National College of Ireland

Configuration Manual

MSc Research Project
MSc. in Cloud Computing

Nipun Bakshi
Student ID: X23202513

School of Computing
National College of Ireland

Supervisor: Yasantha Samarawickrama

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Nipun Bakshi
Student ID:	X23202513
Programme:	MSc. in Cloud Computing **Year:** 2024-2025
Module:	Research Project
Supervisor:	Yasantha Samarawickrama
Submission Due Date:	12th Dec 2024
Project Title:	Advancing the Gaming Industry through Hybrid Computational Strategies
Word Count:	1407 **Page Count:** 12

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Nipun Bakshi

Date: 12/12/2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Nipun Bakshi
Student ID: X23202513

1 Setting up the Cloud Environment

This section provides how to configure and deploy the game on cloud environment using AWS services.

1.1 An EC2 Windows Instance Creation

1. Login to AWS Management Console
2. Go to the EC2 dashboard then click Launch Instance.

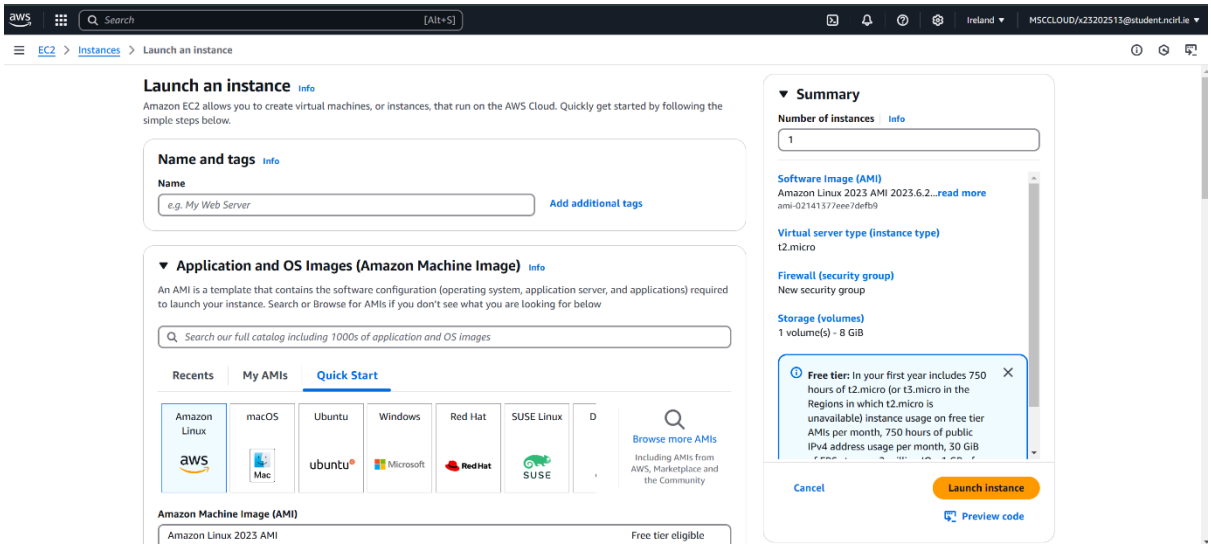

Fig.1 AWS EC2 instance Creation page

3. Select an AMI (Amazon Machine Image)
4. Pick one Windows Server AMI (like Windows Server 2019).
5. Choose Instance Type- For balanced performance select t2.xlarge.
6. Create a new Key pair
7. Change the numbers of instances to 1.
8. Enable Auto-assign Public IP for access from the external.
9. Add Storage (like 30GB).
10. Select: Create security group
11. Configure Instance Details and select “Launch Instance”
12. The key pair will be generated and will be downloaded to access the instance.
13. Once created, go to instance page and select “Security groups”

14. Add new inbound rules, allow
    - RDP (port 3389)
    - HTTP (port 80)
    - Prometheus Server (9090)
    - WMI Exporter (9182)
    - Grafana (3000)

