3rd Assignment (2nd part of big assignment)

Common requirements for the assignments

You need to extend the program and the documentation of the 2nd assignment with the extensions provided in this document. The program is only acceptable if you also have the documentation.

Similarly to the 2nd assignment, there may be some things that are not defined. In these cases you are free to design and implement them as you see fit.

The documentation should include these additions

- an extended class diagram
- a state diagram
- two sequence diagrams

The code should include

- javadoc comments for each class and method (except getters and setters)
- unit tests for each of the important methods (not getters or setters)

1 University Library Management System

1.1 Additional functionality

As the university library evolves to support a wider variety of materials, borrowing behaviors and system decisions now adapt based on the nature of each item and the circumstances of the request. Members still interact through a unified interface, but behind the scenes, the process unfolds differently depending on the item type.

Books remain the most common item for borrowing. When a book is requested, the system checks for available physical copies. If at least one copy exists, the book is lent to the member, and both the item and member records are updated. If all copies are out, the book is marked as unavailable for others until a return occurs.

Theses are handled with similar rules to books. However, they are typically rarer, often with just one physical copy. While they can still be borrowed, no digital copies are produced. Their data include author, advisor, and defense year — is retained for academic referencing but does not influence borrowing logic.

Online Research Papers are not physically borrowed. When a member requests one (tries to borrow), the system prepares a digital access instance. The paper is immediately made available, regardless of how many others are viewing it. Each access is recorded as a view, and if the member chooses to download it, the system logs the download and increments the usage count.

1.2 State diagram

The state diagram for an Item in the University Library Management System captures the dynamic transitions between various states. Initially, an item is in the Available state, indicating that it can be borrowed. When a member borrows it, the item transitions to Borrowed, but only if there are available physical copies. If all copies are borrowed, it moves to Unavailable. Upon return, if the item is marked as Preserved, it returns to the Available state; however, if returned as Damaged, it transitions to a Needs Repair state. From there, the item either proceeds to the Being Repaired state or, if deemed severely damaged (based on a probabilistic condition), transitions to Archived, permanently removing it from circulation.

1.3 Scenarios to model with sequence diagrams

1.3.1 Scenario 1: Book Borrowing with Condition Tracking

A student borrows a book with multiple copies in circulation. The system verifies availability, assigns a copy to the student, and records the transaction. Upon return, the book is inspected and marked as preserved, incrementing the student's loyalty progress. The book becomes available again.

1.3.2 Scenario 2: Damaged Return and Removal

A faculty member returns a thesis copy in damaged condition. The system flags the item and sends it to the repair process. After evaluation, the damage is deemed irreversible. The thesis is removed from borrowing circulation and marked as archived.

2 Recipe Book System for a Restaurant

2.1 Additional functionality

To better reflect real kitchen workflows, the Recipe Book System introduces three distinct types of ingredients — Meat, Vegetables, and Bakery Items. Each calculates preparation time differently, based on the specific steps required to ready each type, allowing for more precise planning and coordination in the kitchen.

Meat typically require thawing if frozen, followed by cooking, making them one of the more time-consuming ingredients to prepare. Their total prep time is calculated by combining thawing time and cooking time, ensuring that dishes involving meat account for these essential stages. Vegetables, while faster to handle, still require cleaning and chopping. The system adds both washing and cutting times to determine their readiness, with variations depending on the type and freshness of the vegetable. Bakery Items, such as doughs and batters, involve a different kind of process — needing time to rest or rise before baking. Their total preparation time factors in both resting and baking durations.

2.2 State diagram

A recipe typically begins in a neutral idle state, waiting in the system until it is requested by a chef. Once selected, the system evaluates whether all necessary ingredients are available in stock. If everything is ready, the recipe proceeds directly to preparation. However, if even one required item is missing, the system temporarily halts the process, flags the missing ingredients, and adds them to the shopping list—placing the recipe on hold until restocking is complete. Once all components are present, the kitchen can begin the preparation. After the cooking is completed, the system finalizes the recipe's preparation, updates ingredient usage statistics, and resets the recipe's readiness, allowing it to be selected again in future cooking sessions.

2.3 Scenarios to model with sequence diagrams

2.3.1 Scenario 1: Preparing a Recipe with Complete Stock

The chef chooses a pasta recipe. The system confirms that all ingredients — including pasta, tomatoes, olive oil, and cheese — are available in the required quantities. The dish is prepared, and each ingredient's quantity in stock is reduced accordingly. The ingredient usage count is updated, and the preparation is logged.

2.3.2 Scenario 2: Missing Ingredient Handling

A dessert recipe is requested. However, the system detects that eggs are not available in stock. The preparation is halted, and “eggs” are added to the shopping list. The system marks the recipe as pending until the missing ingredient is purchased.

3 Venue Booking System in University

3.1 Additional functionality

The university offers three types of venues tailored to different needs: Small Team, Classroom-Sized Halls, and Big Event Halls. Team rooms are compact, seat up to five people, and can be booked instantly for short slots. They may include a whiteboard or a smart board. Bookings are free, and maintenance occurs automatically after every ten uses.

Classroom halls vary in size and always include a whiteboard and digital device and may include other equipment. The fee is based on a base amount (fixed for all classrooms) plus the number of equipment items in the room. Maintenance is performed after every 5 events.

Big Event Halls support large-scale activities and may include features like podiums, sound systems, or stages and other equipments. Fees are calculated based on both the number of seats and available equipment, with maintenance performed after every event.

