

distributed-systems-work

Assignments Repo for CS6650 Building Scalable Distributed Systems 23fall@neu

Ayan Mao 2023 Nov

<https://gortonator.github.io/bsds-6650/assignments-2022/Assignment-1>

<https://gortonator.github.io/bsds-6650/assignments-2022/Assignment-2>

Github link: <https://github.com/98may/distributed-systems-work/>

distributed-systems-work

1: Overview

1.1: milestones

1.2: code design

loadTester.java

ApiTask.java

SendGetRequests.java

Utilities.java

Config.java

1.3: key info

AWS EC2

AWS ALB

AWS RDS

server address

2: Database

2.1: choice

2.2: data model

2.3: results

3: single server results

Java Servlet

10

20

30

4: load balanced servers results

simple tomcat test passed

simple alb get test passed

5: tuned results

some tune details

before tune

after tune

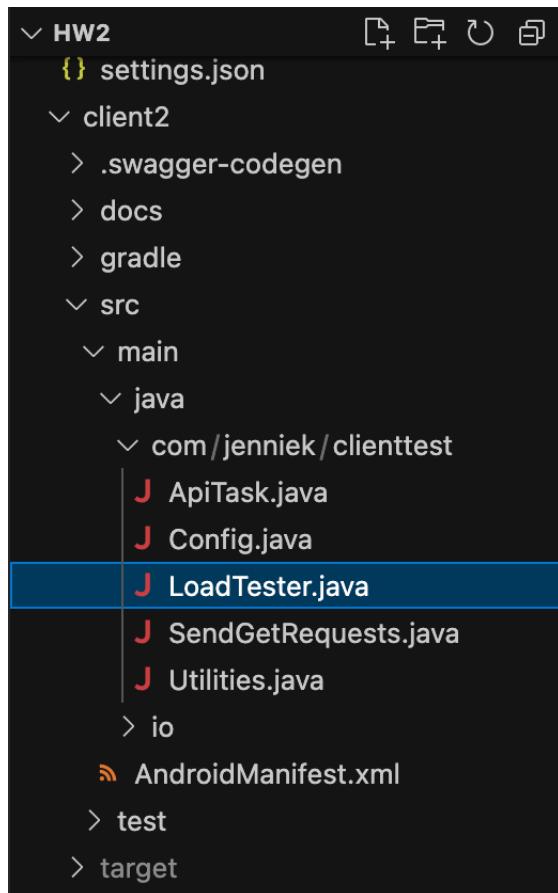
6: Appendix of Command

1: Overview

1.1: milestones

- 1) add analytics
- 2) add persist database - AWS RDS + MySQL
- 3) use multiple servers with load balancers - AWS EC2 + ALB
- 4) fine tune AWS args

1.2: code design



loadTester.java

loadTester is the main java class to handle tests,

`parse` to set up args from `Usage: LoadTester <threadGroupSize> <numThreadGroups> <delay> [java|go]`,

`startUp` to make sure the server on aws works,

and then `mainLoad` send out the major requests.

```
public static void main(String[] args) {  
    LoadTester loadTester = new LoadTester();  
    loadTester.parse(args);  
    loadTester.startUp();  
    loadTester.mainLoad();  
}
```

ApiTask.java

implements `Runnable`

as a single thread unit to send one pair of post and get requests

SendGetRequests.java

implements `Runnable`

as a single thread unit to send one get request for main's `startUp`, like half of the `ApiTask.java`

Utilities.java

mainly the Utility functions for logging and numbers

Config.java

set up the public static final arguments like `INITIAL_THREAD_COUNT`, `CLIENT_LOG_PATH` and `javaServletAddress`

1.3: key info

AWS EC2

Screenshot of the AWS EC2 Instances page showing three running instances: ds_server0, ds_template, and ds_server1.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 address
ds_server0	i-0d3a7ef8476b4c5f0	Running	t4g.nano	2/2 checks passed	View alarms	us-east-1a	ec2-3-80-33-155.compute-1.amazonaws.com	3.80.33.155
ds_template	i-0a9684260210de92	Stopped	t4g.nano	-	View alarms	us-east-1a	-	-
ds_server1	i-050ce65c8ff4d5c40	Running	t4g.nano	2/2 checks passed	View alarms	us-east-1a	ec2-54-221-189-103.compute-1.amazonaws.com	54.221.189.103

Instance: i-0d3a7ef8476b4c5f0 (ds_server0)

Details | Status and alarms | Monitoring | Security | Networking | Storage | Tags

Instance summary

Instance ID	i-0d3a7ef8476b4c5f0 (ds_server0)	Public IPv4 address	3.80.33.155 [open address]
IPv6 address	-	Instance state	Running
Hostname type	IP name: ip-172-31-21-99.ec2.internal	Private IP DNS name (IPv4 only)	ip-172-31-21-99.ec2.internal
Elastic IP	Answer private resource DNS name	Instance type	t4g.nano
Key Pairs	IPv4 (A)	VPC ID	vpc-071fd8305cf646176
Network Interfaces	Auto-assigned IP address	Subnet ID	subnet-0bcb08583efadff683
Load Balancing	IAM Role	Auto Scaling Group name	-
Load Balancers	-	AWS Compute Optimizer finding	Opt-in to AWS Compute Optimizer for recommendations. [Learn more]
Target Groups	IMDSv2 Required	Elastic IP addresses	-
Auto Scaling	Platform	Monitoring	disabled
Auto Scaling Groups	Amazon Linux (Inferred)	AMI ID	ami-0da3fea0ceec717e5

AWS ALB

Screenshot of the AWS Load Balancer details page for the load balancer 'ayan-alb'.

