App42 Paas: Architecture Discussion

1. **AMI Creation**: Different Container Templates(Based on different languages) are to be made.

**Discussion needed on**: New Language support.

**Assumed approach**: 1. Single Language Template.

2. Multi Language Template.

1. Container Name should be the **deployment id**.

Different fields discussed:

1. App environment
2. Infra
3. App version
4. Source code url
5. App type
6. Container specs
7. Service - Service Info
8. App current status
9. App actual status
10. Date created
11. Date deleted/updated
12. App url
13. **Deployment Table Fields**:
    1. Deployment Id
    2. Host
    3. Source code cdn url
    4. App version
    5. App Id
    6. Date
    7. App Status
14. **PaaS Flow**:
15. New (First time app deploy) --------------- discussed
16. App Update --------------- discussed
17. Start/Stop/Delete --------------- discussed
18. Scale --------------- discussed
19. De-Scale --------------- discussed
20. Rebuild --------------- Still to be discussed
21. App Health --------------- In-progress
22. VM Capacity Planning --------------- Still to be discussed
23. Service Create --------------- Still to be discussed
24. Service bind/unbind --------------- Still to be discussed

New (First time app deploy)

1. **Agents**: Two agents are there
   1. Agent1 (Provisioner): Will be inside the VM.

For container creation based on deployment id, type and source url .

* 1. Agent2: Will be inside the container.

For App Deployment and listening message on the queue based on the deployment id.

1. **Issue**:

When an app is pushed, how to decide which VM to deploy on ?

Approach:

1. **Agent will listen using Message/queue**: Message will be broadcasted to all the VM’s and the VM with the available configuration same as the request will be given the request based on First come first serve priority.

**Spike**:

**Case**: If a user requests the deploy command with 1 GB RAM, the agent will broadcast the request and the VM which responds first will be given that request. Suppose the VM has only 2GB RAM available(but the deploy operation is not completed yet, so the memory left is 1GB ) , but the VM hasn’t lock that memory (1GB), so if another request comes on the same VM with 2 GB RAM, then that request will also be available. Hence problem arises.

So we have to find a way to lock that memory and if the request is killed by the user, then that memory should be un-locked too.

1. **Agent will use the database**: Agent decides the VM based on the available data in the database.

**Issue**: Too much database operation.

APP UPDATE

App Update will be done on the basis of deployment table. When the user request the update command, the agent will update the code on cdn with its version and updates the deployment table for the url. The agent present inside the container will listen on the queue for any update in the deployment table, if any update is there it will check it with the current version and will therefore update the code on the container.

START/STOP/DELETE

Stop: When an app is stopped ,as per discussion, we will delete the app as well as the container and on start command, the agent will get the latest version on the code and will redeploy the app.

SCALE / De-Scale

Scale will be done on the basis of availability zone. If an app is running on aws-east-1a , then it will be first scale to aws-east-1b

Descale will be done randomly.

APP HEALTH

**Problem**:

1. **App** **Level** :
   1. App Problem:
      1. Memory leak
      2. System Exit
2. **Container** **Level**:
   1. Manual
3. **VM** **Level**:
   1. Manual
   2. Infrastructure is down (availability zone)

As per discussion, there will be two agents and restart scripts:

1. Inside the VM, which will check the container’s status and the cause of problem and on that basis will run the restart script.
2. Inside the Container, which will check the app’s status and the cause and on that basis will run the restart script.

**Monitor – Action approach**:

When a component is down, monitor will monitor the cause of the failure and report it, while the action will perform the self healing task(i.e will start the container/app only iff it is not deleted).

Based on container config we have to configure web-server too(perm size etc)

Services -> Backup and restore data