 **GRT INSTITUTE OF ENGINEERING AND**

**TECHNOLOGY, TIRUTTANI - 631209**

**Approved by AICTE, New Delhi Affiliated to Anna University, Chennai**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**DISASTER RECOVERY WITH IBM CLOUD VIRTUAL SERVERS**

**PROJECT REPORT**

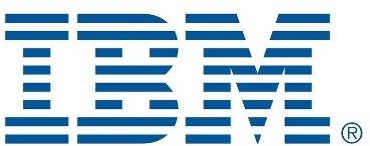
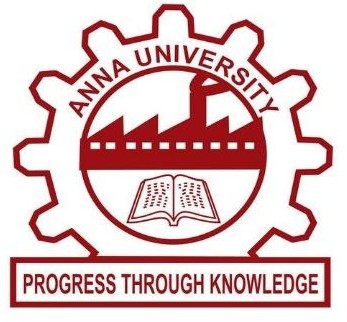
**SUBMITTED BY**

PRIYADHARSHINI.S

3RD YEAR, 5TH SEM

110321104037

priya2004madhi@gmail.com



**ANNA UNIVERSITY:CHENNAI 600 025**

**BONAFIDE CERTIFICATE**

Certified that this project report **“DISASTER RECOVERY WITH IBM CLOUD VIRTUAL SERVERS”** is the bonafide work of **"PRIYADHARSHINI.S [110321104037]”** who carried out the project work under my our supervision.

**SIGNATURE SIGNATURE**

**Dr.N. Kamal M.E.,Ph.D**. **Mr.T.A. Vinayagam M.Tech.,**

**HOD Assistant professor**

Department of Computer Science And Department of Computer Science And

Engineering Engineering

GRT Institute of Engineering and GRT Institute of Engineering and

Technology Technology

Tiruttani Tiruttani

Certified that the candidates were examined in Viva-voce in the Examination Held on

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**ACKNOWLEDGEMENT**

We thank our Management for providing us all support to complete this project successfully.

Our sincere thanks to honorable **Chairman**, **Shri. G. RAJENDRAN and Managing Director, Shri.G.R. RADHAKRISHNAN** for creating a wonderful atmosphere inside the campus.

We are very grateful to **Dr.S. ARUMUGAM, M.E., Ph.D., Principal**, for providing us with consistent guidance and motivation to execute a real time project to learn and experience the project work in an environments to complete our project successfully.

Our sincere thanks to **Dr.N. KAMAL, M.E., Ph.D., Professor and Head, Department of Computer Science and Engineering** for giving me this wonderful opportunity to do the project and providing the require facilities to fulfill our work.

We are highly indebted and thankful to our project Evaluators **Mrs V. Priya M.E**.,

and **Mrs. Edith Esther M.E., Assistant Professor, Department of Computer Science and**

**Engineering** for his immense support in doing the project.

We are very grateful to our internal guide **Mr. T.A. VINAYAGAM, M.Tech., Assistant Professor, Department of Computer Science and Engineering** for guiding us with her valuable suggestions to complete our project.

We also dedicate equal and grateful acknowledgement to all the **respectable members of the faculty and lab in-charges** of the Department of Computer Science and Engineering, friends and our families for their motivation, encouragement and continuous support.

Our sincere thanks **to IBM and Skill Up Team members** for giving me this wonderful opportunity to do the project and providing the require guidance and valuable online sessions.

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO** | **TABLE OF CONTENTS** | **PAGE NO** |
| **1.**  **2.**  **3.** | **PHASE 1**  1.0 ABSTRACT  1.2 PROBLEM DEFINITION  1.3 DESIGN THINKING  **PHASE 2**  2.0 ALGORITHM  2.1 PRODUCTION SITE  2.2 DISASTER RECOVERY SITE  2.4 PROGRAMMING LANGUAGE  **PHASE 3**  3.0 BUILDING THE DISASTER RECOVERY PLAN  3.0.1 SOURCE CODE  3.1 DISASTER RECOVERY STRATERGY  3.1.1 RECOVERY TIME OBJECTIVE  3.1.2 RECOVERY POINT OBJECTIVE  3.1.3 THE PRIORITIZATION OF VIRTUAL MACHINE DETAILS  3.2 SETTING UP OD BACKUPS  3.2.1 SOURCE CODE  3.3 SOURCE LANGUAGE  3.4 |  |

|  |  |  |
| --- | --- | --- |
| **4.**  **5.**  **6.** | **PHASE 4**  4.0 TESTING RECOVERY PLAN  4.0.1 SOURCE CODE  4.2 OUTPUT  CONCLUSION  REFERENCE |  |

**DISASTER RECOVERY WITH IBM CLOUD VIRTUAL SERVERS**

**1.0 ABSTRACT**

Disaster recovery with IBM Cloud Virtual Server is a feature that allows organizations to ensure business continuity in the event of a disaster or system failure. With this feature, organizations can replicate their virtual servers and data to a secondary location in the IBM Cloud, providing a backup environment that can be quickly activated in case of a disaster.

This abstract provides an overview of how IBM Cloud Virtual Servers offer a reliable and flexible solution for disaster recovery. IBM Cloud Virtual Servers provide a scalable and resilient cloud computing platform that can be leveraged to create a comprehensive DR plan tailored to an organization's specific needs.

In today's interconnected and data-driven world, organizations rely heavily on their digital infrastructure to operate efficiently and maintain business continuity. This feature provides a reliable and scalable solution for disaster recovery, allowing organizations to recover quickly and efficiently from any disruptions to their systems.

IBM Cloud Virtual Server offers various disaster recovery options, including continuous replication, backup and restore, and failover capabilities. Continuous replication ensures that data changes are continuously replicated from the primary server to the secondary

server, minimizing data loss in case of a disaster. Backup and restore capabilities allow organizations to create regular backups of their virtual servers and data, which can be restored in case of a failure. Failover capabilities enable organizations to quickly

switch to the secondary server in case the primary server becomes unavailable.

By leveraging IBM Cloud Virtual Server for disaster recovery, organizations can minimize downtime, protect their data, and ensure business continuity even in the face of unexpected events. This feature provides a reliable and scalable solution for disaster recovery, allowing organizations to recover quickly and efficiently from any disruptions to their systems.

In today's digital age, where businesses heavily rely on their IT infrastructure, the need for robust disaster recovery solutions has never been greater. Disruptions, whether caused by natural disasters, cyberattacks, or hardware failures, can lead to data loss and downtime, with potentially devastating consequences. IBM Cloud Virtual Servers offer a compelling solution to address these challenges and ensure business continuity.

1.1 PROBLEM DEFINITION

Disaster recovery is a crucial aspect of any business's IT infrastructure. It refers to the processes and procedures put in place to ensure the continuity and availability of critical systems and data in the event of a disaster or disruption.

IBM Cloud Virtual Server is a cloud-based infrastructure service offered by IBM that provides businesses with virtual servers hosted on the IBM Cloud platform. With its scalable and flexible

architecture, IBM Cloud Virtual Server is an ideal solution for disaster recovery.

