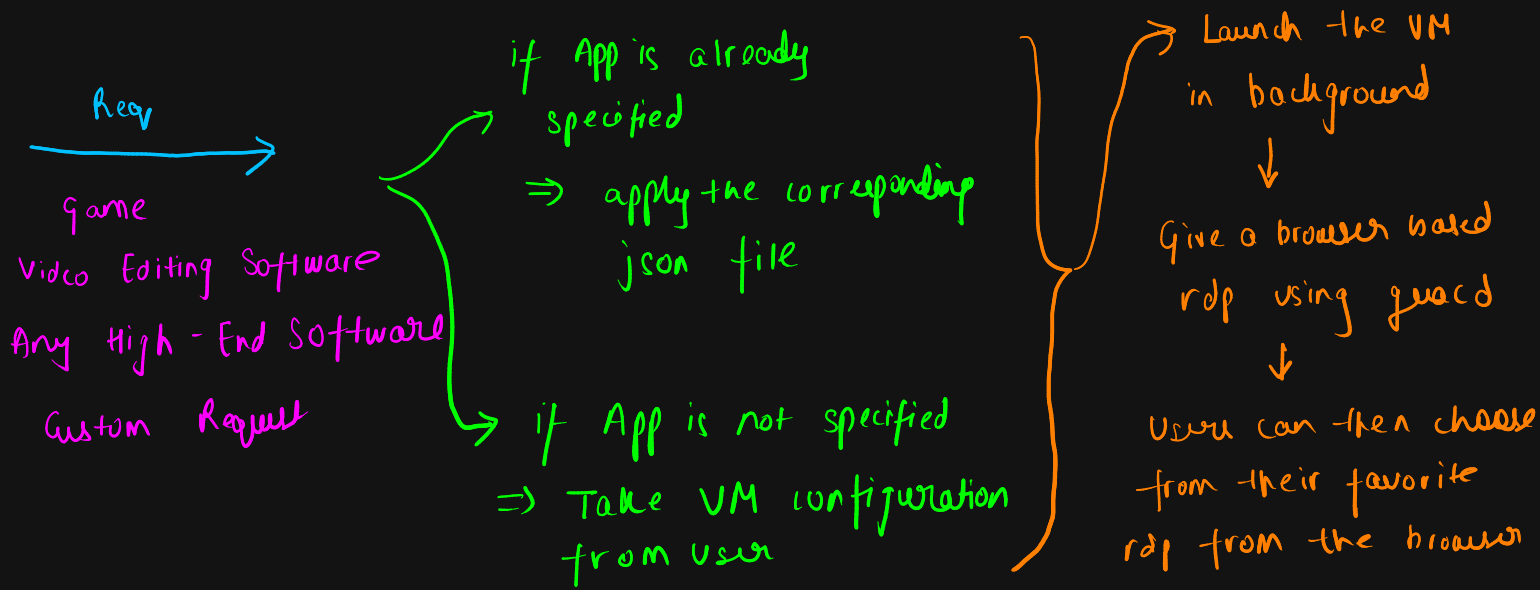


Cloud Remote Desktop Services

* Basic Architecture



* Json structure for the Applications.

• Games

- ↳ High-End-Games
- ↳ Mid-, low
- ↳ Base (for those case when no application is specified)

• Editing

- ↳ High-End-Software
- ↳ Mid
- ↳ Low
- ↳ Base

etc.

```
{
  Apps: [
    {
      ami_id: "_____"
      instance_type: "g4.xdn.large"
      storage_space: "_____"
    }
  ],
  azure: {
  },
  gcp: {
  },
  ...
}
```


* Timer

→ Frontend: Once the VM launches a simple information about start time should be present at frontend. There we can start a stop-watch like timer for UX.

→ Backend: Push all the important information about the VM in a time based queue & terminate the VM once the timer finishes.