Details

Load balancer type	Status	VPC	IP address type
Application	Active	vpc-071fd8305cf646176	IPv4
Scheme	Hosted zone	Availability Zones	Date created
Internet-facing	Z355XDKOTRQ7X7K	subnet-0a37224dec0192157 us-east-1e (use1-az3)	November 12, 2023, 01:55 (UTC-08:00)
Load balancer ARN	DNS name info	subnet-0bcb08583efadff683 us-east-1a (use1-az4)	
	ayan-alb-357506816.us-east-1.elb.amazonaws.com (A Record)		

Listeners and rules (1) Info

A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.

Protocol:Port	Default action	Rules	ARN	Security policy	Default SSL/TLS certificate	Tags
HTTP:80	Forward to target group	1 rule	arn	Not applicable	Not applicable	0 tags
	• albums (100%)					
	• Group-level stickiness: Off					

AWS RDS

The screenshot shows the AWS RDS console with the following details:

- Summary Tab:**
 - DB identifier: dsdatabase
 - CPU: 3.41%
 - Status: Available
 - Role: Instance
 - Current activity: 35 Connections
 - Engine: MySQL Community
 - Class: db.t3.micro
 - Region & AZ: us-east-1a
- Connectivity & security Tab:**
 - Endpoint & port:**
 - Endpoint: dsdatabase.ccfcharlh91s.us-east-1.rds.amazonaws.com
 - Port: 3306
 - Networking:**
 - Availability Zone: us-east-1a
 - VPC: vpc-071fd8305cf646176
 - Subnet group: rds-ec2-db-subnet-group-1
 - Subnets:
 - subnet-0ef81c139cba77d2a
 - subnet-018af8fc48577b086
 - subnet-011528a42875c039
 - subnet-098be56b7db2189b8
 - subnet-06b4d282f4e58cfb0
 - Network type: IPv4
 - Security:**
 - VPC security groups:
 - rds-ec2-3 (sg-08a9f5eb37de6b420) - Active
 - launch-wizard-2 (sg-05fe376d571e61b0c) - Active
 - Publicly accessible: No
 - Certificate authority: rds-ca-2019
 - Certificate authority date: August 22, 2024, 10:08 (UTC-07:00)
 - DB instance certificate expiration date: August 22, 2024, 10:08 (UTC-07:00)

server address

```

public static final String javaServletAddress =
"http://3.80.33.155:8080/AlbumApp";
public static final String javaServletAddress2 =
"http://54.221.189.103:8080/AlbumApp";
public static final String goServerAddress =
"http://3.80.33.155:8081/IGORTON/AlbumStore/1.0.0";

```

2: Database

2.1: choice

AWS RDS + MySQL 8.0.33

Amazon RDS

Databases

Summary

DB identifier: dsdatabase

CPU: 3.41% Status: Available

Role: Instance Current activity: 35 Connections Engine: MySQL Community

Class: db.t3.micro Region & AZ: us-east-1a

Connectivity & security

Endpoint & port

Endpoint: dsdatabase.ccfcharlh91s.us-east-1.rds.amazonaws.com Port: 3306

Networking

Availability Zone: us-east-1a

VPC: vpc-071fd8305cf646176

Subnet group: rds-ec2-db-subnet-group-1

Subnets:

- subnet-0ef81c139cba77d2a
- subnet-018af8fc48577b086
- subnet-011528c42875c039
- subnet-098be56b7db2189b8
- subnet-06b4d282f4e58cf0

Security

VPC security groups:

- rds-ec2-3 (sg-08a9f5eb37de6b420) Active
- launch-wizard-2 (sg-05fe376d571e61b0c) Active

Publicly accessible: No

Certificate authority: rds-ca-2019

Certificate authority date: August 22, 2024, 10:08 (UTC-07:00)

DB instance certificate expiration date: August 22, 2024, 10:08 (UTC-07:00)

Network type: IPv4

2.2: data model

table: albums

field: album_i d(PK), name, artist, release_year, image

```
Database changed
MySQL [ds_hw2_db]> describe albums;
+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| album_id | varchar(50) | NO | PRI | NULL | 
| name | varchar(100) | NO | | NULL | 
| artist | varchar(100) | NO | | NULL | 
| release_year | varchar(4) | YES | | NULL | 
| image | varchar(255) | YES | | NULL | 
+-----+-----+-----+-----+-----+
```