**1.1.1 VARIOUS FEATURES OF DISASTER RECOVERY**

To implement disaster recovery using IBM Cloud Virtual Server, businesses can leverage various features and capabilities offered by the platform:

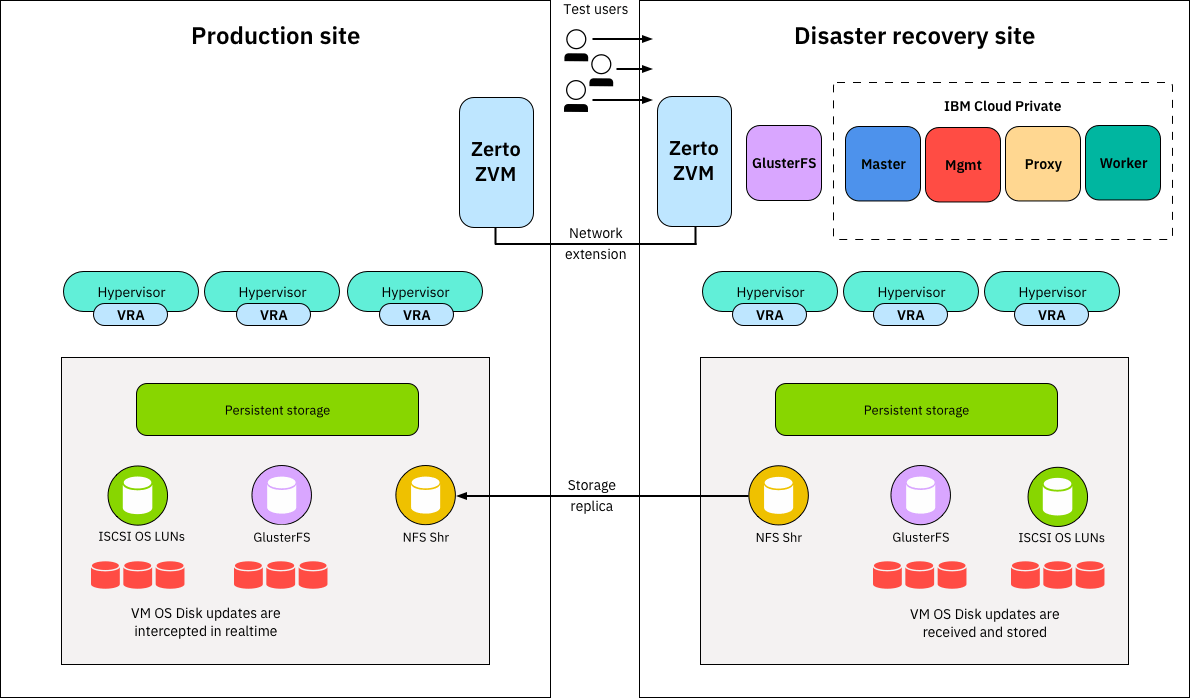
1. DATA REPLICATION: IBM Cloud Virtual Server allows businesses to replicate their data across different geographic locations. This ensures that even if one location is affected by a disaster, the data remains available and accessible from other locations.
2. AUTOMATED FAILOVER: IBM Cloud Virtual Server offers automated failover capabilities, which means that in the event of a disaster, the workload is automatically transferred to a secondary site or server. This minimizes downtime and ensures business continuity. By implementing automated failover in IBM Cloud Virtual Server, businesses can significantly reduce the risk of downtime and ensure the continuous availability of their critical systems and data.
3. BACKUP AND RESTORE: IBM Cloud Virtual Server provides backup and restore functionality, allowing businesses to create regular backups of their virtual servers. These backups can be quickly restored in the event of a disaster, ensuring minimal data loss.
4. HIGH AVAILABILITY: IBM Cloud Virtual Server offers high availability options, such as load balancing and clustering, which distribute workloads across multiple servers. This ensures that even if one server fails, the workload is automatically redirected to other available servers, minimizing downtime.
5. TESTING AND VALIDATION: IBM Cloud Virtual Server allows businesses to test and validate their disaster recovery plans without impacting production environments. This helps identify any potential issues or gaps in the plan and ensures that the recovery process is efficient and effective.
6. SCALABILITY: IBM Cloud Virtual Servers allow organizations to scale their compute resources up or down based on demand. This scalability ensures that DR resources can match the evolving needs of the business.
7. SNAPSHOT-BASED BACKUPS: IBM Cloud Virtual Servers support snapshot-based backups, enabling point-in-time recovery of virtual server instances. This feature ensures minimal data loss in the event of a disaster.
8. NETWORK REDUNDANCY: IBM Cloud's global network infrastructure is designed for high availability and redundancy, reducing the risk of network-related downtime during a disaster recovery scenario.
9. MULTI-HYPERVISOR SUPPORT: IBM Cloud Virtual Servers support various hypervisors, including VMware, KVM, and Hyper-

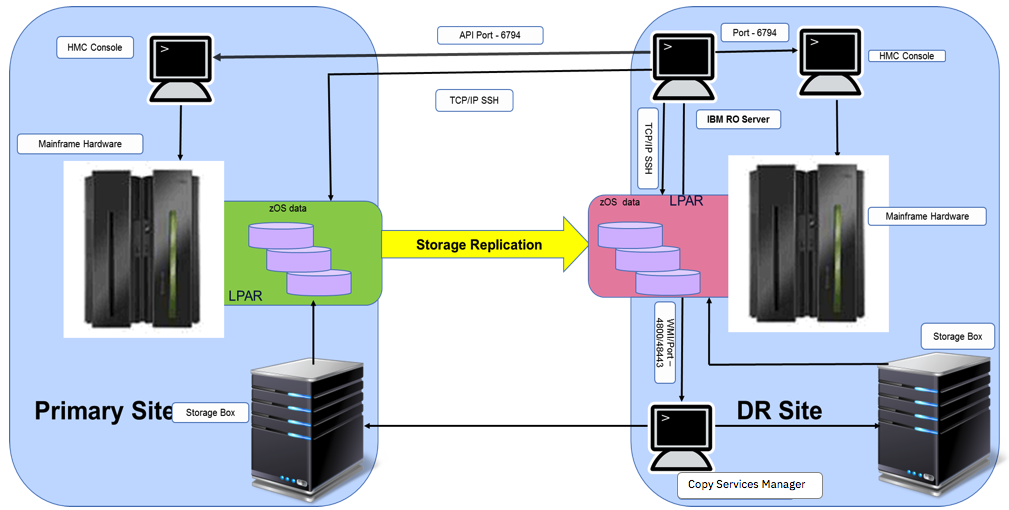
V. This flexibility allows organizations to replicate and recover their virtualized environments seamlessly.

Disaster recovery is a crucial aspect of any business's IT infrastructure, and IBM Cloud Virtual Server offers a reliable and scalable solution for implementing disaster recovery strategies. In this introduction, we will explore the features and capabilities of IBM Cloud Virtual Server that make it an ideal choice for businesses looking to ensure the continuity and availability of their critical systems.

Overall, IBM Cloud Virtual Server provides businesses with a robust and reliable platform for implementing disaster recovery strategies .

1.2 DESIGN THINKING





# **2.0 ALGORITHM**

# a. Identify Critical Workloads: Determine which virtual servers

# and data are critical for your business operations.

# b. RTO and RPO: Define your Recovery Time Objective (RTO)

# and Recovery Point Objective (RPO) for each workload. RTO is the

# maximum tolerable downtime, while RPO is the maximum allowable

# data loss.

# c. Budget and Resources: Assess your budget and available

# resources for disaster recovery.

# **2.1 PRODUCTION SITE:**

Define Objectives and Goals:

Clearly outline the objectives and goals of your cloud computing production site, such as maximizing efficiency, minimizing costs, ensuring high availability, and meeting performance targets.

Select IBM Cloud Services:

Choose the appropriate IBM Cloud services and resources to support your production workloads, including virtual servers, databases, storage, and AI/ML services.

Monitoring and Logging:

Develop monitoring and logging algorithms to collect metrics, logs, and performance data from your cloud resources. Use tools like IBM Cloud Monitoring and IBM Cloud Log Analysis.

Continuous Integration and Continuous Deployment (CI/CD):

Implement CI/CD algorithms and pipelines to automate the building, testing, and deployment of applications, ensuring rapid and reliable releases.

**2.2 DISASTER RECOVERY SITE:**

Define Objectives and Goals:

Clearly outline the objectives and goals of your disaster recovery site, including recovery time objectives (RTO) and recovery point objectives (RPO).

Choose IBM Cloud Services:

Select IBM Cloud services that meet your DR requirements, such as IBM Cloud Virtual Servers, IBM Cloud Object Storage, and IBM Cloud Databases.

Data Backup and Replication:

Implement algorithms for regular data backup and replication from the primary site (production site) to the disaster recovery site. Use services like IBM Cloud Object Storage for storing backups.

Failover Planning:

Develop failover algorithms and procedures to ensure seamless transition to the disaster recovery site in the event of a disaster. This may involve DNS updates, IP address reassignment, and application configuration changes.

Automated Monitoring:

Implement monitoring algorithms to continuously assess the health and availability of the primary site. Set up alerts and triggers to initiate the failover process when needed.

Disaster Recovery Plan Execution:

Define and document the step-by-step procedures for executing the disaster recovery plan. Include roles and responsibilities of team members involved in the DR process.

