INFORMATICS INSTITUTE OF TECHNOLOGY

UNIVERSITY OF WESTMINSTER

UNIVERSITY OF WESTMINSTER

Informatics Institute of Technology

Object Oriented Programming

5COSC019C.1

Coursework Report

Student Name: J.G.S.Yasith Subodha Gunawardhana

Student UoW ID: w2082270

Student IIT ID: 20232817

Tutorial Group: CS-G25

INFORMATICS INSTITUTE OF TECHNOLOGY

UNIVERSITY OF WESTMINSTER

Test Cases for Command Line Interface

Test Case	Input	Expected Output	Actual Output	Pass/Fail
1. System Initialization	Run the program and configure the system with valid inputs (e.g., max ticket capacity: 100, vendors: 2, customers: 3).	Configuration saved successfully, Ticket Pool created with max capacity, Vendors and customers initialized	Configuration saved successfully, Ticket Pool created with max capacity 100, Vendors and customers initialized correctly	Pass
2. Start Simulation	Start the simulation after configuration.	Vendor and Customer threads start, Tickets are released and bought as per configuration	Simulation started successfully. Tickets are being released and bought.	Pass
3. Stop Simulation	Stop the simulation mid-execution.	Vendor and Customer threads are interrupted, No further tickets are released or bought	Threads interrupted. Simulation stopped.	Pass
4. Vendor Ticket Release	Vendors release tickets at specified rate (e.g., 5 tickets per second).	Tickets are added to the pool, Ticket Pool capacity updates correctly	Tickets released successfully. Ticket Pool updated.	Pass
5. Customer Ticket Purchase	Customers buy tickets at specified rate (e.g., 2 tickets per second).	Tickets removed from the pool, Customer ticket count increases	Tickets bought successfully. Customer ticket count updated.	Pass
6. Exceed Maximum Capacity	Attempt to add tickets beyond the maximum capacity.	Pool blocks additional tickets, No overflow occurs	Ticket release blocked beyond capacity.	Pass

7. Exceed Customer Purchase Limit	Attempt to buy tickets exceeding available quantity.	Purchase blocked when tickets run out	Customers cannot buy tickets when the pool is empty.	Pass
8. System Status	Request system status during simulation.	Current ticket count, total released, and sold statistics are displayed	System status displayed correctly.	Pass
9. Log File Generation	Perform operations and check `logs.txt`.	Log file is created, Entries match operations performed	Log file generated with correct entries.	Pass
10. Thread Termination	Stop threads during simulation.	All threads terminate gracefully	Threads terminated successfully.	Pass

Spring Boot backend test cases.

Test Case	API Endpoint	Input	Expected Output	Actual Output / Pass/Fail
System Configuration	/api/simulation/configure	Valid payload: {maxTicketPool: 100, totalVendors: 2, releasePerVendor: 5, vendorReleaseRate: 2, ...}	200 OK with response 'Configuration saved successfully.'	pass
Invalid Configuration	/api/simulation/configure	Invalid payload: {maxTicketPool: -1, totalVendors: 0, ...}	400 Bad Request with error details.	pass
Start Simulation	/api/simulation/start	No payload required.	200 OK with response 'Simulation Started Successfully.'	pass
Start Without Config	/api/simulation/start	No payload, no prior configuration.	400 Bad Request with error 'System must be configured before starting simulation.'	pass
Stop Simulation	/api/simulation/stop	No payload required.	200 OK with response 'Simulation stopped successfully.'	pass
Stop Without Start	/api/simulation/stop	No payload, simulation not started.	400 Bad Request with error 'No simulation is currently running.'	pass
Get System Status	/api/simulation/status	No payload required.	200 OK with payload {maximumPoolSi	pass

			ze: 100, currentPoolSize: 50, availableSpace: 50, ...}	
Fetch Logs	/api/simulation/logs	No payload required.	200 OK with list of log entries {id, threadType, action, timestamp, message}.	pass
Exceed Ticket Pool	/api/simulation/start	Configure with a low pool size, start simulation, and add tickets beyond the capacity.	Simulation blocks ticket addition when the pool is full; no errors occur.	pass
Exceed Ticket Purchase	/api/simulation/status followed by customer actions	Current pool size reflects zero tickets; purchase attempts return appropriate error messages.	Customers cannot purchase tickets beyond availability.	pass
Invalid Origin Access	Any API endpoint	Access any API endpoint from an unallowed origin.	403 Forbidden with CORS policy error.	pass
Vendor Logs	/api/simulation/logs	Verify vendor actions such as 'STARTED,' 'RELEASED,' and 'COMPLETED' in log entries.	Logs reflect vendor thread operations.	pass
Customer Logs	/api/simulation/logs	Verify customer actions like ticket purchases in log entries.	Logs reflect customer thread operations.	pass
Simultaneous Start/Stop	/api/simulation/start followed quickly by /api/simulation/stop	Ensure no race conditions; logs and status confirm smooth operation.	No issues observed in simulation lifecycle.	pass
Error Handling	Any endpoint with forced error (e.g., DB unavailable)	Force an error and observe API response.	500 Internal Server Error with detailed error	pass

			response and proper logging.	