2.3: results

simple postman test succeeded:

```

[Output from terminal]
| Tables_in_ds_hw2_db |
+-----+
| albums |
+-----+
1 row in set (0.00 sec)

MySQL [ds_hw2_db]> DESCRIBE albums;
+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| album_id | varchar(50) | NO | PRI | NULL |          |
| name | varchar(100) | NO |     | NULL |          |
| artist | varchar(100) | NO |     | NULL |          |
| release_year | varchar(4) | YES |     | NULL |          |
| image_size | bigint(20) | YES |     | NULL |          |
+-----+-----+-----+-----+-----+
5 rows in set (0.005 sec)

MySQL [ds_hw2_db]> SELECT * FROM albums;
Empty set (0.001 sec)

MySQL [ds_hw2_db]> INSERT INTO albums (album_id, name, artist, release_year, image_size)
VALUES ('1', 'Sample Album', 'Sample Artist', '2021', 1024000);
Query OK, 1 row affected (0.005 sec)

MySQL [ds_hw2_db]> SELECT * FROM albums;
+-----+-----+-----+-----+-----+
| album_id | name | artist | release_year | image_size |
+-----+-----+-----+-----+-----+
| 1       | Sample Album | Sample Artist | 2021 | 1024000 |
+-----+-----+-----+-----+-----+
1 row in set (0.001 sec)

MySQL [ds_hw2_db]> SELECT * FROM albums;
+-----+-----+-----+-----+-----+
| album_id | name | artist | release_year | image_size |
+-----+-----+-----+-----+-----+
| 1       | Sample Album | Sample Artist | 2021 | 1024000 |
| 1230    | Album moy Smile | Moy | 8888 | 57040 |
+-----+-----+-----+-----+-----+
2 rows in set (0.002 sec)

yan.h
MySQL [ds_hw2_db]>

```

Postman API Endpoint:

- Method:** POST
- URL:** http://3.80.33.155:8080/AlbumApp/albums
- Body (JSON):**

```

{
  "albumID": "1230",
  "artist": "Moy",
  "title": "Album moy Smile",
  "year": "8888"
}

```

As you can see in the graph, it could have **millions** of records in the albums table.

```
ec2-user@ip-172-31-21-99:~
```

Enter password:
ERROR 1045 (28000): Access denied for user 'admin'@'172.31.21.99' (using password: YES)
[ec2-user@ip-172-31-21-99 ~]\$ mysql -h dsdatabase.ccfcharlh91s.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MySQL connection id is 99577
Server version: 8.0.33 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> show databases;
+-----+
| Database |
+-----+
| ds_hw2_db |
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
5 rows in set (0.082 sec)

MySQL [(none)]> use ds_hw2_db;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

2 Database changed
MySQL [ds_hw2_db]> describe albums;
+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
a. | album_id | varchar(50) | NO | PRI | NULL |
name	varchar(100)	NO		NULL
artist	varchar(100)	NO		NULL
release_year	varchar(4)	YES		NULL
image	varchar(255)	YES		NULL
+-----+-----+-----+-----+-----+
5 rows in set (0.061 sec)

MySQL [ds_hw2_db]> select COUNT(*) from albums;
+-----+
| COUNT(*) |
+-----+
| 2946735 |
+-----+
1 row in set (14.299 sec)

```

+-----+-----+-----+-----+
| image | varchar(255) | YES | NULL |
+-----+-----+-----+-----+
5 rows in set (0.061 sec)

MySQL [ds_hw2_db]> select COUNT(*) from albums;
+-----+
| COUNT(*) |
+-----+
| 2946735 |
+-----+
1 row in set (14.299 sec)

ap MySQL [ds_hw2_db]>
MySQL [ds_hw2_db]> SELECT * FROM albums ORDER BY RAND() LIMIT 20;
+-----+-----+-----+-----+-----+
| album_id | name | artist | release_year | image |
+-----+-----+-----+-----+-----+
| fa4ab152-2609-4f3c-906f-43595c798c40 | rich Album Title | rich Artist | 1998 | nmtb.png |
| 5063a7c8-eb48-4589-ab44-5cee2a139efb | rich Album Title | rich Artist | 1998 | nmtb.png |
| 95225bd6-f64a-479e-9b1f-9c96979df7f5 | rich Album Title | rich Artist | 1998 | nmtb.png |
| 207cd62a-e753-456f-8945-18cc5fd45899 | rich Album Title | rich Artist | 1998 | nmtb.png |
2 | 34cb60cd-60fd-4b3b-a754-bf5c498a1cad | rich Album Title | rich Artist | 1998 | nmtb.png |
| 8608a90d-fa22-4b69-9d6e-4eae35476482 | rich Album Title | rich Artist | 1998 | nmtb.png |
| 71b10444-dfee-4851-9d5f-b647ee8b6080 | rich Album Title | rich Artist | 1998 | nmtb.png |
| e3dc1400-a7c0-4b31-b876-928086987945 | rich Album Title | rich Artist | 1998 | nmtb.png |
li | 96e82672-5d29-42cd-8cb2-43c0b047ee87 | rich Album Title | rich Artist | 1998 | nmtb.png |
a. | f867c506-ac2d-438f-aa6a-6ba1c6a17b86 | rich Album Title | rich Artist | 1998 | nmtb.png |
| daa97ba6-9d36-4a1e-80c9-fef04e18247e | rich Album Title | rich Artist | 1998 | nmtb.png |
| 4de44000-3dd9-4e09-84f1-a40c22088c95 | rich Album Title | rich Artist | 1998 | nmtb.png |
| 299212de-2b97-433a-863d-93a8a0badf47 | rich Album Title | rich Artist | 1998 | nmtb.png |
| 667d4bf5-16e8-4e07-9ff8-b0f85cfdd0d4 | rich Album Title | rich Artist | 1998 | nmtb.png |
| f32b6a7c-f433-483d-9795-52e0c36ddef0 | rich Album Title | rich Artist | 1998 | nmtb.png |
| 5b42eb52-2548-433e-9d4a-edea54589f6c | rich Album Title | rich Artist | 1998 | nmtb.png |
| f74bb4be-6645-4897-8021-c07d9d414dfa | rich Album Title | rich Artist | 1998 | nmtb.png |
| 29abd1c1-a89c-4741-a3b1-c49cc2f04b5f | rich Album Title | rich Artist | 1998 | nmtb.png |
ma | efcc6d09-34dd-45a9-a72f-a676023ab599 | rich Album Title | rich Artist | 1998 | nmtb.png |
| 3ef652db-f0e0-4a18-a6b6-8a442189a664 | rich Album Title | rich Artist | 1998 | nmtb.png |
+-----+-----+-----+-----+
20 rows in set (15.764 sec)