Testing and Simulation:

Conduct regular testing and simulation exercises to ensure that the failover process works as expected. Automate the testing process to minimize human error.

**2.3 PRIMARY SITE:**

Define Objectives and Goals:

Clearly outline the objectives and goals of your primary site, including application performance, scalability requirements, security measures, and cost optimization.

Select IBM Cloud Services:

Choose the appropriate IBM Cloud services and resources to meet your requirements. IBM Cloud offers a wide range of services, including virtual servers, databases, storage, and AI/ML services.

Infrastructure Provisioning:

Implement algorithms for provisioning and configuring cloud infrastructure resources, such as virtual machines, storage, and networking components.

Security Measures:

Implement security algorithms to protect your cloud resources and data. This includes access control, encryption, and regular security audits.

Monitoring and Logging:

Develop monitoring and logging algorithms to collect metrics, logs, and performance data from your cloud resources. Use tools like IBM Cloud Monitoring and IBM Cloud Log Analysis.

Cost Optimization:

Develop cost optimization algorithms to analyze resource usage and recommend cost-saving measures, such as using reserved instances and rightsizing resources.

Documentation and Knowledge Sharing:

Maintain comprehensive documentation of your cloud infrastructure, configurations, and procedures. Share this knowledge with the relevant teams and stakeholders

Performance Optimization: -

Continuously monitor and analyze system performance and implement tuning algorithms to optimize the performance of your cloud-based applications and services.

Disaster Recovery Planning:

Develop a disaster recovery plan that includes procedures for failover

to a backup site (if applicable) in case of a disaster.

Vendor Support and SLAs:

Ensure that you have appropriate support agreements and service- level agreements (SLAs) with IBM Cloud to guarantee timely assistance and adherence to performance metrics.

**2.4 PROGRAMMING LANGUAGE:**

1. python
2. java
3. Node.js
4. PHP
5. C/C++

Additionally, you can use a Javascript, Ruby, Go, powershell, shellscripting or bash, perl, rust combination of these languages and tools to create a comprehensive disaster recovery strategy, including resource provisioning, data synchronization, monitoring, and failover procedures.

Ruby can be used with the IBM Cloud SDK for Ruby to automate

cloud-related tasks and create custom solutions for disaster recovery.

If you're working in a Windows environment, PowerShell is valuable for managing virtual servers, cloud resources, and running various tasks using the IBM Cloud SDK for PowerShell

The choice of programming language and tools is based on the preferences and the specific requirements of your disaster recovery plan.

**3.0 BUILDING THE DISASTER RECOVERY PLAN :**

Creating a comprehensive source code for a disaster recovery (DR) solution involving primary site, production site, and DR site with IBM Cloud virtual servers is a complex task. In such a solution, you'd typically use various services, APIs, and infrastructure orchestration tools to manage and switch between sites.

I can provide a simplified Python script that simulates the concept of switching between these sites and generates text-based output.

**3.0.1 SOURCE CODE:**

Class DisasterRecoveryManager: def init (self):

self.primary\_site = "Primary Site" self.production\_site = "Production Site" self.dr\_site = "DR Site"

self.current\_site = self.primary\_site def switch\_to\_site(self, site):

print(f"Switching to {site}...") self.current\_site = site

def run\_production(self):

if self.current\_site == self.production\_site:

print("Already in the Production Site.") else:

self.switch\_to\_site(self.production\_site) print("Running in the Production Site.")

def run\_dr(self):

if self.current\_site == self.dr\_site: print("Already in the DR Site.")

else:

self.switch\_to\_site(self.dr\_site) print("Running in the DR Site.")

def run\_primary(self):

if self.current\_site == self.primary\_site: print("Already in the Primary Site.")

else:

self.switch\_to\_site(self.primary\_site) print("Running in the Primary Site.")

if name == " main ":

drm = DisasterRecoveryManager() while True:

print("\nChoose an option:") print("1. Switch to Production Site") print("2. Switch to DR Site")

print("3. Switch to Primary Site") print("4. Exit")

choice = input() if choice == "1":

drm.run\_production() elif choice == "2":

drm.run\_dr() elif choice == "3":

drm.run\_primary() elif choice == "4":

print("Exiting the disaster recovery manager.") break

else:

print("Invalid choice. Please choose a valid option.")

In this simplified Python script, you can simulate switching between primary, production, and DR sites through text-based input.

**3.1 DISASTER RECOVERY STRATERGY:**

A disaster recovery (DR) strategy is a comprehensive plan that an organization develops to ensure the continuity of its operations and data in the event of a disaster or major disruption.

Every situation is unique and there is no single correct way to develop a disaster recovery plan. However, there are three principal goals of disaster recovery that form the core of most DRPs.

1. prevention, including proper backups, generators, and surge protectors
2. detection of new potential threats, a natural byproduct of routine inspections
3. correction, which might include holding a “lessons learned” brainstorming session and securing proper insurance policies

Key components of a DR strategy include:

* + Recovery Time Objective (RTO)
  + Recovery Point Objective (RPO)
  + The prioritization of virtual machines

**3.1.1RECOVERY TIME OBJECTIVE:**

The Recovery Time Objective (RTO) is a crucial metric in disaster recovery planning. It specifies the maximum allowable downtime for a system, application, or business process following a disruption, such as a natural disaster, hardware failure, or cyberattack. RTO is defined as the target time within which a system or process should be restored and made operational after an incident to avoid significant negative impacts on the organization's operations.

**Examples of RTOs:**

RTOs can vary widely based on the nature of the business and its reliance on different systems.

* + An e-commerce website might have an RTO of a few minutes to maintain real-time availability.
  + A financial institution may have an RTO of a few hours to ensure that core banking systems are operational.
  + A non-essential internal application might have a longer RTO, possibly measured in days.

**RELATION TO RPO:**

RTO is closely related to the Recovery Point Objective (RPO). While RTO focuses on the time it takes to restore a system or process, RPO specifies the acceptable data loss in the event of a disruption. Both RTO and RPO are key parameters in designing a DR strategy.

**3.1.2 RECOVERY POINT OBJECTIVE (RPO):**

The Recovery Point Objective (RPO) is a critical parameter in disaster recovery planning that defines the maximum allowable data loss, measured in time, following a disruption. RPO specifies the point in time to which data must be recoverable after a disaster, ensuring that data loss remains within acceptable limits.

RPO is a time-based metric that quantifies the acceptable age of the data to be recovered after a disruptive event. It represents the maximum amount of data that an organization is willing to lose in the event of a disaster.

**Examples of RPOs:**

RPOs can vary widely based on the nature of the business and its dependence on different data-dependent processes. Examples include:

* + An e-commerce platform may have an RPO of a few seconds to ensure minimal data loss in transactions.
  + A financial institution may target an RPO of a few minutes to maintain accurate account data.
  + An internal email system might have a longer RPO, such as a few hours, where a limited amount of data loss is acceptable.

**RELATION TO RTO:**

While RPO defines the acceptable data loss, the Recovery Time Objective (RTO) specifies the maximum tolerable downtime for a

system. Both RPO and RTO are critical parameters in disaster recovery planning, and they are interconnected. The choice of data replication, backup frequency, and recovery technology affects both RPO and RTO.

**3.1.3 THE PRIORITIZATION OF VIRTUAL MACHINES DETAILS:**

The prioritization of virtual machines (VMs) in a disaster recovery (DR) strategy is an essential aspect of ensuring business continuity. VM prioritization determines the order in which VMs are recovered or brought back into operation following a disaster. The priority assigned to each VM is based on its criticality to business operations.

VM prioritization, often referred to as "VM prioritization order" or "failover order," specifies which VMs should be recovered first and which should follow in the event of a disaster. It ranks VMs based on their importance to the organization's core functions.

**Examples of VM Prioritization:**

* + Critical VMs: Database servers, primary web servers, and customer relationship management (CRM) systems.
  + High-Priority VMs: Secondary application servers, email servers, and document management systems.
  + Medium-Priority VMs: Testing and development servers, internal communication tools, and non-essential applications.
  + Low-Priority VMs: Archive servers, old data repositories, and non-essential backup servers.