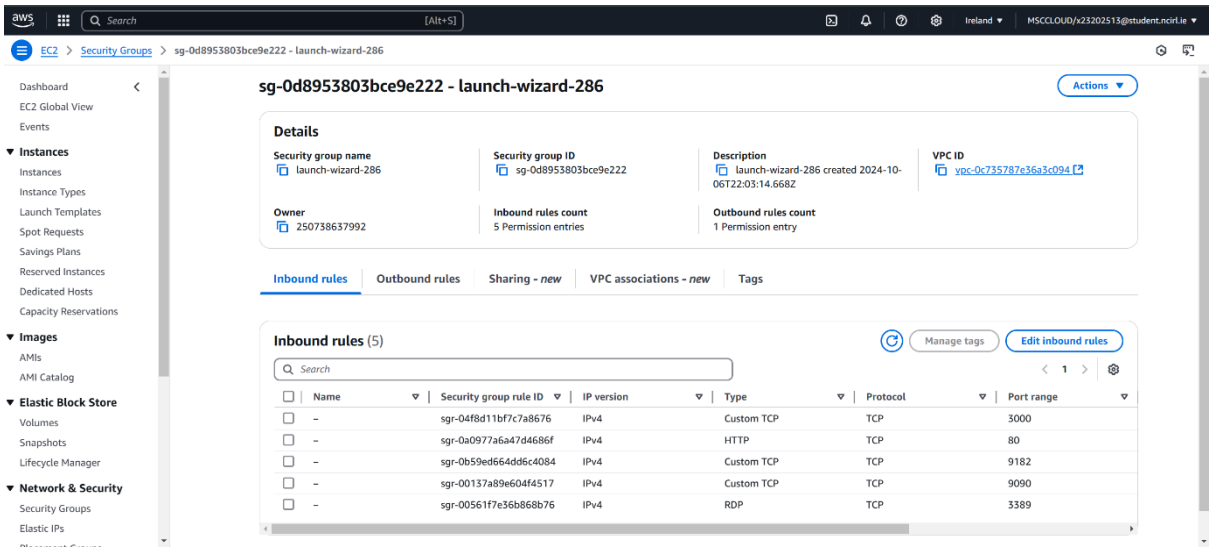

Fig.2 Added inbound Rules

15. Launch The Instance and connect to it via RDP using the generated .pem key.

2 Setting Up Hybrid Environment

2.1 Set up S3 Bucket

1. Go to AWS S3 and click on “create Bucket”
2. Give bucket a name and make sure “Block all public access” option is checked.