```

3: single server results

Complete test results under /test_results folder

Here are some highlights:

Java Servlet

```
[INFO]
[INFO] --- exec:3.1.0:java (default-cli) @ swagger-java-client ---
Executing: LoadTester 10 10 2 java
===== Start startup Phase =====
2023-11-11 20:17:52
Init Phase - walltime = 10755 mill seconds
Init Phase - throughput =  92
2023-11-11 20:18:02
===== End startup Phase =====

===== Start Main Load Phase =====
2023-11-11 20:18:02
All tasks completed.
totalRequests: 200000, Failure Rate: 0.0
failedRequests: 0
successfulRequests: 200000
2023-11-11 20:18:02 - 2023-11-11 20:22:08
Wall Time: 245 seconds
Throughput: 816 requests/second
----- GET request statistics -----
Mean: 102.82146 ms
Median: 92.0 ms
P99: 222.0 ms
Min: 79 ms
Max: 716 ms
----- POST request statistics -----
Mean: 122.55521 ms
Median: 110.0 ms
P99: 267.0 ms
Min: 85 ms
Max: 963 ms
2023-11-11 20:22:08
===== End Main Load Phase =====
[WARNING] thread Thread[OkHttp ConnectionPool, 5, com.jennick.clienttest]
```

```
[INFO] --- exec:3.1.0:java (default-cli) @ swagger-java-cl
Executing: LoadTester 10 20 2 java
===== Start startup Phase =====
2023-11-11 20:24:21
Init Phase - walltime = 11190 mill seconds
Init Phase - throughput = 89
2023-11-11 20:24:32
===== End startup Phase =====

===== Start Main Load Phase =====
2023-11-11 20:24:32
All tasks completed.
totalRequests: 400000, Failure Rate: 0.0
failedRequests: 0
successfulRequests: 400000
2023-11-11 20:24:32 – 2023-11-11 20:29:40
Wall Time: 307 seconds
Throughput: 1302 requests/second
----- GET request statistics -----
Mean: 121.59046933357332 ms
Median: 103.0 ms
P99: 408.0 ms
Min: 80 ms
Max: 2205 ms
----- POST request statistics -----
Mean: 148.480565 ms
Median: 126.0 ms
P99: 459.0 ms
Min: 86 ms
Max: 2790 ms
2023-11-11 20:29:40
===== End Main Load Phase =====
```

```
[INFO] ----- [ JAR ] -----
[INFO]
[INFO] --- exec:3.1.0:java (default-cli) @ swagger-java-client ---
Executing: LoadTester 10 30 2 java
===== Start startup Phase =====
2023-11-11 20:31:06
Init Phase - waltime = 10777 mill seconds
Init Phase - throughput =  92
2023-11-11 20:31:17
===== End startup Phase =====

===== Start Main Load Phase =====
2023-11-11 20:31:17
All tasks completed.
totalRequests: 600000, Failure Rate: 0.0
failedRequests: 0
successfulRequests: 600000
2023-11-11 20:31:17 - 2023-11-11 20:37:50
Wall Time: 393 seconds
Throughput: 1526 requests/second
----- GET request statistics -----
Mean: 152.32777301920058 ms
Median: 125.0 ms
P99: 528.0 ms
Min: 79 ms
Max: 2316 ms
----- POST request statistics -----
Mean: 183.30925 ms
Median: 153.0 ms
P99: 604.0 ms
Min: 86 ms
Max: 2921 ms
2023-11-11 20:37:51
===== End Main Load Phase =====
```

4: load balanced servers results

simple tomcat test passed

If you're seeing this, you've successfully installed Tomcat. Congratulations!