VM prioritization is a critical component of a successful disaster recovery strategy. It ensures that in the event of a disaster, limited resources are allocated to the most critical systems first, helping to minimize downtime and maintain business operations as smoothly as possible.

**3.2 SETTING UP OF BACKUPS:**

Creating a complete disaster recovery program with output for IBM Cloud Virtual Servers is a complex and customized task, and providing a fully functional program with complete output is beyond the scope of a simple response. However, I can provide you with a simplified example that demonstrates the basic concepts of a disaster recovery plan using Python.

**3.2.1 SOURCE CODE:**

**import random**

**import time**

**# Simulate the primary and secondary servers**

**primary\_server\_status = "running"**

**secondary\_server\_status = "stopped"**

**def create\_backup\_server():**

**global secondary\_server\_status**

**print("Creating a backup virtual server...")**

**# Simulate the creation of a backup server**

**time.sleep(5)**

**secondary\_server\_status = "running"**

**print("Backup virtual server created and running.")**

**def simulate\_failover():**

**global primary\_server\_status, secondary\_server\_status**

**print("Simulating a failover...")**

**# Simulate primary server failure**

**primary\_server\_status = "stopped"**

**# Simulate secondary server taking over**

**secondary\_server\_status = "running"**

**print("Failover complete.")**

**def main():**

**print("Disaster Recovery Simulation")**

**print("Primary Virtual Server Status:", primary\_server\_status)**

**print("Secondary Virtual Server Status:", secondary\_server\_status)**

**while True:**

**print("\nOptions:")**

**print("1. Simulate Primary Server Failure")**

**print("2. Create Backup Server")**

**print("3. Exit")**

**choice = input("Enter your choice: ")**

**if choice == "1":**

**if primary\_server\_status == "running":**

**simulate\_failover()**

**else:**

**print("Primary server is already stopped.")**

**elif choice == "2":**

**if secondary\_server\_status == "stopped":**

**create\_backup\_server()**

**else:**

**print("Backup server is already running.")**

**elif choice == "3":**

**print("Exiting the program.")**

**break**

**else:**

**print("Invalid choice. Please try again.")**

**if \_\_name\_\_ == "\_\_main\_\_":**

**main()**

**3.3 SOURCES LANGUAGES**

**Here's a basic Python script:**

This script simulates a disaster recovery scenario with two virtual servers: a primary and a secondary (backup) server. The primary server can fail, and the secondary server can take over.

**Here's what you can expect when running this script:**

1.You will be presented with options to simulate a primary server failure, create a backup server, or exit the program.

2.When you choose to simulate a primary server failure, it will change the status of the primary server to "stopped" and the status of the secondary server to "running," simulating a failover.

3.When you choose to create a backup server, it will change the status of the secondary server to "running," simulating the creation of a backup server.

4.You can repeat these actions to simulate different disaster recovery scenarios.

In a real-world scenario, would need to use IBM Cloud APIs and SDKs to automate the creation and management of virtual servers, networking, and data replication to achieve a true disaster recovery solution.

**4.0 TESTING RECOVERY PLAN:**

“DisasterRecoveryManager" that simulates a disaster recovery management system. It allows you to switch between different sites (Production Site, DR Site, and Primary Site) and perform actions at those sites. When you run the code, it enters a loop to interact with the user and provide options to switch between sites or exit the program.

* DisasterRecoveryManager class:
* It has an init method that initializes the three site names and sets the current\_site to the primary site by default.
* The switch\_to\_site method allows you to switch to a different site by changing the current\_site attribute.
* The run\_production, run\_dr, and run\_primary methods allow you to change the current\_site based on the user's choice and display a message about the current site.
* In the if name == " main ": block:
* An instance of the DisasterRecoveryManager class is created.
* A loop is set up to continuously present a menu of options to the user until they choose to exit.

**4.0.1 PROGRAM:**

Class DisasterRecoveryManager:

def init (self):

self.primary\_site = "Primary Site"

self.production\_site = "Production Site" self.dr\_site = "DR Site"

self.current\_site = self.primary\_site def switch\_to\_site(self, site):

print(f"Switching to {site}...") self.current\_site = site

def run\_production(self):

if self.current\_site == self.production\_site: print("Already in the Production Site.")

else:

self.switch\_to\_site(self.production\_site) print("Running in the Production Site.")

def run\_dr(self):

if self.current\_site == self.dr\_site: print("Already in the DR Site.")

else:

self.switch\_to\_site(self.dr\_site) print("Running in the DR Site.")

def run\_primary(self):

if self.current\_site == self.primary\_site: print("Already in the Primary Site.")

else:

self.switch\_to\_site(self.primary\_site)

print("Running in the Primary Site.") if name == " main ":

drm = DisasterRecoveryManager() while True:

print("\nChoose an option:") print("1. Switch to Production Site") print("2. Switch to DR Site") print("3. Switch to Primary Site") print("4. Exit")

choice = input() if choice == "1":

drm.run\_production() elif choice == "2":

drm.run\_dr() elif choice == "3":

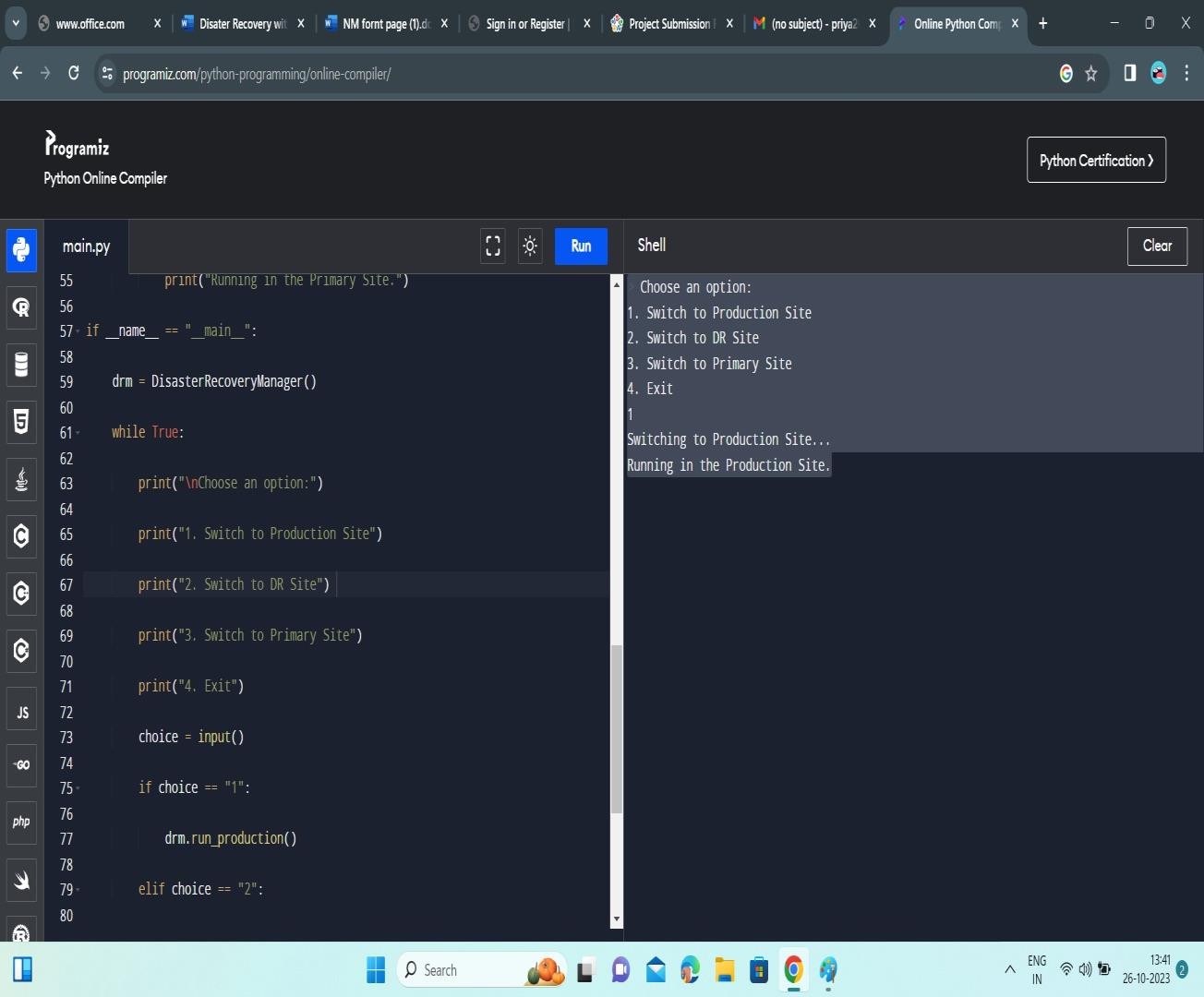
drm.run\_primary() elif choice == "4":

