

Informatics Institute of Technology
in collaboration with
The University of Westminster

Object Oriented Programming Principles

5COSC019C.1

Module Leader: Mr. Poravi Guganathan

Ticket Booking Simulator

Name	B.M.V.L. Buthgamuwa
IIT No	20231974
UoW ID	W2053482
Tutorial Group	Group 7

Table of Contents

Table of Contents	1
Class Diagram	2
Sequence Diagram	3
Test Case Report.....	4

Table of Figures

Table 1: CLI Test Case	4
Table 2:Simulation Execution Test Cases	5
Table 3: User Interface Test Cases	6

```

classDiagram
    class Vendor {
        - id: int
        - delayTime: long
        + run(): void
    }
    class SimulatorManager {
        - db: DBManager
        - vendor_customer_count: int
        - runSimulationEnd: boolean
        - customerBookings: HashMap<Integer, Integer>
        - vendorTickets: HashMap<Integer, Integer>
        - runningSimulator: boolean
        - runSimulationEndLock: ReentrantLock
        - customerBookingLock: ReentrantLock
        - vendorTicketLock: ReentrantLock
        - runningSimulationLock: ReentrantLock
        + runSimulation(): void
        + initialise(): void
        + stopSimulation(): void
        + startRunningSimulator(): void
        + isRunningSimulator(): boolean
        + initialiseCustomerBookings(): void
        + initialiseVendorTickets(): void
        + recordCustomerBooking(int): void
        + recordVendorTicket(int): void
        + getCustomerBookings(): HashMap
        + getVendorTickets(): HashMap
        + isRunSimulationEnd(): boolean
        + asyncIsRunSimulationEnd(): boolean
        + setRunSimulationEnd(boolean): void
    }
    class Customer {
        - id: int
        - delayTime: long
        + run(): void
    }
    class Configuration {
        - totalTickets: int
        - ticketReleaseRate: int
        - customerTicketRetrievalRate: int
        - maxTicketCapacity: int
        + Configuration(totalTickets: int, ticketReleaseRate: int, customerRetrievalRate: int, maxTicketCapacity: int)
        + isValid(): boolean
        + toJson(): String
        + writeConfigToFile(): void
        + getTotalTickets(): int
        + getTicketReleaseRate(): int
        + getCustomerRetrievalRate(): int
        + getMaxTicketCapacity(): int
    }
    class TicketBookingApplicationCLI {
        - input: Scanner
        + main(args: String[]): void
        + getIntInput(promptText: String): int
    }
    class DBManager {
        - url: String
        - logger: Logger
        + DBManager()
        + DBManager(url: String)
        + writeDatabase(SQLCommand: String, parameters: List<Object>): String
        + readDatabase(SQLCommand: String, parameters: List<Object>): List<Map<String, Object>>
        + setup(): void
    }
    class LogManager {
        - logs: List<String>
        - filename: String
        + getLogs(): List<String>
        + clearLogs(): void
        + log(text: String): void
        + writeToFile(): void
        + clearLogFile(): void