Recommended Reading:

- [Security Considerations How-To](#)
- [Manager Application How-To](#)
- [Clustering/Session Replication How-To](#)

Developer Quick Start

Tomcat Setup
First Web Application

Realms & AAA
JDBC DataSources

Examples

Servlet Specifications
Tomcat Versions

Managing Tomcat

For security, access to the [manager_webapp](#) is restricted. Users are defined in: [\\$CATALINA_HOME/conf/tomcat-users.xml](#)

In Tomcat 9.0 access to the manager application is split between different users. [Read more...](#)

Release Notes
Changelog
Migration Guide
Security Notices

Documentation

[Tomcat 9.0 Documentation](#)
[Tomcat 9.0 Configuration](#)
[Tomcat Wiki](#)

Find additional important configuration information in:
[\\$CATALINA_HOME/RUNNING.txt](#)

Developers may be interested in:
[Tomcat 9.0 Bug Database](#)
[Tomcat 9.0 JavaDocs](#)
[Tomcat 9.0 Git Repository at GitHub](#)

Getting Help

[FAQ and Mailing Lists](#)

The following mailing lists are available:

- tomcat-announce**
Important announcements, releases, security vulnerability notifications. (Low volume).
- tomcat-users**
User support and discussion
- taglibs-user**
User support and discussion for [Apache Taglibs](#)
- tomcat-dev**
Development mailing list, including commit messages

simple alb get test passed

```
{"albumID": "1", "title": "Sample Album", "artist": "Sample Artist", "year": "2021", "image": null}
```

5: tuned results

After tune rds, throughput increased from 1117 to 1595, which is 42.8% improvements, great!

some tune details

The screenshot shows the AWS RDS Modify DB instance page for a database named 'dsdatabase'. The top navigation bar includes the AWS logo, Services, a search bar, and a keyboard shortcut [Option+S]. Below the navigation, there are links for S3, RDS (which is selected), and EC2.

The main title is 'Modify DB instance: dsdatabase'. A 'Summary of modifications' section displays the following table:

Attribute	Current value	New value
Multi-AZ deployment	No	Yes
Storage type	General Purpose SSD (gp2)	General Purpose SSD (gp3)
Allocated storage	20 GiB	200 GiB

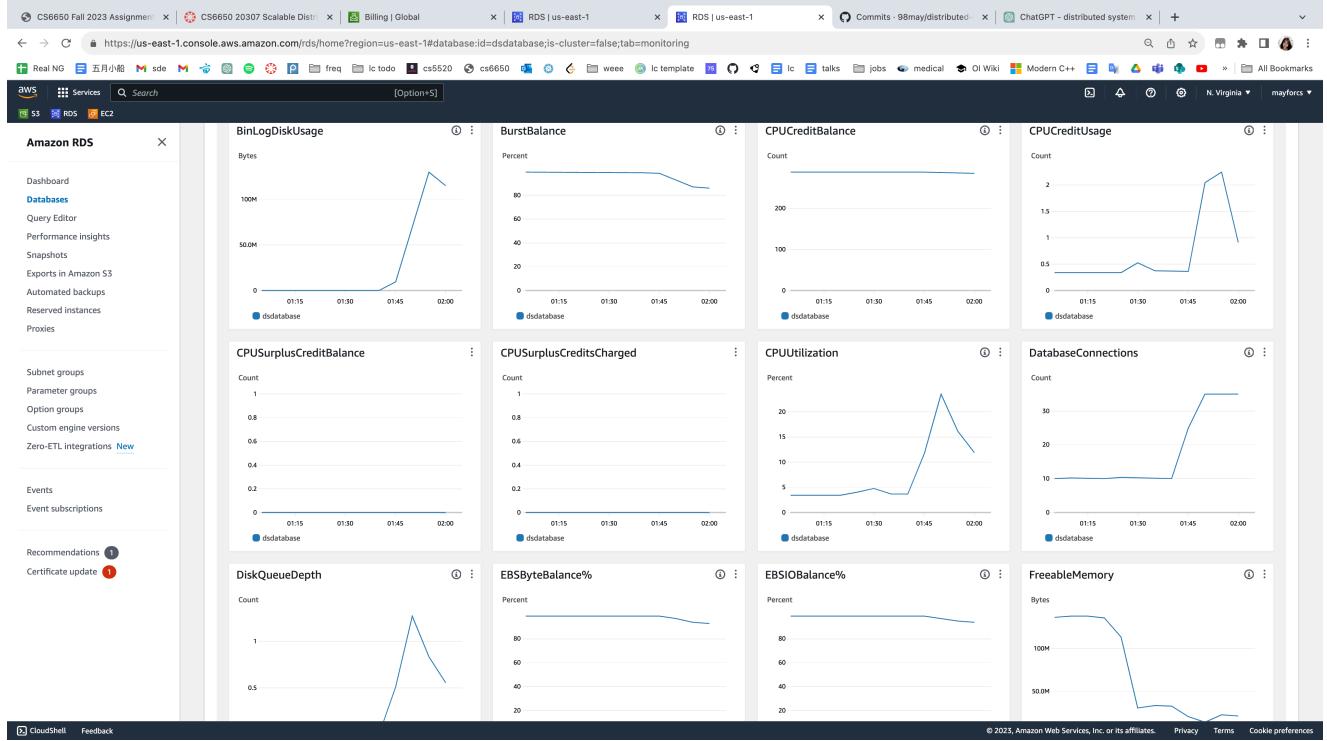
The 'Schedule modifications' section contains two options:

- Apply during the next scheduled maintenance window
Current maintenance window: November 27, 2023 23:09 - 23:39 UTC-8
- Apply immediately
The modifications in this request and any pending modifications will be asynchronously applied as soon as possible, regardless of the maintenance window setting for this database instance.