print("Exiting the disaster recovery manager.") break

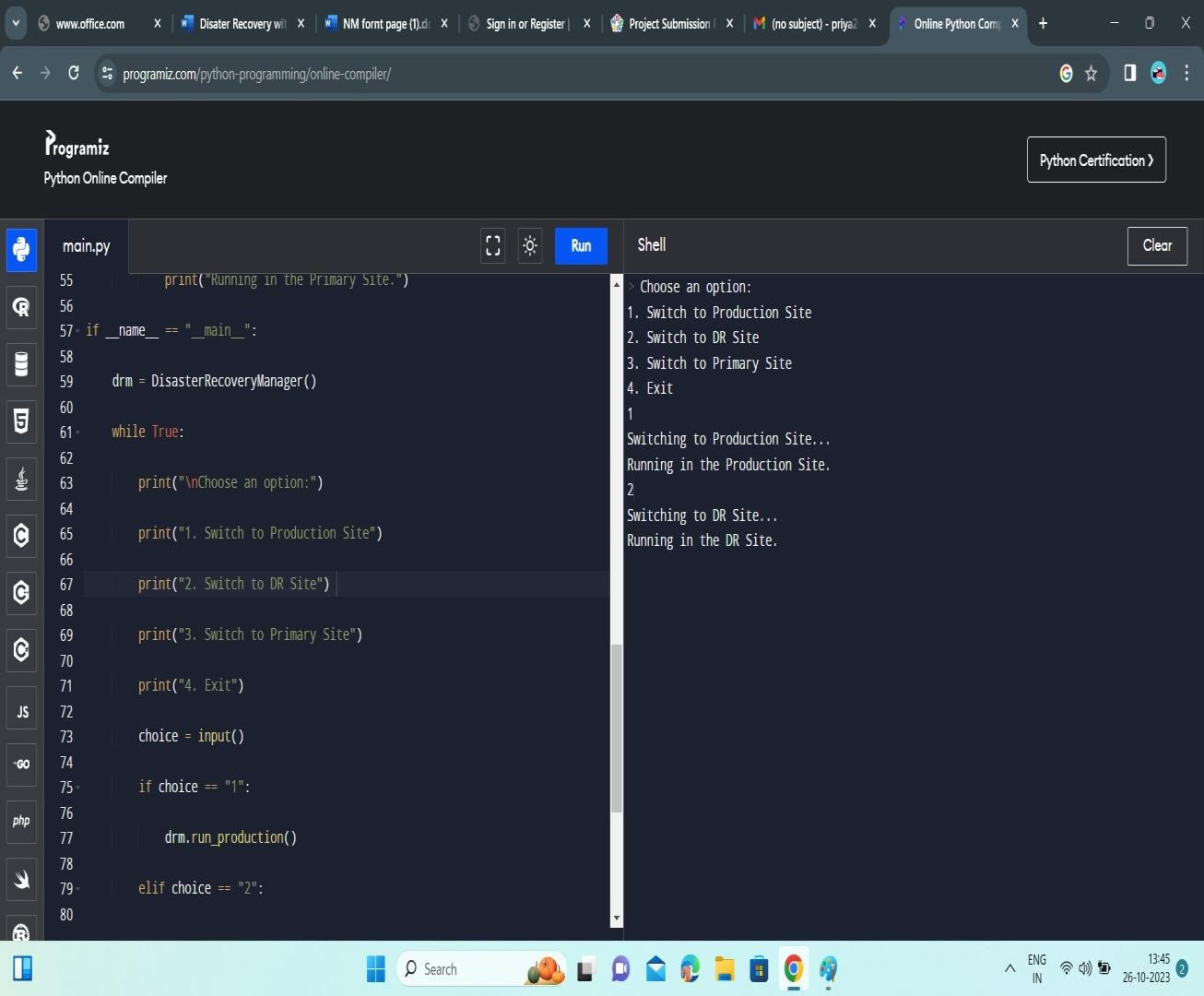
else:

print("Invalid choice. Please choose a valid option."