Class diagrams and sequence diagrams

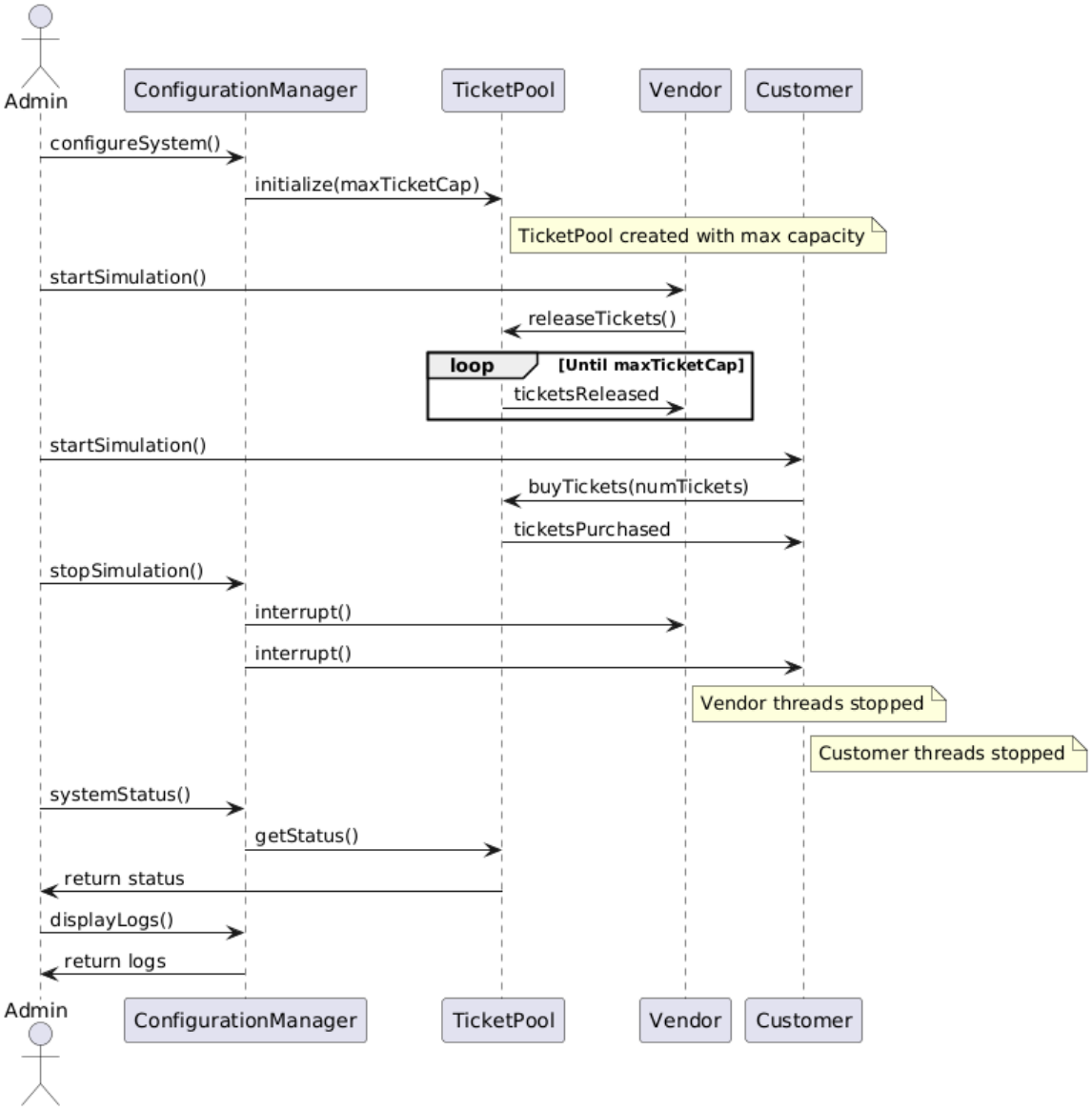

Figure 1: Sequence diagram for cli configuration

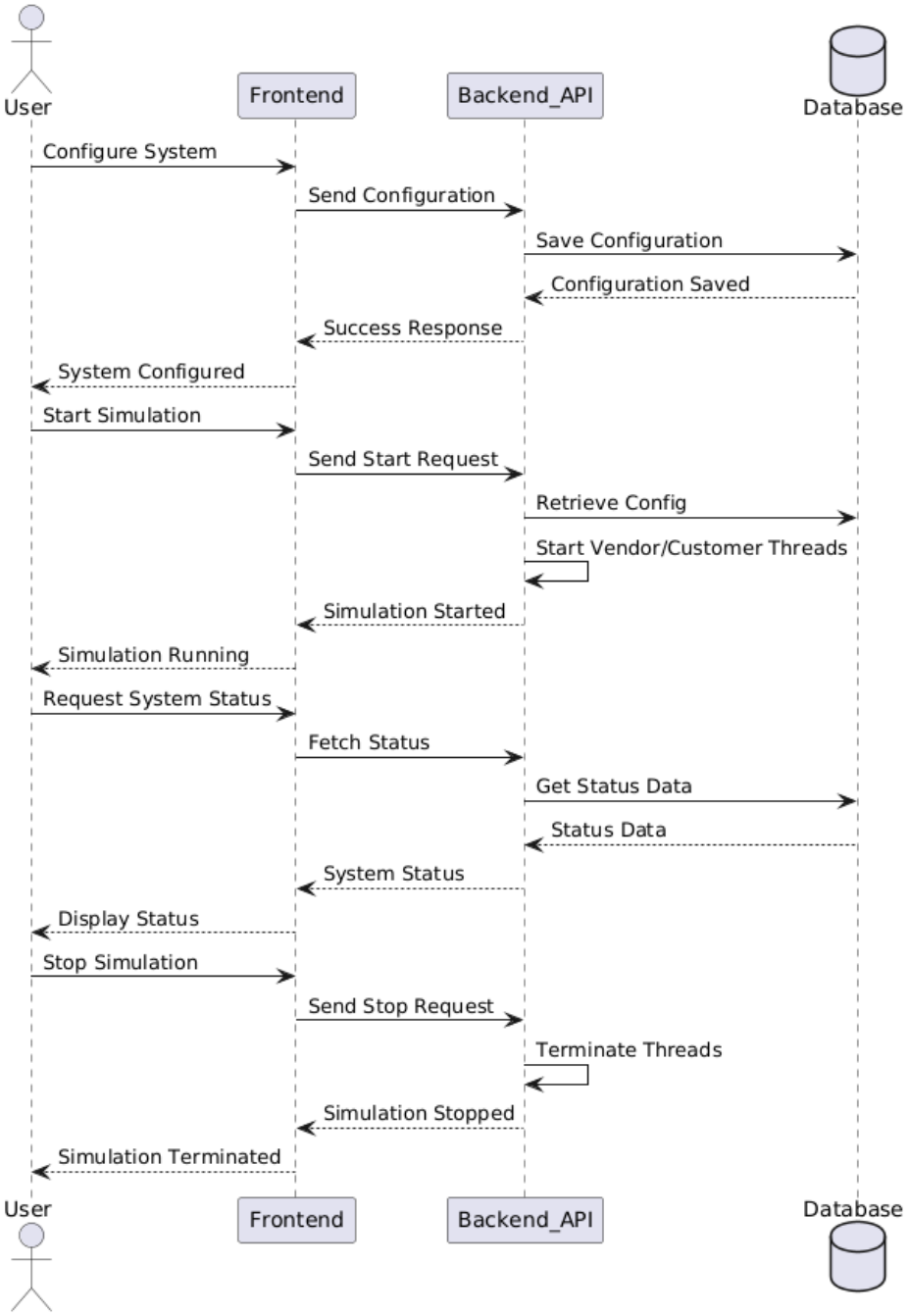