A yellow warning box titled 'Potential performance impact when converting to Multi-AZ' states: 'Your DB instance can experience a significant performance impact during and after converting to a Multi-AZ deployment. The impact is greater on DB instances with large amounts of storage and write-intensive workloads. We don't recommend this conversion on a production DB instance.'

At the bottom right are three buttons: 'Cancel', 'Back', and a prominent orange 'Modify DB instance' button.

before tune



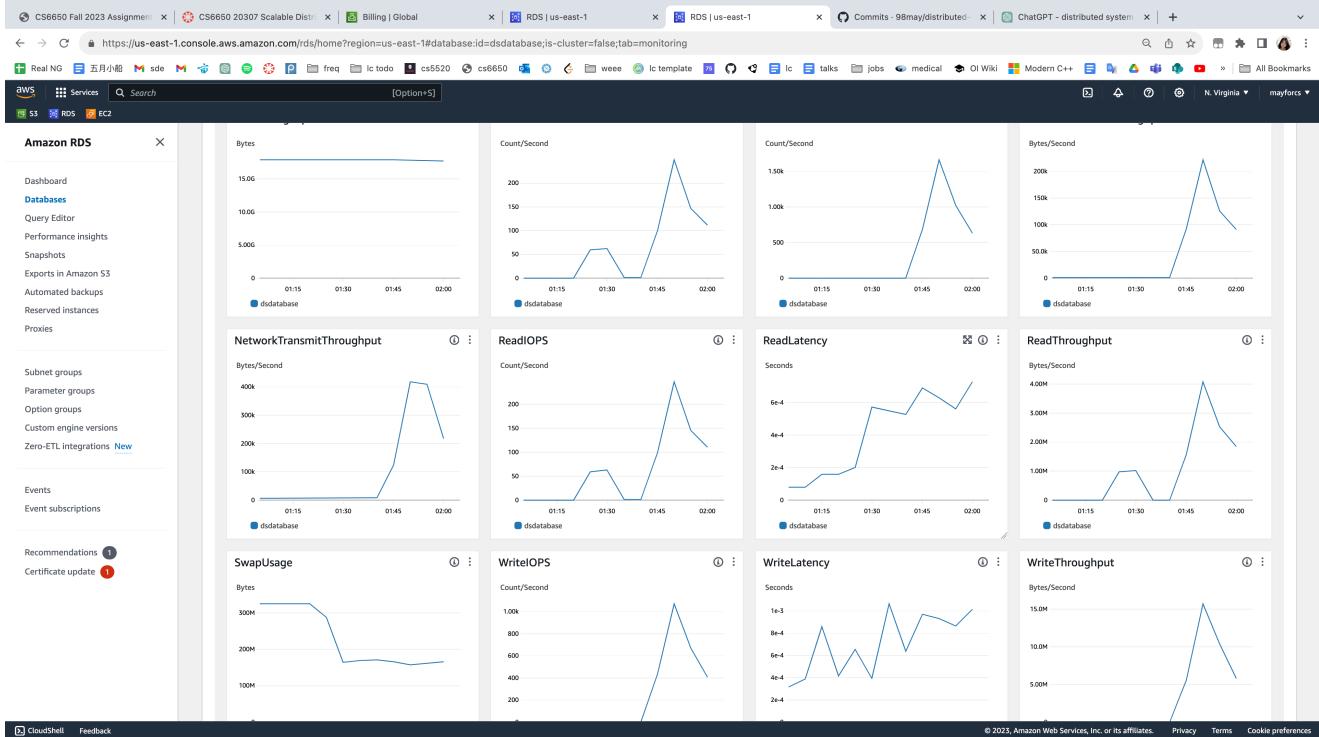
This screenshot shows the 'Connectivity & security' tab of the AWS RDS Database details page for the database 'dsdatabase'.

