.
- UI can request a gateway with a file in the request data.
- UI can request for different data to show on display.

Interaction with other modules

- Service - It provides information about active nodes to the load balancer and creates a new node if the load balancer asks for it.
- Developer UI - It sends heartbeats to monitoring and fault tolerance to tell that it is alive and working.

Block Diagram



Test Cases

- When a service running on a platform requires a request to another service it will request an api gateway. API gateway will look at the service registry and forward request if request service is running.

Input - Service makes POST request to api gateway with required details like request type, name of service and app, etc.
Output - forwarded request will send back a response, that response is returned by api gateway.