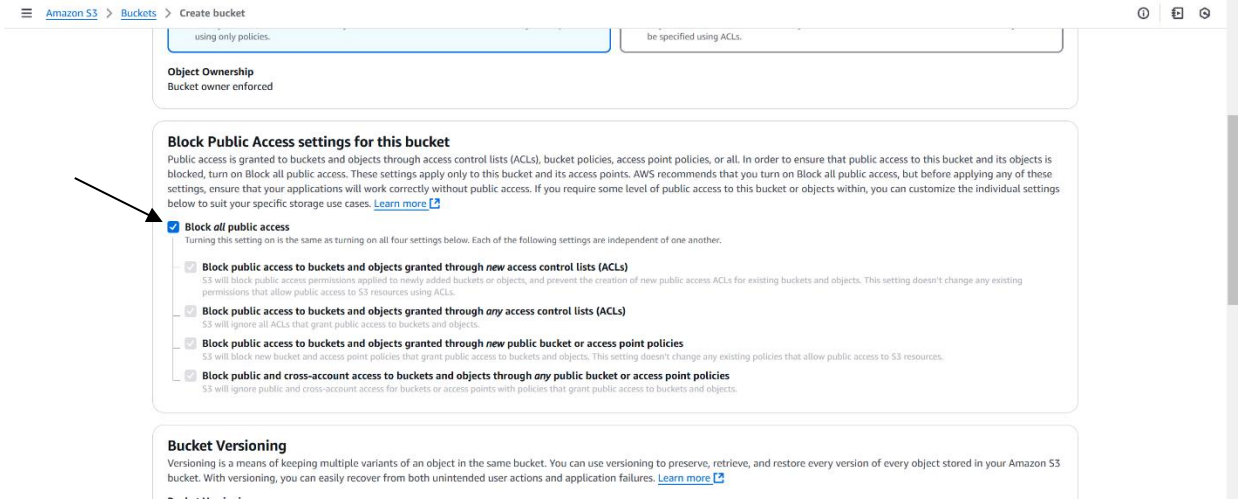

Fig.3 Making Sure Bucket is NOT Public

3. Create the bucket and upload all the files which will be served on the edge to this bucket.

2.2 Configuring CloudFront for File Delivery via Edge

1. Go to CloudFront CDN service
2. Choose Web Distribution and click Create Distribution.

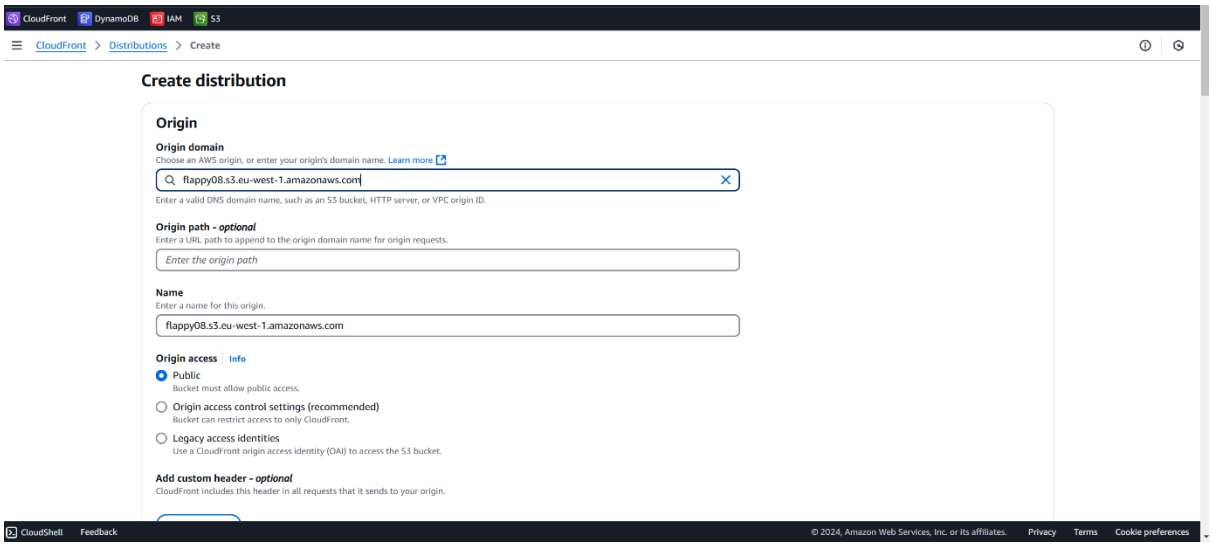

Fig.4 Create Distribution page

3. Set Origin Settings
4. Select the S3 Bucket as the origin.
    - Caches dynamic assets (and images/sound files static assets).
5. Set allowed HTTP methods: GET, HEAD.
6. Select Cache Policy: “CachingOptimized”
7. Enable WAF security protection
8. Deploy CloudFront by clicking on “Create Distribution”
9. Go to the Distribution details page and copy the ARN of the distribution.
10. Go to S3 bucket > permissions and then edit bucket policy.