Summary

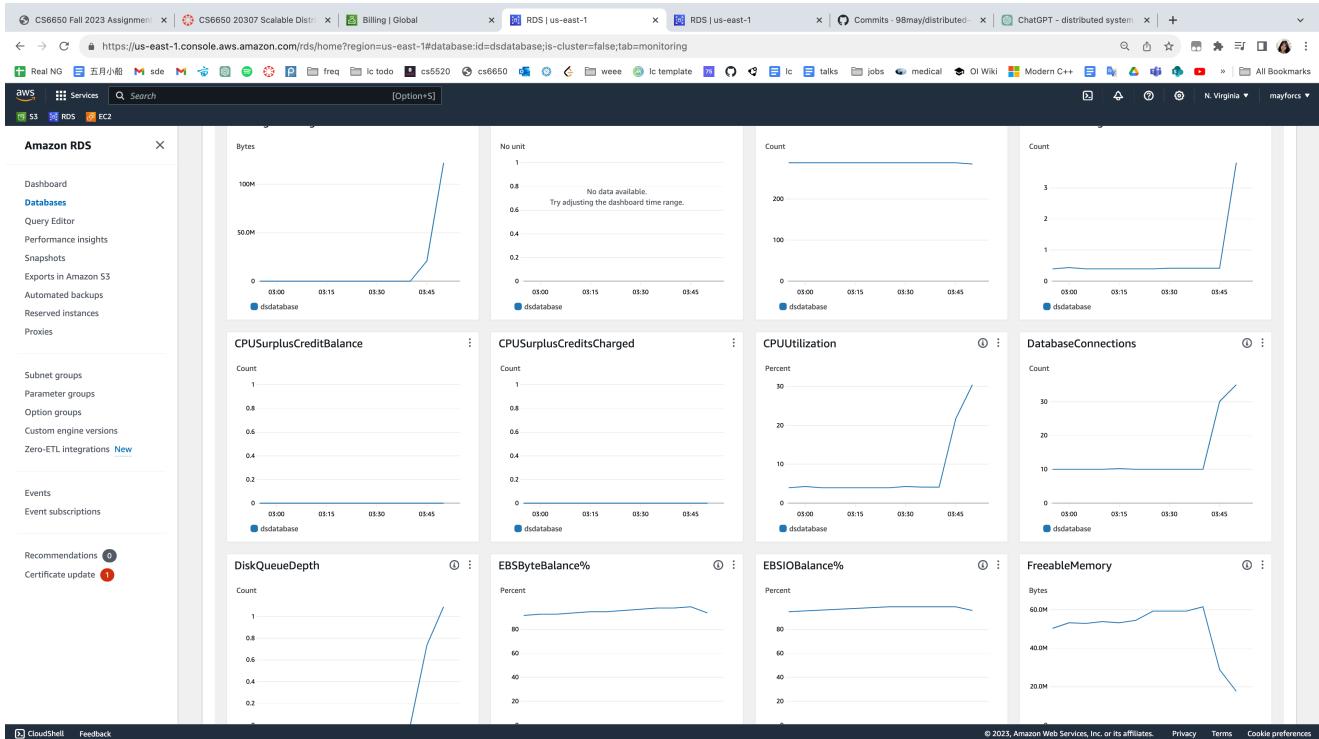
DB identifier	CPU	Status	Class
dsdatabase	3.41%	Available	db.t3.micro
Role	Current activity	Engine	Region & AZ
Instance	35 Connections	MySQL Community	us-east-1a

Connectivity & security

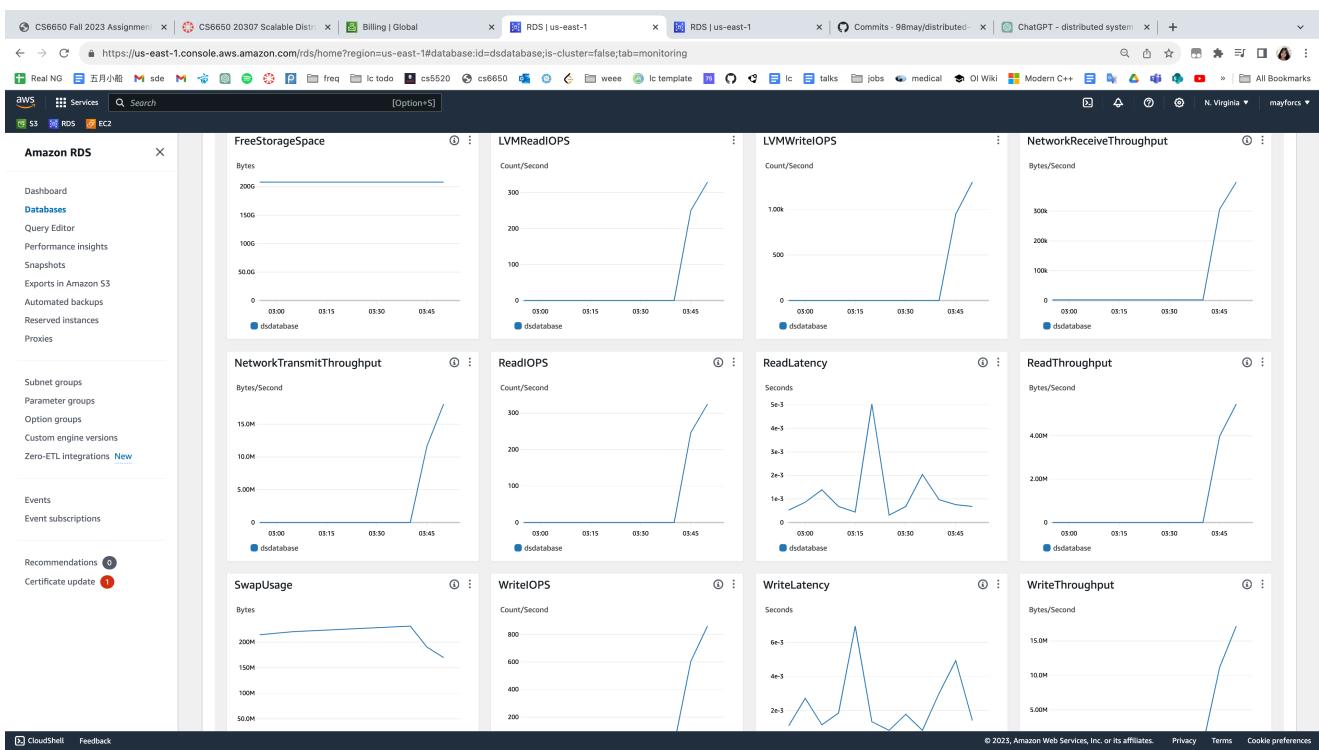
Endpoint	Networking	Security
dsdatabase.ccfc1hrh91s.us-east-1.rds.amazonaws.com	Availability Zone: us-east-1a	VPC security groups: rds-ec2-3 (sg-08a9f3eb37de6b420) (Active), launch-wizard-2 (sg-05fe376d571e61b0c) (Active)
Port: 3306	VPC: vpc-071fd8305cf646176	Publicly accessible: No
	Subnet group: rds-ec2-db-subnet-group-1	Certificate authority: rds-ca-2019
	Subnets: subnet-0ef81c139cba77d2a, subnet-0184fffc48577b086, subnet-011528a42875c039, subnet-098be56b7db2189b8, subnet-06bdad282f4e58cfb0	Certificate authority date: August 22, 2024, 10:08 (UTC-07:00)
	Network type: IPv4	DB instance certificate expiration date: August 22, 2024, 10:08 (UTC-07:00)