    }
    class AppController {
        + start(config: Configuration): ResponseEntity<String>
        + get_logs(): ResponseEntity<List<String>>
        + getTickets(): ResponseEntity<List<Ticket>>
        + getCustomerBookings(): ResponseEntity<Map<Integer, Integer>>
        + getVendorTickets(): ResponseEntity<Map<Integer, Integer>>
        + stopSimulation(): ResponseEntity<String>
        + isRunning(): ResponseEntity<String>
        + getStats(): ResponseEntity<String>
    }
    class TicketBookingSimApp {
        + main(args: String[]): void
    }
    class TicketPool {
        - lock: ReentrantLock
        - ticketBookedCountLock: ReentrantLock
        - ticketCountLock: ReentrantLock
        - ticketsListSizeLock: ReentrantLock
        - totalTickets: int
        - ticketReleaseRate: int
        - customerRetrievalRate: int
        - maxTicketCapacity: int
        - ticketBookedCount: int
        - ticketCount: int
        - ticketsListSize: int
        - ticketsList: List<Ticket>
        - db: DBManager
        + createTicket(ticket: Ticket): void
        + bookTicket(customerId: int): void
        + getTicketsListSize(): int
        + setTicketsListSize(size: int): void
        + changeTicketsListSize(value: int): void
        + getStatus(): String
        + resetTicketPool(): void
        + getTicketsList(): List<Ticket>
        + getAsyncTicketsList(): List<Ticket>
        + clearTicketsList(): void
        + setTotalTickets(tickets: int): void
        + getTicketReleaseRate(): int
        + setTicketReleaseRate(rate: int): void
        + getCustomerRetrievalRate(): int
        + setCustomerRetrievalRate(rate: int): void
        + getMaxTicketCapacity(): int
        + setMaxTicketCapacity(capacity: int): void
        + getTotalTickets(): int
        + getTicketBookedCount(): int
        + setTicketBookedCount(count: int): void
        + getTicketCount(): int
        + setTicketCount(count: int): void
    }
    class Ticket {
        - vendorId: int
        - customerId: int
        - ticketId: int
        - isAvailable: boolean
        + Ticket(vendorId: int)
        + getVendorId(): int
        + setVendorId(vendorId: int): void
        + getCustomerId(): int
        + setCustomerId(customerId: int): void
        + isAvailable(): boolean
        + setAvailable(available: boolean): void
        + getTicketId(): int
        + setTicketId(ticketId: int): void
    }

    Vendor "1..*" -- "1..1" SimulatorManager
    Customer "1..*" -- "1..1" SimulatorManager
    Configuration ..> SimulatorManager
    TicketBookingApplicationCLI ..> Configuration
    TicketBookingApplicationCLI ..> SimulatorManager
    DBManager ..> SimulatorManager
    DBManager ..> TicketPool
    LogManager ..> SimulatorManager
    LogManager ..> TicketPool
    AppController ..> Configuration
    AppController ..> SimulatorManager
    AppController ..> TicketPool
    TicketBookingSimApp ..> AppController
    TicketPool "1..1" *-- "0..*" Ticket