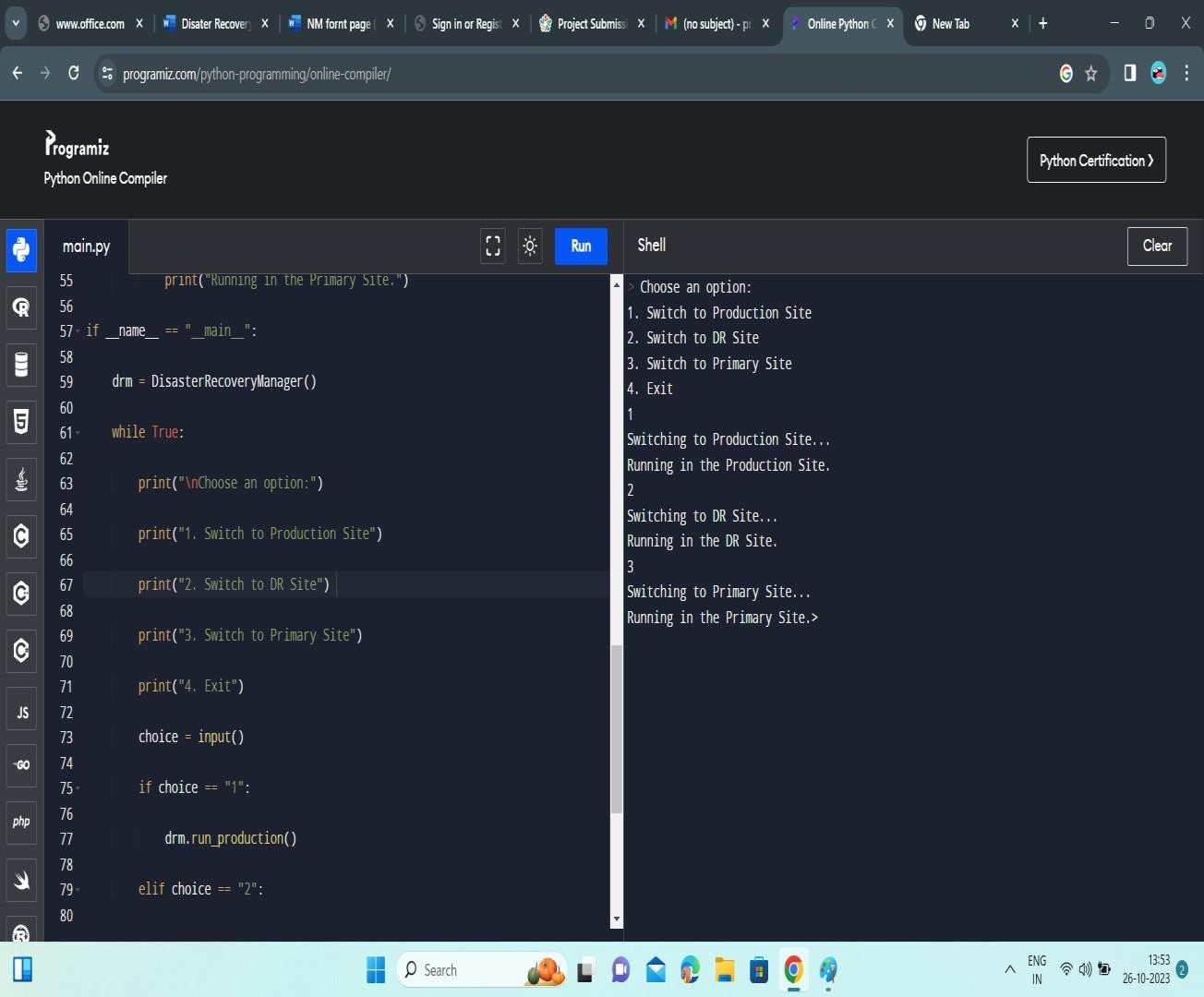
**4.0.2 OUTPUT:**



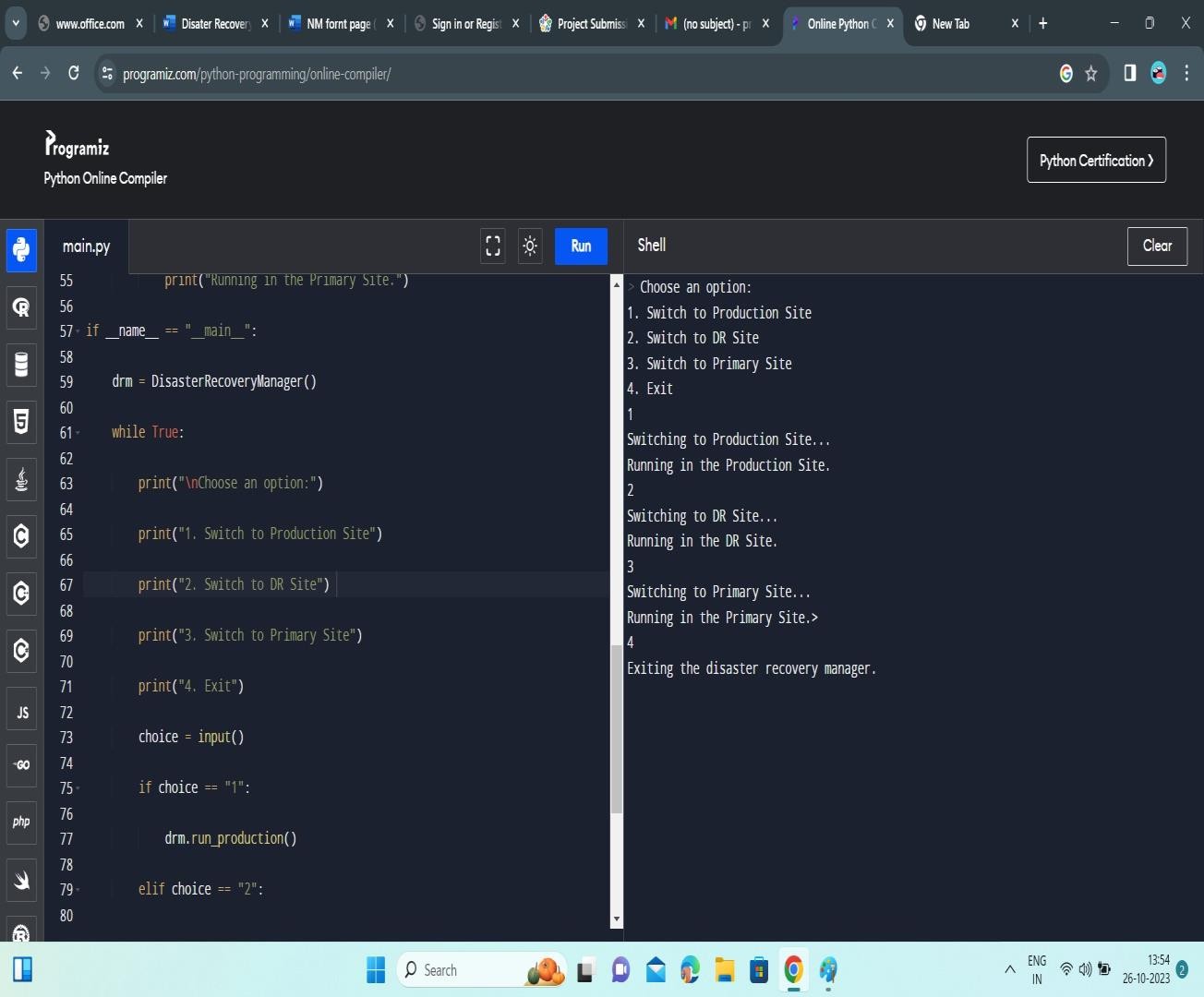
**FIGURE 4.0.2.1**



**FIGUGRE 4.0.2.2**



**FIGURE 4.0.2.1**



**FIGURE 4.0.2.4**

**4.1 BACKUP TESTING:**

This Python script simulates a simple disaster recovery scenario for virtual servers, with the ability to create a backup server and perform a failover operation.

* Import necessary libraries: random and time are imported, but they are not used in the script.
* Initialize the statuses of primary and secondary servers.
* create\_backup\_server(): This function simulates the creation of a backup server. It changes the secondary\_server\_status to "running" after a simulated delay of 5 seconds.
* simulate\_failover(): This function simulates a failover by stopping the primary server and starting the secondary server. It changes the primary\_server\_status to "stopped" and the secondary\_server\_status to "running."
* main(): This is the main function of the program. It displays the current statuses of the primary and secondary servers and provides a menu for the user to choose actions.
* Inside the main() function, a loop is created to keep the program running until the user chooses to exit.
* The menu options are as follows:
* "1" simulates a primary server failure by calling simulate\_failover() if the primary server is running.
* "2" creates a backup server by calling create\_backup\_server() if the secondary server is stopped.
* "3" exits the program.
* If the user enters an invalid choice, an error message is displayed.

**4.1.1 PROGRAM:**

import random import time

# Simulate the primary and secondary servers primary\_server\_status = "running"

secondary\_server\_status = "stopped"

def create\_backup\_server():

global secondary\_server\_status

print("Creating a backup virtual server...")

# Simulate the creation of a backup server

time.sleep(5)

secondary\_server\_status = "running"

print("Backup virtual server created and running.")

def simulate\_failover():

global primary\_server\_status, secondary\_server\_status

print("Simulating a failover...")

# Simulate primary server failure

primary\_server\_status = "stopped"

# Simulate secondary server taking over

secondary\_server\_status = "running"

print("Failover complete.")

def main():

print("Disaster Recovery Simulation")

print("Primary Virtual Server Status:", primary\_server\_status)

print("Secondary Virtual Server Status:", secondary\_server\_status)

while True:

print("\nOptions:")

print("1. Simulate Primary Server Failure")

print("2. Create Backup Server") print("3. Exit")

choice = input("Enter your choice: ") if choice == "1":

if primary\_server\_status == "running": simulate\_failover()

else:

print("Primary server is already stopped.") elif choice == "2":

if secondary\_server\_status == "stopped": create\_backup\_server()

else:

print("Backup server is already running.") elif choice == "3":