Possible Solutions

↳ Job Queue, Message Queue, Task Queue

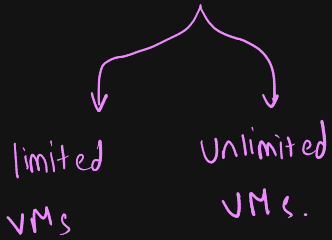
⇒ User should have ability to increase the timer on the go by paying.
If user pays, the job queue & timer should get updated.

User and Membership Management

* Plans

↳ Hourly Plans: No user data required, should be managed just with sessions & cookies. Doesn't matter even if the user has an account.

↳ Monthly Plans: User account is mandatory. User should be able to see their VMs, running or stopped and all the data should be present there.



if idle > 30 mins \Rightarrow Stop the VM automatically to save costs.

user should see their stopped VMs & should be able to start it.

* Automation

1. Instead of launching a default RDP in browser, the game should be launched automatically.
2. Instead of allowing user to choose their favorite RDP ~~from~~ from the browser RDP, allow them to choose beforehand & directly start the required RDP

How to achieve this?

↳ SSH login to the remote VM

↳ Run a script to execute RDP

↳ Fill all the details automatically using python's auto GUI method.

* Database

User

- ↳ ID
- ↳ Name
- ↳ Email
- ↳ Phone-Number
- ↳ Subscription-id
- ↳ User-type
 - ↳ Admin
 - ↳ Editor
 - ↳ client or customer

Applications

- ↳ ID
- ↳ Name
- ↳ Description
- ↳ Service ID
- ↳ Application_Pack_ID
- ↳ Slug

Services/Category

- ↳ ID
- ↳ Name
- ↳ Description

RDP

- ↳ Name ex- Panelc
- ↳ Description
- ↳ ID

Can contain

- ↳ Base types also
- base-games.json
- base-editing.json

VM-Detail

- ↳ ID
- ↳ VM ID
- ↳ Created-At
- ↳ Public-IP
- ↳ Username
- ↳ Password
- or
- ↳ Encoded-Token
- ↳ VM-Info (In json format)
- ↳ State (Running, Stopped, Deleted)
- ↳ duration (no. of hours)
- ↳ Start-time
- ↳ End-time
- ↳ Time-Elapsed

Cloud Providers

- ↳ ID
- ↳ Name (AWS, GCP, Azure, Vult)
- ↳ identifier: AWS, Azure, GCP

Memberships or Plans/Subscription

- ↳ ID
- ↳ Name
- ↳ Description
- ↳ cost
- ↳ Duration (In Months)
- ↳ Allowed_Application_Pack

we should not allow all plans ex- A person purchasing a plan for editing purposes should not pay a high price for a gaming rig

Others required Information

- line, ↳ NO. of-hours/per month
- ↳ no. of VMs can launch at once

Applications - Pack

- ↳ ID
- ↳ config-json
- ↳ Name ("High-End-games or some fancy name")
- ↳ Description

* User-Subscription

- ↳ User-ID
- ↳ Subscription-ID
- ↳ Start-time
- ↳ Expiry-time
- ↳ transaction-ID

* Transaction

- ↳ ID
- ↳ transaction-id
- ↳ user-ID
- ↳ transaction-amount
- ↳ created-at
- ↳ updated-at
- ↳ Successful
- ↳ token
- ↳ Description: which subscription package was chosen, for duration etc.

Application Pack Costing

- ↳ ID
 - ↳ Application-pack-id
 - ↳ cloud-provider-id
 - ↳ cost
- ↳ hourly-cost
monthly cost will already be included in monthly subscription
- ↳ B coz different cloud providers will have different prices for the same VM.

* VM-Info:

↳ Start-time

↳ Duration

Start-time & stopped time will be changed frequently to calculate the time elapsed.

for original start time, created-At is enough

↳ Stopped-time → This is not the actual end time but the time at which the instance was stopped & elapsed-time

Start time = 1 pm

Duration = 5 hrs

Stop time = 2 pm

} 1 hr

still 4 hrs are left.

Thus start-time & stop-time can be updated frequently.

* Elapsed-time = Stop-time - Start-time

Every time the same VM is launched, it should also be added in the queue to delete with the scheduled time

= Duration - Elapsed-time