```

Figure 1: Class Diagram for the Ticket Booking Simulator

Sequence Diagram

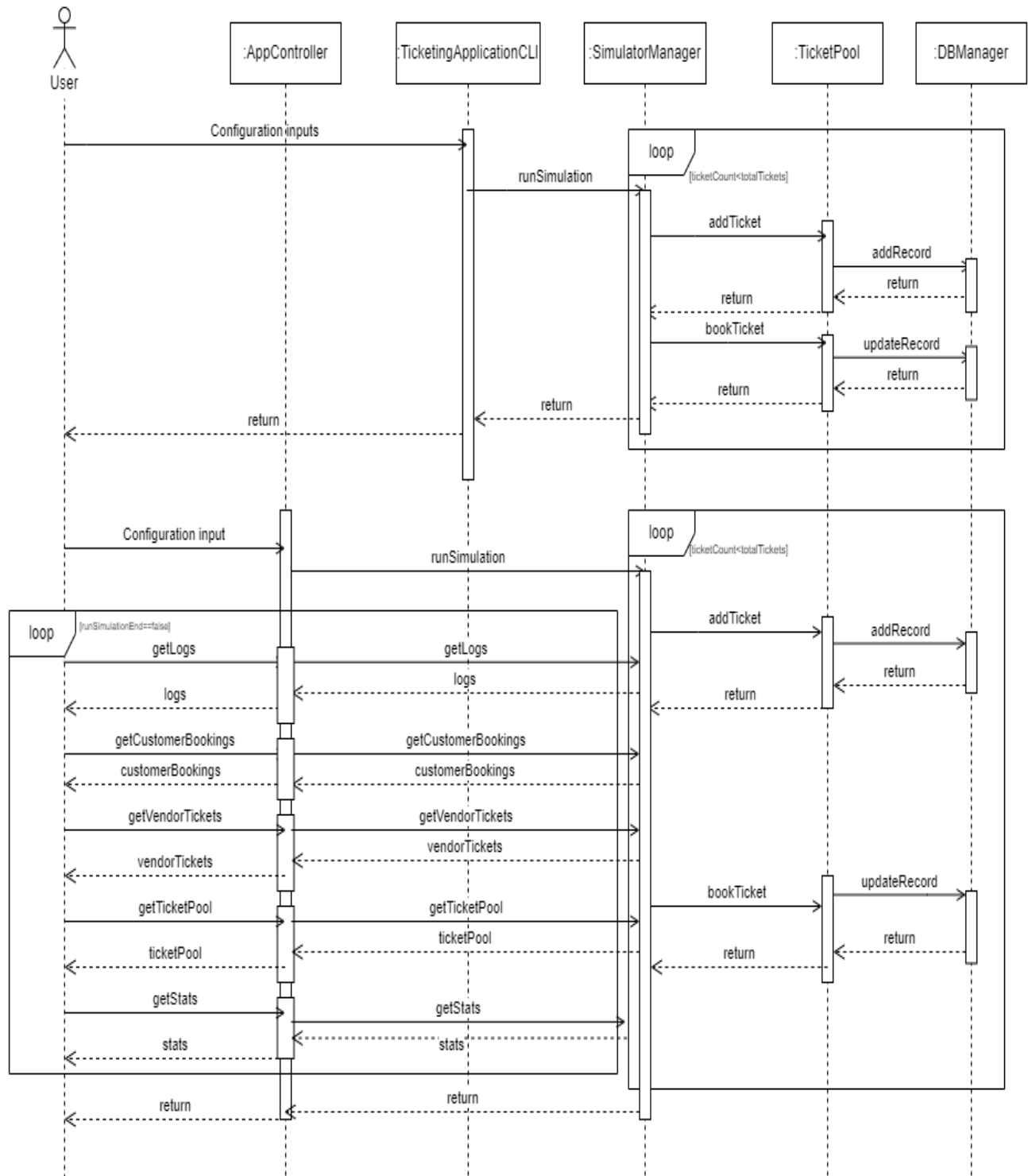


Figure 2: Sequence Diagram for the Ticket Booking Simulator

Test Case Report

Table 1: CLI Test Case

SCREEN NAME		Command Line Interface						
PREPARED BY		Vethum Buthgamuwa						
DESIGNATION		Tester/Developer						
DATE		6/12/2024						
SCENARIO ID	1	SCENARIO DESCRIPTION	Validate the user input and run the ticket booking simulation					
S.NO	TEST CASE ID	TEST CASE DESCRIPTION	PRECONDITION	TEST DATA	EXPECTED RESULT	POSTCONDITION	ACTUAL RESULT	STATUS (PASS/FAIL)
1	TC_CONFIGURATION_VALIDATION_001	Verify the system accepts valid inputs	CLI is Running	totalTickets: 100 ticketReleaseRate: 10 customerRetrieveRate: 5 maxTicketCapacity: 50	The simulation starts running	Simulation is Running	The simulation started running	Pass
2	TC_CONFIGURATION_VALIDATION_002	Verify the system handles values less than 1 for the configuration input without crashing	CLI is Running	totalTickets: 100 ticketReleaseRate: -5 customerRetrieveRate: 5 maxTicketCapacity: 0	The simulation re-prompts the input.	Input re-prompted	Input re-prompted	Pass
3	TC_CONFIGURATION_VALIDATION_003	Verify the system handles non-numeric values for the configuration input without crashing	CLI is Running	totalTickets: 100 ticketReleaseRate: 10 customerRetrieveRate: 5 maxTicketCapacity: abc	The simulation re-prompts the input.	Input re-prompted	Input re-prompted	Pass