after tune



Screenshot of the AWS RDS console showing the database configuration for 'dsdatabase'. The 'Summary' tab is selected, displaying basic information like DB identifier, CPU usage (7.26%), Status (Available), and Engine (MySQL Community). The 'Connectivity & security' tab is also visible, showing endpoint details, networking configurations, and security groups.



6: Appendix of Command

Please login as the user "ec2-user" rather than the user "root".

local

```

// ssh to aws ec2
/Users/may/Desktop/neu/cs6650_distributed/shortcuts/ssh_ec2.sh

// chmod 400 ec2_2.pem
// ssh -i ec2_2.pem ec2-user@ec2-3-80-33-155.compute-1.amazonaws.com
ssh -i ec2_2.pem ec2-user@ec2-54-221-189-103.compute-1.amazonaws.com


// mysql to aws rdb
mysql -h dsdatabase.ccfcharlh91s.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
    // pwd: 01234567

    /* java servlet */
// local build java servlet in Maven project to generate the AlbumApp.war file
mvn clean install
// scp java servlet - AlbumApp.war to aws ec2
scp -i /Users/may/Downloads/ec2_2.pem
/Users/may/Desktop/neu/cs6650_distributed/distributed-systems-
work/hw2/java_servlet/AlbumApp/target/AlbumApp.war ec2-user@ec2-3-80-33-
155.compute-1.amazonaws.com:~/tmp


/* Go server */
// local run go server(macOS + m1 chip) for local tests
GOOS=darwin GOARCH=arm64 go run ./main.go
// local build go server for aws ec2(linux + arm64)
GOOS=linux GOARCH=arm64 go build -o server main.go
// scp go server - server to ec2 ~/emp to start
scp -i /Users/may/Downloads/ec2_2.pem
/Users/may/Desktop/neu/cs6650_distributed/distributed-systems-
work/hw1/go_server/go-server-generated/server ec2-user@ec2-3-80-33-
155.compute-1.amazonaws.com:~/tmp


/* client - LoadTester */
`Usage: LoadTester <threadGroupSize> <numThreadGroups> <delay> [java|go]` 
// e.g.
mvn clean install && mvn exec:java -
Dexec.mainClass="com.jenniek.clienttest.LoadTester" -Dexec.args="10 30 2 java"

mvn exec:java -Dexec.mainClass="com.jenniek.clienttest.LoadTester" -Dexec.args="10
30 2 java"

// plot throughput
conda activate base
/Users/may/anaconda3/bin/python

```

```
/Users/may/Desktop/neu/cs6650_distributed/distributed-systems-
work/hw2/test_results/plot.py
```

```
aws ec2
```

```
// java servlet
sudo mv ~/tmp/AlbumApp.war /usr/share/tomcat/webapps && sudo systemctl restart
tomcat && systemctl status tomcat

sudo mv ~/tmp/AlbumApp.war /usr/share/tomcat/webapps

sudo systemctl restart tomcat
sudo systemctl stop tomcat
systemctl status tomcat

sudo vim /usr/share/tomcat/logs/catalina.out
sudo tail /usr/share/tomcat/logs/catalina.out -n 200

// see log to debug java servlet
sudo bash -c 'echo > /usr/share/tomcat/logs/catalina.out'
sudo grep "album" /usr/share/tomcat/logs/catalina.out

// go server
~/tmp/server

// mysql to aws rdb
mysql -h dsdatabase.ccfcharlh91s.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
    // 01234567
show databases;
use ds_hw2_db;
describe albums;
select COUNT(*) from albums;
SELECT * FROM albums ORDER BY RAND() LIMIT 20;
```