Q A T A R U N I V E R S I T Y
College of Engineering
Dept. of Computer Science &

Operating Systems Lab (CMPS
Spring 2025 Semester

Project Phase # 2

Socket Programming: Priority-Based Task Execution System with Script Dispatcher

Due to 3rd May 2025

يعد الغش مخالفة أكاديمية وفقا للوائح والقوانين المعمول بها في جامعة قطر، وقد تصل عقوبة هذه المخالفة في بعض الحالات إلى الفصل النهائي من الجامعة وعلى الطلاب تجنب القيام أو المشاركة في أي عمل يخالف ميثاق النزاهة الأكاديمية وإجراءات الاختبارات المعمول بها في جامعة قطر

Cheating is an academic violation according to Qatar University rules and regulations, and in some cases, it may result in final dismissal from the university. Students should not under any circumstances commit or participate in any cheating attempt or any act that violates student code of conduct.

It is the right of the instructor to test the student's undersetting of the project in any way during the demo and discussion session. So, a project that is 100% working might be may be graded (-100%) due to student not being able to explain a functionality they have

Project Objectives:

Build a multithreaded client-server Java application that allows clients to request prioritized execution of Linux shell scripts on a server. The system ensures safe task distribution, synchronization, task history logging, and allows clients to query and manage their submitted tasks , **building on your Phase 1 work.**

Environment Setup:

- **Virtual Machines:** Use **Ubuntu Server 22.04 LTS** for three VMs:
  - **VM1 (Server):** (4GB RAM, 4 CPUs) Hosts user accounts, shared directories, and critical services (SSH, MySQL).
  - **VM2 (Client - Development Team):** (2GB RAM, 2 CPU each) Used for development tasks and directory monitoring.
  - **VM3 (Client - Operations Team):** (2GB RAM, 2 CPU each) Used for system monitoring and reporting.

-

Note: [If it is not possible to install it on the same machine, feel free to install it on different physical machines].

Follow these detailed specifications:

Server.java

1. **Listening Port**: The server listens on port **2500** continuously for client connections.
2. **Connection Check**: On any new client connection, the server executes **Network.sh** to validate connectivity with both clients (Client1VM and Client2VM).
3. **Task Requests with Priority**:
   - Clients can request execution of the following scripts, each associated with a service number and **priority levels**:

Script Name	**Service Number**
user_setup.sh	2001
dir_perms.sh	2002
system_monitor.sh	2003
file_audit.sh	2004
MySQL_login_<USER_NAME>.sh	2005

4. **Request Format**:

```
REQUEST_TASK;ServiceNumber;ClientName;Priority
Example: REQUEST_TASK;2003;Client1;1
```

Priority: 1 = High, 2 = Medium, 3 = Low

5. **Task Queue and Synchronization**:
   - Tasks are stored in a PriorityQueue sorted by priority and request timestamp.
   - The server handles execution using threads, but synchronizes to allow only one instance of the same script at a time.

6. **Rate Limiting**:
   - Each client can submit a new task every **5 minutes**.
   - If violated, the task is rejected and logged.

7. **Communication Protocol Format**

   Server responses to client commands must follow this format:

```
STATUS;TIMESTAMP;MESSAGE
```

Possible STATUS values:

QUEUED:	Task successfully added to the queue.
EXECUTING:	Task has started execution.
COMPLETED:	Task finished successfully.
REJECTED:	Task was not accepted (rate limit, invalid format).

ERROR:	An execution error occurred.

Examples
QUEUED → Task added to queue

```
STATUS;2025-04-05 12:01:12;Task queued with ID 105
```

ERROR → Script failed or crashed

```
STATUS;2025-04-05 12:09:00;ERROR: TaskID 107 encountered execution failure.
```

8. **Tracking and Display**:
Clients may also send the following commands:

Command	**Description**
QUEUE_STATUS	View the current queue of pending tasks.
CANCEL_TASK;TaskID	Cancel a pending task submitted by the client.
TASK_HISTORY	Retrieve logs of completed, cancelled, or failed tasks.

Examples

a) QUEUE_STATUS
Shows all currently pending tasks in the queue.

Client request:

```
QUEUE_STATUS
```

Server response:

```
STATUS;2025-04-05 12:30:01;Pending Tasks:
1. TaskID=101, Script=system_monitor.sh, Priority=1, Client=Client1
2. TaskID=102, Script=dir_perms.sh, Priority=2, Client=Client2
```

b) CANCEL_TASK;TaskID
Cancels a pending task by ID (if not already executing).
Client request:

```
CANCEL_TASK;102
```

Server responses:
- If successful:

```
STATUS;2025-04-05 12:32:44;TaskID 102 cancelled successfully.
```

- If task already executing or not found:

```
STATUS;2025-04-05 12:32:44;REJECTED: Task not found or already running.
```

c) TASK_HISTORY
Returns a log of all completed, cancelled, or failed tasks.

Client request:

```
TASK_HISTORY
```

Server response:

```
STATUS;2025-04-05 12:34:22;History:
1. TaskID=95, Script=file_audit.sh, Client=Client2, Status=COMPLETED
2. TaskID=96, Script=user_setup.sh, Client=Client1, Status=CANCELLED
```

Client1.java

- Runs the following scripts locally:
  - ssh_config.sh, fix_perms.sh, login_audit.sh
- Sends requests for server-side script execution every 5 minutes.
- May also send QUEUE_STATUS, CANCEL_TASK, or TASK_HISTORY.

Client2.java

- Runs the following scripts locally:
  - resource_report.sh, quota_check.sh
- Sends requests for server-side script execution every 5 minutes.
- May also send QUEUE_STATUS, CANCEL_TASK, or TASK_HISTORY.

Instructions & Deliverables: Please read carefully

1. **Submission Requirements:**
    - **Word Document:**
        - Cover page with group member details (Name, ID).
        - Task distribution table (tasks assigned per member + contribution %).
        - Screenshots for every task.
    - **Java classess and Scripts:** All scripts must be error-free and include inline comments explaining logic.
    - **Zip File:** Named as Student1_Student2_Student3_Student4.zip.
2. **Anti-Plagiarism Measures:**
    - **Demo Session:** Randomly assigned tasks must be executed live.
    - **Script Defense:** Students must explain every line of code during the demo.
3. **Penalties:**

- Copying and/or plagiarism (-100%) which includes:
    - **Inappropriate interaction with any other student, outside agency, website, or software that generates assessment responses.**
- Shell Script should be error free (Errors) (-25%).
- In case of late submission, (-10%) for each day of delay (Max 3 days delay).
- A group of three to four students can work on the project.
- Team members are required to meet regularly for discussion and workload distribution.
- It is the right of the instructor to use any way of testing the student in the discussion and demo session, and according to that in some cases (100%) graded project may be down to (-100%) graded project.
- Discussion & Demo 50%. + Student Work 50%.
- One demo after the submission of the second project phase.