--	--	--	--	--	--	--	--	--

Table 2:Simulation Execution Test Cases

SCREEN NAME		Simulation Execution						
PREPARED BY		Vethum Buthgamuwa						
DESIGNATION		Tester/Developer						
DATE		6/12/2024						
SCENA RIO ID	2	SCENARIO DESCRIPTION	Tests various conditions that arise while the simulation is running. (Applies to both CLI and Web Application)					
S.NO	TEST CASE ID	TEST CASE DESCRIP TION	PRECONDI TION	TEST DATA	EXPEC TED RESUL T	POSTCOND ITION	ACTU AL RESUL T	STAT US (PAS S/ FAIL)
1	TC_MAX_TICKET_CA PACITY	Verify the system handles maximu m ticket capacity reached.	CLI or web applicati on is Running	totalTickets: 200 ticketReleaseRa te: 70 customerRetrie valRate: 4 maxTicketCapac ity: 5	Maximu m ticket capacity reached messag e.	No additional tickets are created, and the simulation continues	Syste m blocke d additi onal ticket creatio n	Pass
2	TC_TOTAL_TICKETS_ REACHED	Verify the system handles total ticket count reached.	CLI or web applicati on is Running	totalTickets: 100 ticketReleaseRa te: 10 customerRetrie valRate: 5 maxTicketCapac ity: 50	Total ticket count reached messag e was display ed.	No additional tickets are created, and the simulation continues	Syste m blocke d additi onal ticket creatio n	Pass
3	TC_NO_TICKETS_AV AILABLE	Verify the system handles no tickets available for custome r when the ticket	CLI or web applicati on is Running	totalTickets: 100 ticketReleaseRa te: 5 customerRetrie valRate: 30 maxTicketCapac ity: 50	A messag e saying no tickets availabl e for custome r will be display ed.	Tickets will not be removed	A messag e saying no tickets availabl e for custom er will be display ed.	Pass

		pool is empty.						
--	--	----------------	--	--	--	--	--	--

Table 3: User Interface Test Cases

SCREEN NAME		User Interface						
PREPARED BY		Vethum Buthgamuwa						
DESIGNATION		Tester/Developer						
DATE		6/12/2024						
SCENA RIO ID	3	SCENARIO DESCRIPTION	User Interface Functionality is tested					
S.NO	TEST CASE ID	TEST CASE DESCRIP TION	PRECOND ITION	TEST DATA	EXPEC TED RESUL T	POSTCON DITION	ACTU AL RESU LT	STAT US (PAS S/ FAIL)
1	TC_CONFIGURATION_VALIDATION_001	Verify the system accepts valid inputs	Web Application is Running	totalTickets: 100 ticketReleaseRate: 10 customerRetrieveRate: 5 maxTicketCapacity: 50	The simulation starts running.	Simulation is Running	The simulation started running	Pass
2	TC_CONFIGURATION_VALIDATION_002	Verify the system handles values less than 1 as inputs for configuration values without crashing	Web Application is Running	totalTickets: 100 ticketReleaseRate: -5 customerRetrieveRate: 5 maxTicketCapacity: 0	The web app shows an alert saying the value must be greater than or equal to 1	Simulation does not run	The web app shows an alert saying the value must be greater than or equal to 1	Pass
3	TC_CONFIGURATION_VALIDATION_003	Verify the system handles non-numeric inputs for configuration	Web Application is Running	totalTickets: 100 ticketReleaseRate: 10 customerRetrieveRate: 5 maxTicketCapacity: abc	The application does not allow the user to enter a value	Simulation does not run	The application does not allow the user to enter a value	Pass

		values without crashing			that is not a number.		that is not a number.	
4	TC_CONFIGURATION_VALIDATION_004	Verify the system handles empty inputs for configuration values without crashing	Web Application is Running	totalTickets: [Blank] ticketReleaseRate: 10 customerRetrieveRate: 5 maxTicketCapacity: [blank]	The web app shows an alert saying the relevant field cannot be empty.	Simulation does not run	The web app shows an alert saying the relevant field cannot be empty.	Pass
5	TC_STOP_FUNCTION	the user clicks the stop button while the simulation is running.	The simulation is running.	Click stop button	The simulation stops running and shows an alert saying Ended.	The simulation stops running	The simulation stops running and shows an alert saying Ended.	Pass

Links

1. GitHub Repository Link: <https://github.com/vethumIIT/Tickets-Booking-Simulator.git>
2. Video Demonstration Link: <https://youtu.be/sPbViCDCplQ?si=cqemlFH2b1U90iBo>