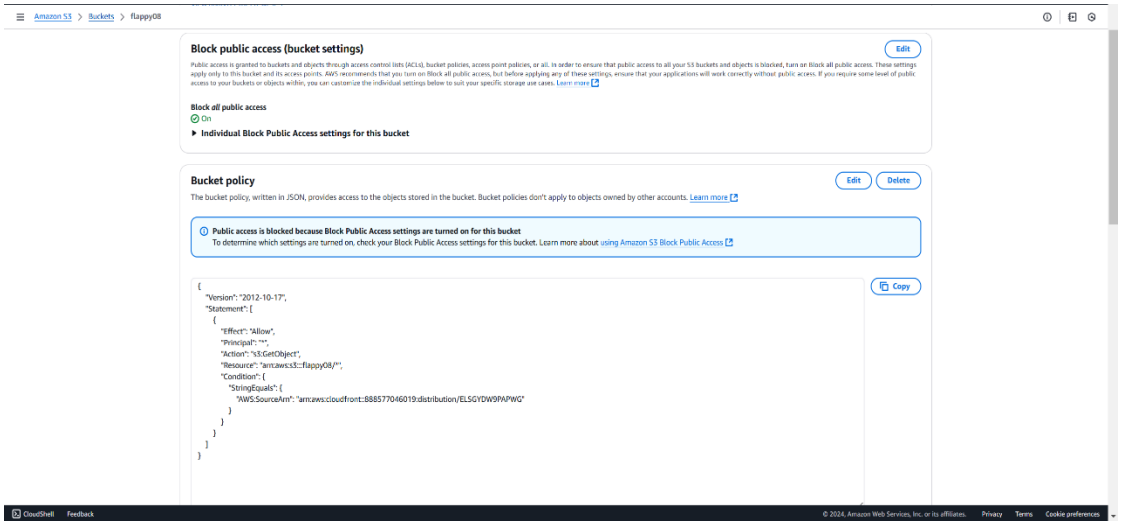

Fig. 5 Editing Bucket Policy to keep the files secure

11. Edit the policy to allow only specific cloudfront distribution to access the bucket.
12. CloudFront is properly set up now and is ready for use.
13. Then go back to CloudFront page and copy the provided distribution domain for integration with the game code.

2.3 Creating DynamoDB with Database of Player Scores

1. Navigate to DynamoDB dashboard and got to Create Table.
2. Define the Schema
   - Primary Key: PlayerID (string).
   - Sort Key: Score (string).
3. Enable On-Demand Mode- It brings scalability without capacity set management.

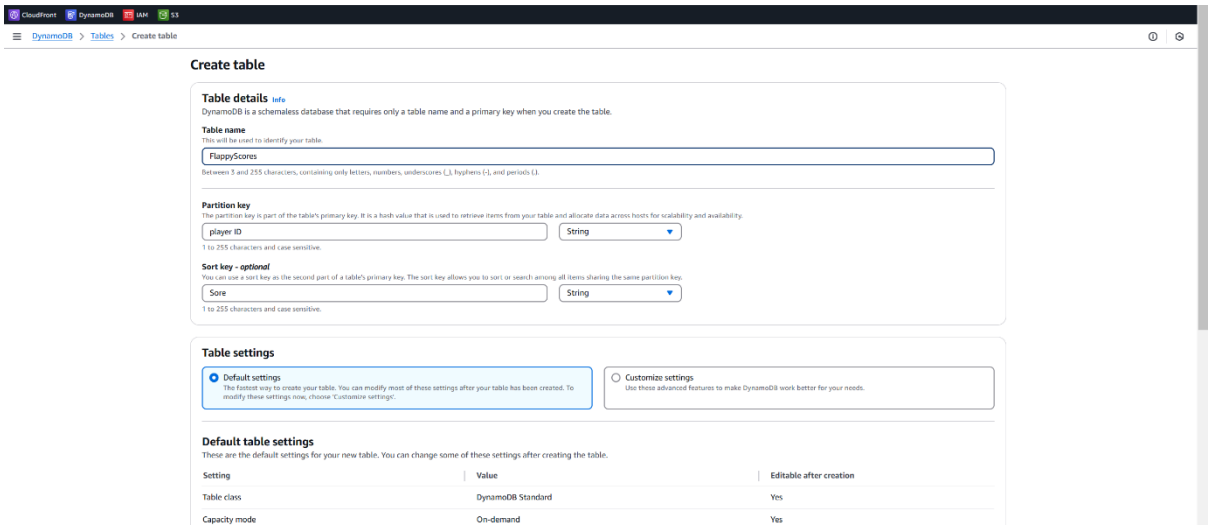

Fig. 6 Creating a DynamoDB Table

4. To Integrate with the Game, will require AWS “access key” and “secret access key”
5. Will save, retrieve scores using AWS SDK (boto3) written in Python

3 Setting up monitoring and visualisation tools

3.1 Installing Prometheus

1. Download the Prometheus binary from the official site.
2. It must be setup on and used by all the 3 architectures (local, hybrid, cloud)
3. Configure Prometheus:
4. Open command prompt in the “Prometheus” root directory. Use following start command and pass web listen address and config file default Prometheus yml file as options.

```
prometheus.exe --config.file prometheus.yml --web.listen-address ":9090" --
storage.tsdb.path "data"
```

5. Prometheus is now running and can be accessed on https://localhost:9090/targets to check whether targets are being successfully scraped or not.