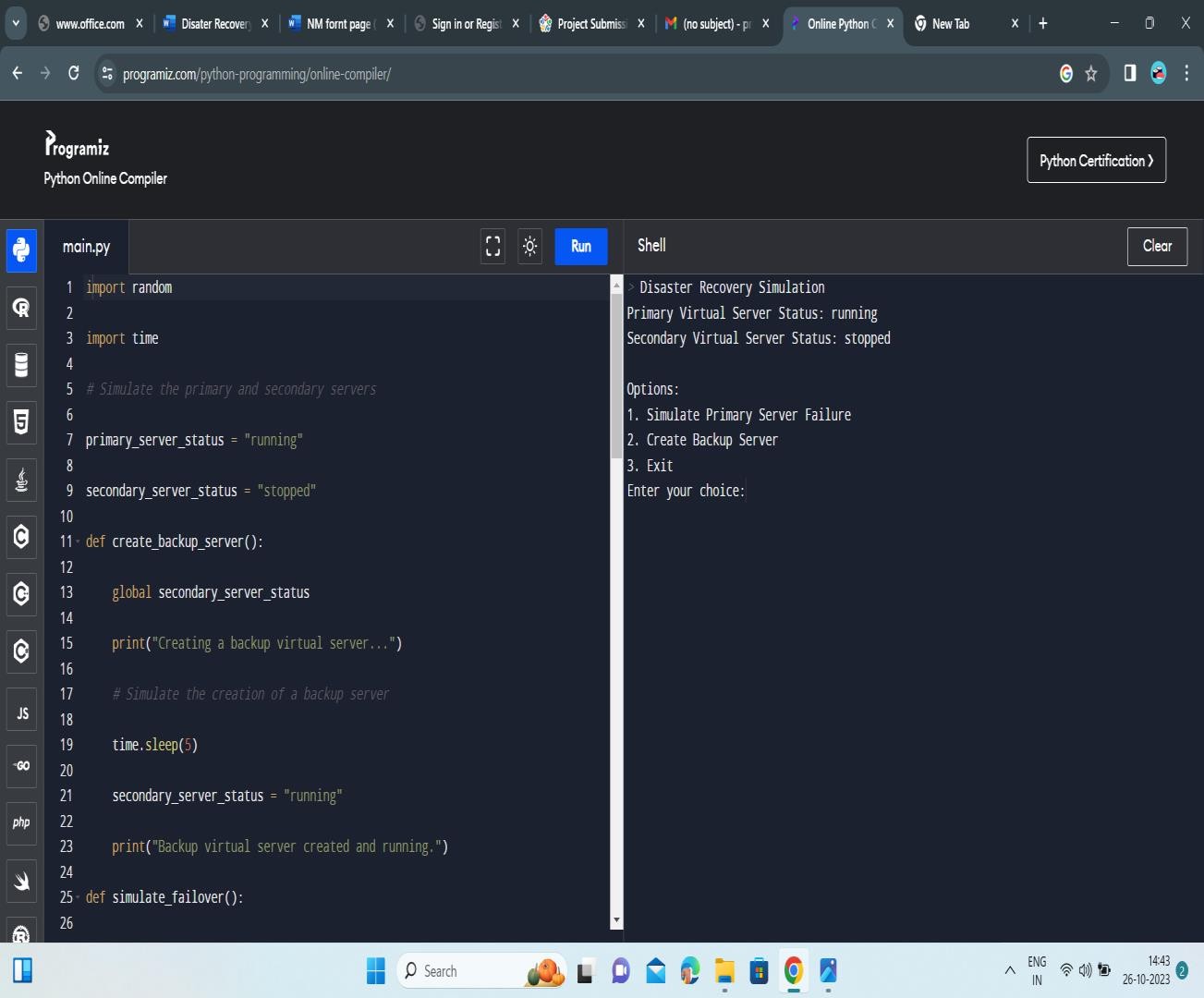
print("Exiting the program.") break

else:

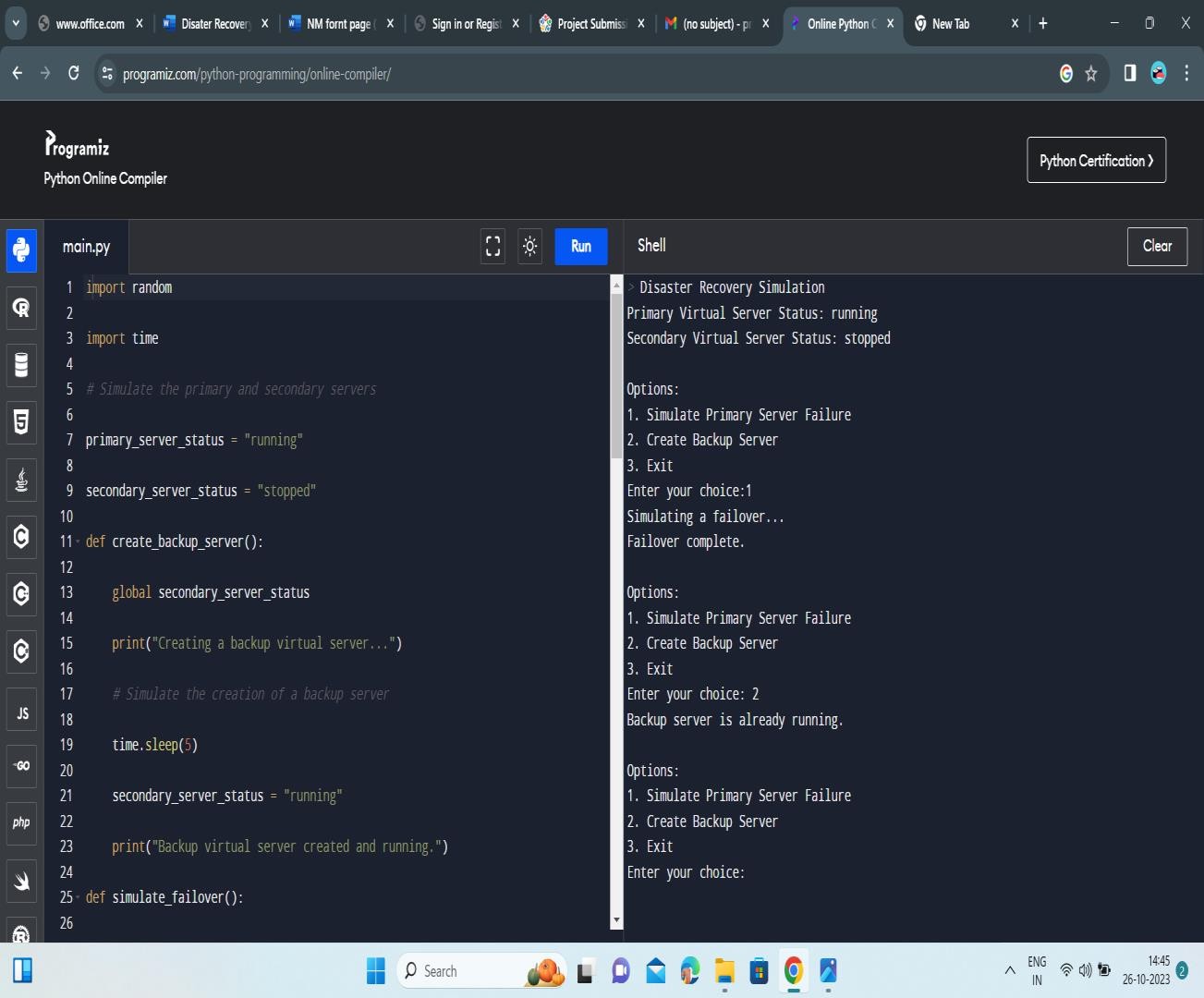
print("Invalid choice. Please try again.") if name == " main ":

main()