Figure2: Sequence Diagram for backend and frontend.

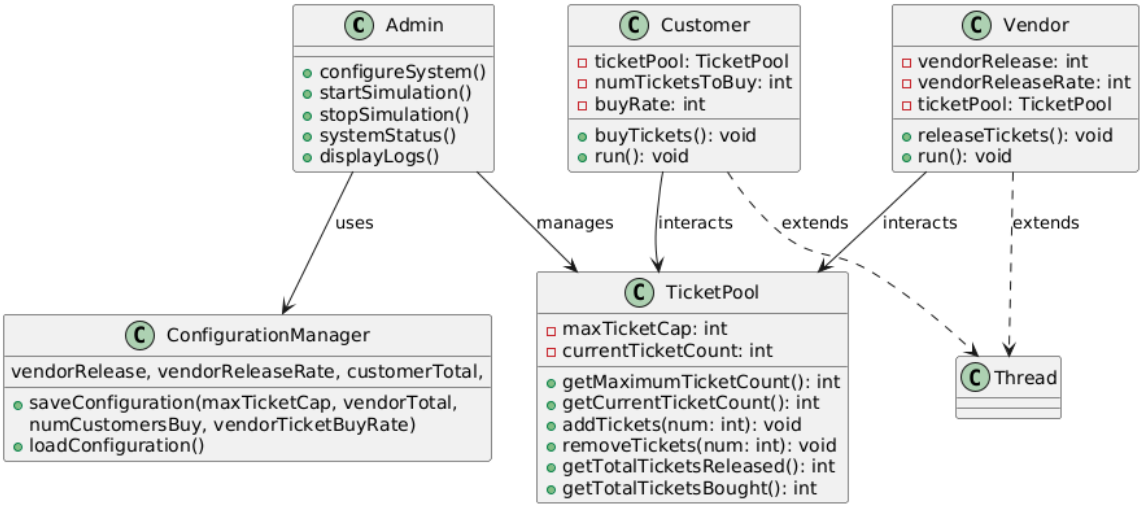

Figure 3: Class Diagram for configuration cli