3.2 Setting Up WMI Exporter

1. Download WMI Exporter from WMI Exporter GitHub
2. Install WMI Exporter
3. After installation, verify if Exporter is running

Fig. 7 make sure WMI Exporter is running change startup type to “automatic”

4. Exporter should start exposing metrics on

```
http://localhost:9182/metrics
```

5. Update the Prometheus.yml file with below code and restart Prometheus server:

```
- job_name: "WMI Exporter"

  # metrics_path defaults to '/metrics'
  # scheme defaults to 'http'.

  static_configs:
    - targets: ["Host_ip:9182"]
```

6. Access the metrics endpoint by navigating to <EC2-Instance-IP/Private-IP>:9182/metrics in a browser.
7. Now, https://localhost:9090/targets should show something like this

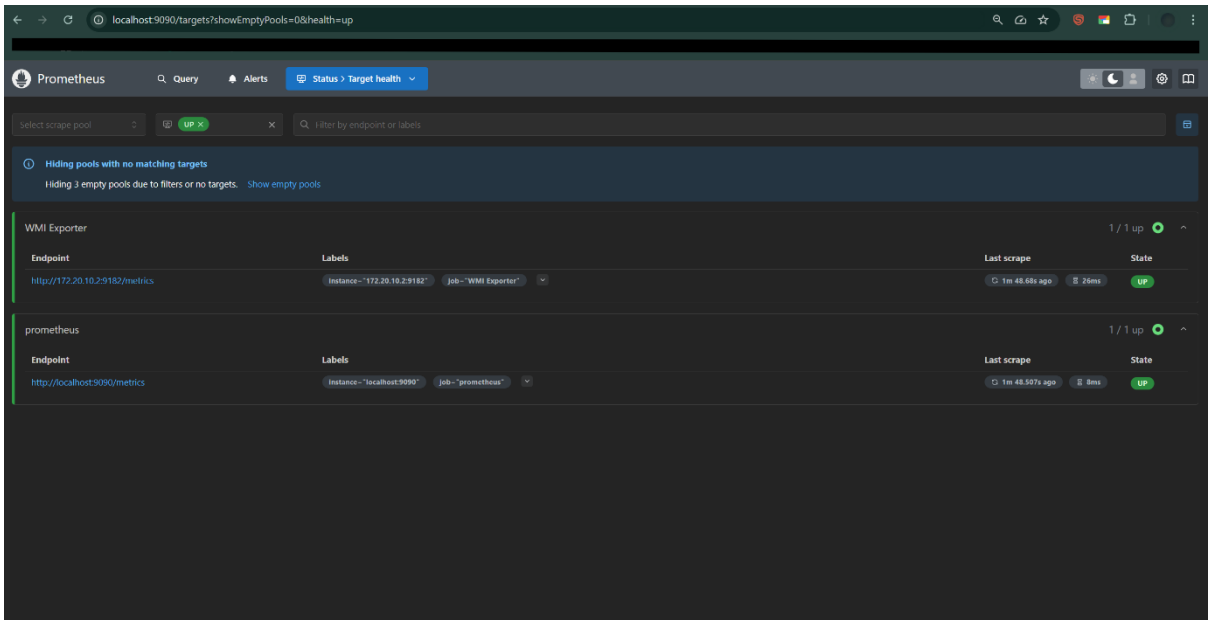

Fig. 8 WMI exporter up and running

3.3 Metrics Collection

1. In the python game code, define functions which exposes the specified metric to an endpoint. For e.g. to send network latency info :

```
Codeium: Refactor | Explain | ×
async def measure_network_latency(self):
    """Measure the network latency by sending a request to a specified endpoint."""
    url = "http://localhost:8000/metrics"  # Change this to your target URL

    try:
        async with aiohttp.ClientSession() as session:
            start_time = time.time()  # Start the timer
            async with session.get(url) as response:
                await response.text()  # Await the response to complete
                end_time = time.time()  # End the timer

            # Calculate latency in milliseconds
            latency = (end_time - start_time) * 1000
            self.network_latency_metric.set(latency)  # Set the Prometheus metric
    except Exception as e:
        print(f"Failed to measure network latency: {e}")
```

Fig. 9 Sending Network Latency data to be scraped

2. Update the prometheus.xml config file to this code:

```yaml
global:
  scrape_interval: 1s
  evaluation_interval: 1s

alerting:
  alertmanagers:
    - static_configs:
        - targets: ["<alertmanager-ip>:9093"]

rule_files:
  # Define your alert rules here
  # - "alert_rules.yml"

scrape_configs:
  - job_name: "prometheus"
    static_configs:
      - targets: ["localhost:9090"]

  - job_name: "WMI Exporter"
    static_configs:
      - targets: ["172.20.10.2:9182"]

  - job_name: 'flappybird_fps'
    static_configs:
      - targets: ['172.20.10.2:8000']

  - job_name: 'flappybird_network_latency'
    static_configs:
      - targets: ['172.20.10.2:8000']

  - job_name: 'flappybird_bandwidth_usage'
    static_configs:
      - targets: ['172.20.10.2:8000']
```