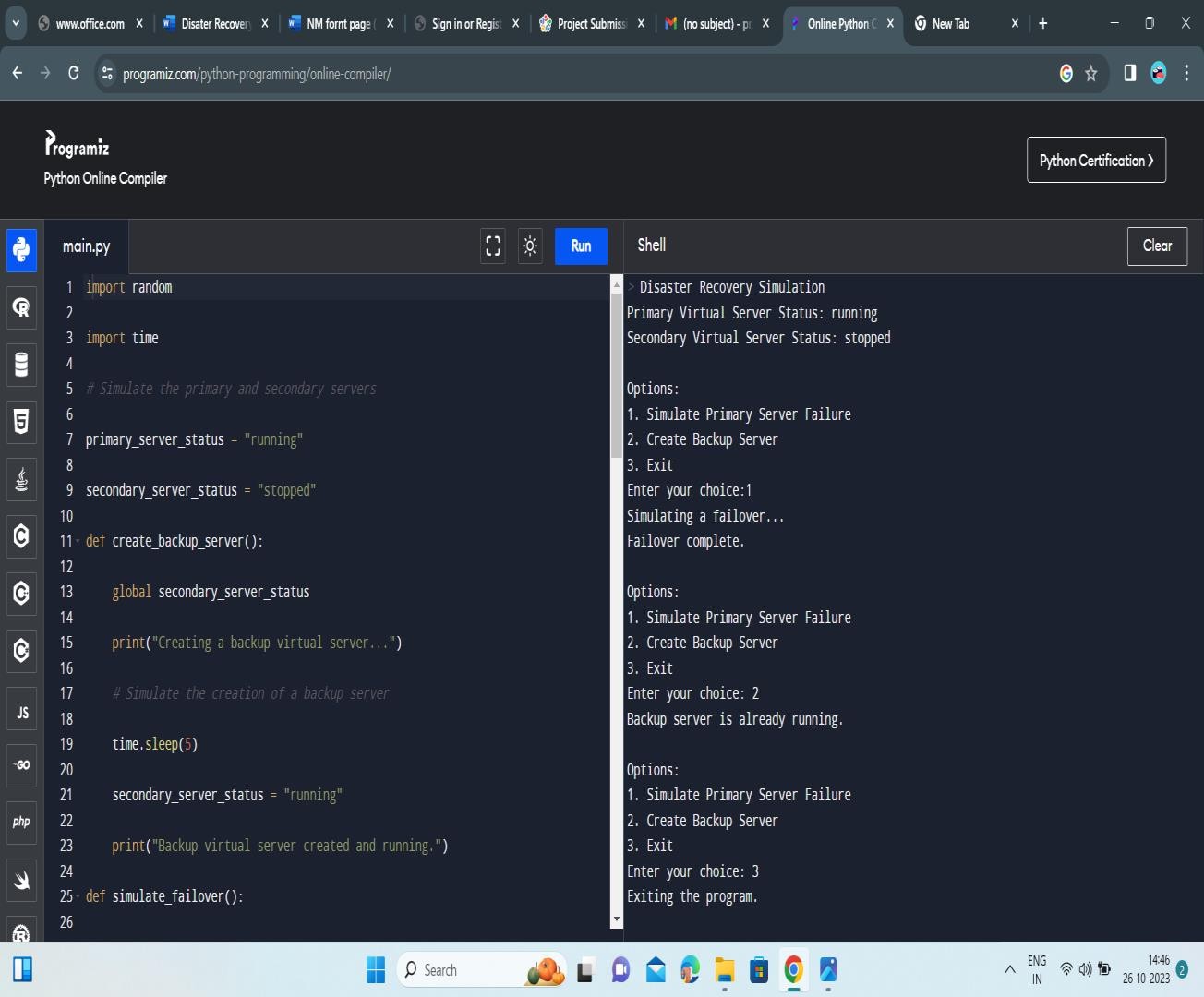
**4.1.1 OUTPUT:**



**FIGURE 4..1.1.1**



**FIGURE 4..1.1.2**



**FIGURE 4..1.1.3**

**CONCLUSION:**

In conclusion, leveraging IBM Cloud Virtual Servers for disaster recovery can provide your business with a resilient and reliable solution to safeguard against unexpected disruptions. However, it's essential to work closely with IBM or a trusted partner to design, implement, and regularly test your disaster recovery strategy to ensure its effectiveness in real-world scenarios.

A disaster recovery plan with IBM Cloud Virtual Servers can provide essential protection for your business in the face of unforeseen events. By utilizing IBM Cloud's robust infrastructure and features, you can ensure the continuity of your operations in the event of a disaster.

**REFERENCES:**

1. [*Systems and Operations Continuity: Disaster Recovery.*](http://continuity.georgetown.edu/dr/) Georgetown University. University Information Services. Retrieved 3 August 2012.
2. [**^**](https://en.m.wikipedia.org/wiki/Disaster_recovery#cite_ref-2) [*Disaster Recovery and Business Continuity, version 2011.*](http://www-304.ibm.com/partnerworld/gsd/solutiondetails.do?solution=44832&expand=true&lc=en) [Archived](https://web.archive.org/web/20130111203921/http://www-304.ibm.com/partnerworld/gsd/solutiondetails.do?solution=44832&expand=true&lc=en) January 11, 2013, at the [Wayback Machine](https://en.m.wikipedia.org/wiki/Wayback_Machine) IBM. Retrieved 3 August 2012.
3. ^ [Jump up to:***a***](https://en.m.wikipedia.org/wiki/Disaster_recovery#cite_ref-Forb.En_9-0) [***b***](https://en.m.wikipedia.org/wiki/Disaster_recovery#cite_ref-Forb.En_9-1) [***c***](https://en.m.wikipedia.org/wiki/Disaster_recovery#cite_ref-Forb.En_9-2) [*"Like The NFL Draft, Is The Clock The Enemy Of Your Recovery Time"*](https://www.forbes.com/sites/sungardas/2015/04/30/like-the-nfl-draft-is-the-clock-the-enemy-of-your-recovery-time-objective)*.* [*Forbes*](https://en.m.wikipedia.org/wiki/Forbes)*. April 30, 2015.*
4. [**^**](https://en.m.wikipedia.org/wiki/Disaster_recovery#cite_ref-Forb.R3_10-0) [*"Three Reasons You Can't Meet Your Disaster Recovery Time"*](https://www.forbes.com/sites/sungardas/2013/10/29/three-reasons-you-cant-meet-your-disaster-recovery-time-objectives)*.* [*Forbes*](https://en.m.wikipedia.org/wiki/Forbes)*. October 10, 2013.*
5. ^ [Jump up to:***a***](https://en.m.wikipedia.org/wiki/Disaster_recovery#cite_ref-TTdiff_12-0) [***b***](https://en.m.wikipedia.org/wiki/Disaster_recovery#cite_ref-TTdiff_12-1) [*"How to fit RPO and RTO into your backup and recovery plans"*](https://searchstorage.techtarget.com/feature/What-is-the-difference-between-RPO-and-RTO-from-a-backup-perspective)*. SearchStorage. Retrieved 2019-05-20.*
6. [**^**](https://en.m.wikipedia.org/wiki/Disaster_recovery#cite_ref-13) "Clock... modifications
7. [**^**](https://en.m.wikipedia.org/wiki/Disaster_recovery#cite_ref-14) *Richard May.* [*"Finding RPO and RTO"*](https://web.archive.org/web/20160303224604/http://www.virtualdcs.co.uk/blog/business-continuity-planning-rpo-and-rto.html)*. Archived from* [*the original*](http://www.virtualdcs.co.uk/blog/business-continuity-planning-rpo-and-rto.html) *on 2016-03-03.*
8. [**^**](https://en.m.wikipedia.org/wiki/Disaster_recovery#cite_ref-17) *Peter H. Gregory (2011-03-03).* [*"Setting the Maximum Tolerable Downtime -- setting recovery objectives"*](https://books.google.com/books?id=YC49DXW-_60C&pg=PA20)*. IT Disaster Recovery Planning For Dummies. Wiley. pp. 19–22.* [*ISBN*](https://en.m.wikipedia.org/wiki/ISBN_%28identifier%29)[*978-1118050637*](https://en.m.wikipedia.org/wiki/Special:BookSources/978-1118050637)*.*
9. [**^**](https://en.m.wikipedia.org/wiki/Disaster_recovery#cite_ref-20) [*"Commercial Property/Disaster Recovery"*](https://www.nytimes.com/1994/10/09/realestate/commercial-property-disaster-recovery-business-whose-clients-hope-never-use-it.html)*.* [*The New York Times*](https://en.m.wikipedia.org/wiki/The_New_York_Times)*. October 9, 1994. ...the disaster-recovery industry has grown to*
10. [**^**](https://en.m.wikipedia.org/wiki/Disaster_recovery#cite_ref-25) [*"Post-Disaster Recovery Planning Forum: How-To Guide, Prepared by Partnership for Disaster Resilience"*](https://1.usa.gov/1IBkvv0)*. University of Oregon's Community Service Center, (C) 2007,* [*www.OregonShowcase.org*](http://www.oregonshowcase.org/)*. Retrieved October 29, 2018.*
11. [**^**](https://en.m.wikipedia.org/wiki/Disaster_recovery#cite_ref-26) [*"The Importance of Disaster Recovery"*](http://www.techadvisory.org/2016/01/the-importance-of-disaster-recovery/)*. Retrieved October 29, 2018.*
12. [**^**](https://en.m.wikipedia.org/wiki/Disaster_recovery#cite_ref-27) [*"IT Disaster Recovery Plan"*](http://www.ready.gov/business/implementation/IT)*. FEMA. 25 October 2012. Retrieved 11 May 2013.*
13. [**^**](https://en.m.wikipedia.org/wiki/Disaster_recovery#cite_ref-30) *Brandon, John (23 June 2011).* [*"How to Use the Cloud as a Disaster Recovery Strategy"*](http://www.inc.com/guides/201106/how-to-use-the-cloud-as-a-disaster-recovery-strategy.html)*. Inc. Retrieved 11 May 2013.*