Fig. 10 Prometheus Collect all data to be scraped

3. This indicates that all the game metrics are accessed by Prometheus on port:8000

4. This should how the Prometheus Dashboard should look like now

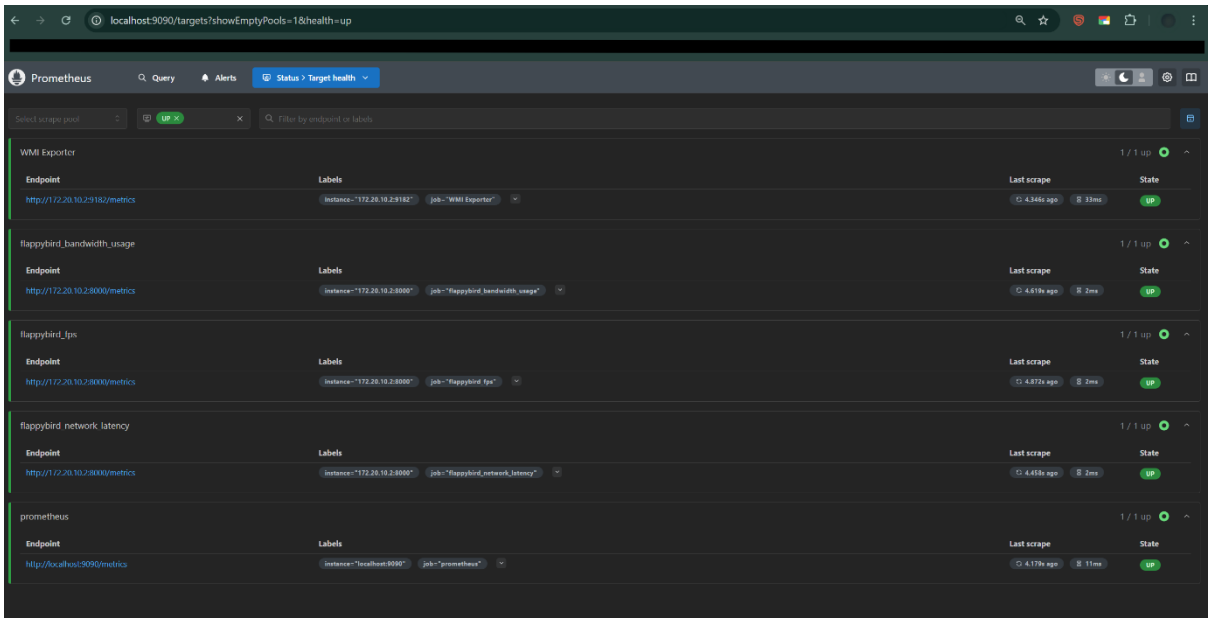

Fig. 11 All targets are in UP state and giving input to Prometheus

3.4 Configuring Grafana

1. Download Grafana from grafana.com and install it.
2. go to C: > Program Files > GrafanaLabs > Grafana > conf > defaults.ini

```
################################### SMTP / Emailing ##########################
[smtp]
enabled = true
```

   Change enabled = true
3. Grafana dashboard can be now accessed at http://localhost:3000
4. In the login Page, enter “admin” in both username and password fields
5. Go to Configuration > Data Sources > Add Data Source.
6. Select Prometheus and provide the server URL (http://<EC2-Instance-IP/ Private-IP >:9090).
7. Go to Dashboards > New Visualisation > select “Prometheus” in data source.
8. Go to the “Queries” tab > select your metric and go to “run queries”.

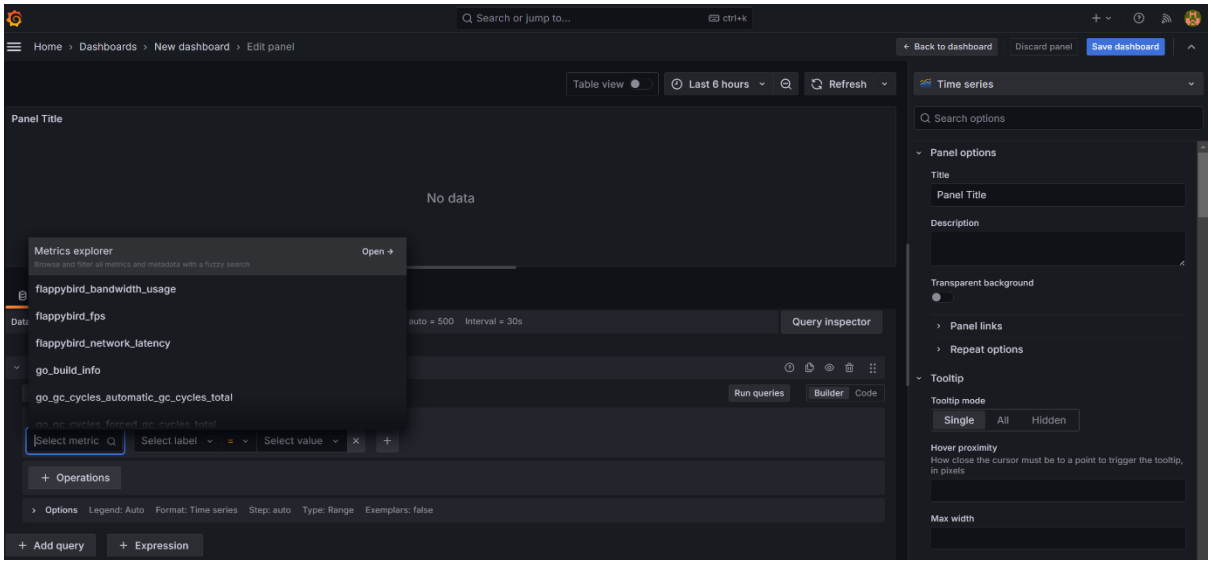

Fig.12 adding metrics for visualisation

9. Prometheus queries can add panels for latency, FPS, and bandwidth.
10. This will create a live graph that will keep updating as long as it is receiving statistics from the game.
11. After adding all the Metrics for visualisation, dashboard should look like this.

Fig. 13 live Graphs

4 Running and Monitoring the Game

4.1 Running the game on cloud

1. Deploy Game Files- Upload the game assets and code to EC2 instance.
2. Start the EC2 Game Server
3. Run the game code with Python.
4. Command: python main.py

4.2 Monitoring Game Performance

1. Grafana Visualizing Metrics
2. Go to Grafana dashboard.
3. See the monitor latency, FPS, and bandwidth usage in real time.
4. Grafana offers a feature to save the monitoring info in csv file format for further analysis

References

Mamidwar, S. and Juneja, G.D.S. (2023) Install prometheus and Grafana with WMI exporter on window server 2022 base, FOSS TechNix. Available at: https://www.fosstechnix.com/install-prometheus-and-grafana-with-wmi-exporter-on-window-server-2022-base/ (Accessed: 10 December 2024).

Get started with Amazon EC2 - Amazon Elastic Compute Cloud. Available at: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html (Accessed: 10 December 2024).

Rani, E.G. and Chetana, D.T. (2023) 'Using GitHub and Grafana tools: Data Visualization (DATA VIZ) in big data,' in Algorithms for intelligent systems, pp. 477–491. https://doi.org/10.1007/978-981-19-7892-0_38.

Singh, A.P., Rai, E.P. and Dixit, A.K., 2023, August. Web Content Distribution with Low Latency on Worldwide Using Amazon CloudFront Service. In 2023 7th International Conference On Computing, Communication, Control And Automation (ICCUBEA) (pp. 1-6). IEEE.

Setting up DynamoDB - Amazon DynamoDB (no date). https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/SettingUp.html.

Getting started with Amazon S3 - Amazon Simple Storage Service (no date). https://docs.aws.amazon.com/AmazonS3/latest/userguide/GetStartedWithS3.html.

Skatteetaten (no date) GitHub - Skatteetaten/wmi_exporter: Prometheus exporter for Windows machines using WMI. https://github.com/Skatteetaten/wmi_exporter.

Visualization methods for game data in Python | Restackio (no date). https://www.restack.io/p/ai-for-texture-generation-in-games-answer-visualization-methods-python-cat-ai.