3.2 State diagram

Each Classroom venue transitions through a set of operational states depending on its usage and condition. Initially, a venue is available and ready to be booked by users. Once a reservation is confirmed, it moves into a booked state, and when the scheduled event begins, the venue is marked as in use. After every five uses, it flags the venue for a standard maintenance check, placing it in a pending maintenance state. If the inspection reveals any issues, the venue enters an under repair phase, during which it becomes temporarily unavailable for booking. Once maintenance or repairs are completed, the venue becomes available again, restarting the cycle. These behind-the-scenes transitions help ensure that all venues stay in top condition and continue serving the university community reliably.

3.3 Scenarios to model with sequence diagrams

3.3.1 Scenario 1: Booking a Conference Room

A staff member browses available venues, The system filters and shows Conference Rooms. The user selects a room and requests additional equipment (e.g., microphone). The system checks room availability and equipment stock. Booking is confirmed, the room's booking count is incremented, and rental revenue is updated for the department.

3.3.2 Scenario 2: Booking Fails for an Big Event Hall

A university club member requests a Big Event Hall for an upcoming show. The system checks the venue's availability. The system finds the venue is not available, Booking is denied and the user is notified.

4 Orion Expedition

4.1 Additional functionality

As the Orion space shuttle ventures deeper into the cosmos, NASA has developed advanced systems to enhance its exploration capabilities. Each celestial region now includes a scientific value metric, observation completion percentage, and specific fuel consumption rate.

The navigation system now features a navigation mode with three distinct strategies to interstellar travel. Standard mode takes the direct approach found in the original design. Efficient mode factors fuel consumption into routing decisions by calculating a fuel-to-distance ratio for each destination. Exploration mode chooses destinations with the highest value, sometimes choosing longer paths deliberately to study cosmic phenomena of interest.

4.2 State diagram

The shuttle's operational behavior is now governed by a state model that modifies its functionality. Initially, the shuttle is in the docked state, where it can refuel but cannot move as the engine remains inactive. When the engine starts, it transitions to cruising state, traveling at standard velocity by consuming fuel per move and increasing travel progress by 5%. Upon reaching a destination, the shuttle enters orbiting state, performing observations with reduced fuel consumption at 3%. If fuel levels drop critically low, the shuttle transitions to emergency state, where it stops the observation and attempts to refuel. If refueling succeeds, the shuttle returns to the cruising state; if unsuccessful, the engine stops, and shuttle returns to docked state.

4.3 Scenarios to model with sequence diagrams

4.3.1 Scenario 1: Successful Planetary Visit

When the shuttle goes to a new destination, the navigation course begins with the selected destination. The engine starts, and the shuttle enters cruising state. As the shuttle advances, its movement gradually increases the destination's travel progress until it reaches 100%. Once complete, the current destination is updated, the destination is removed from the navigation system, and the closest destination becomes the next target using the active navigation strategy.

4.3.2 Scenario 2: Fuel Emergency Management

When the fuel drops below 15% of capacity, the shuttle enters Emergency state. The system attempts refueling with a 50% success chance. If successful, the fuel is restored to maximum, and the mission continues. If unsuccessful, the engine stops, halting progress until the situation is resolved. The shuttle can implement an enhanced emergency protocol that enables solar energy collection in deep space, modifying the random refueling chance based on proximity to stars, or calculating a minimum-energy return trajectory to the nearest known refueling point by selecting the closest destination with refueling capabilities.

5 Hospital Emergency

5.1 Additional functionality

The city's emergency medical system has been upgraded with advanced protocols and technologies to improve resource utilization during crisis situations.

The hospital now features a treatment strategy with three specialized variants, each handling patient care differently. Standard treatment applies doctors' efficiency directly to reduce patient severity following established protocols. Intensive treatment adds a 1.5x efficiency multiplier for patients with severity above 5, using more resources for faster healing. Emergency treatment sorts the patients by urgency using a formula (severity/10 + 1) to boost efficiency for critical cases.

5.2 State diagram

The emergency system tracks patients through a state model that manages the medical process. Initially, the patients enter the waiting state, where they undergo initial assessment with severity assigned but remain not transported. When an ambulance is dispatched, patients enter the transit state as they are transported to a hospital. Upon arrival, patients transition to the admitted state, where they are successfully added to the hospital's patient list. When a doctor becomes available, patients move to the "under treatment" state, receiving care with severity gradually reduced. Finally, when severity reaches zero, patients transition to a recovered state, marked as treated and ready to discharge.

5.3 Scenarios to model with sequence diagrams

5.3.1 Scenario 1: Routine Emergency Response

When a new patient arrives with assessed name and severity, the emergency alert system identifies the best facility and dispatches an ambulance. The ambulance is assigned to the hospital and transports the patient. Upon arrival, the hospital registers the patient, then begins treatment. Doctors apply their efficiency to reduce patient severity until the patient is fully treated and can be discharged.

5.3.2 Scenario 2: System Under Capacity Constraints

During mass casualty events with many patients, the system's capacity is tested when hospitals approach their patient limits. As the system struggles to find hospitals with available capacity, incoming patients must wait. The system implements priority scheduling by sorting patients by severity before treatment, ensuring doctors treat critical patients first even under constraints. If all hospitals reach capacity, the system activates additional resources by temporarily increasing hospital capacity by 25% for emergency overflow and implements accelerated treatment protocols. Throughout this process, each patient's state transitions are carefully tracked to ensure no cases are overlooked.

6 Adventurer's Guild

6.1 Additional functionality

The Adventurer's Guild has expanded its operations with advanced combat systems, equipment management, and quest logistics to better prepare its members. Each guild is now offering a repair service to restore weapons' durability.

The Guild now offers three specialized weapon types that extend the base weapon system. Swords excel in melee combat, delivering powerful strikes with a 20% change of inflicting critical hits that double damage output — though they lose 2x durability points. Bows specialize in range combat, providing precision attacks that bypass 30% of enemy's armor, with durability reduction proportional to the target's armor durability. Staffs harness magical energy, launching spells with a 25% chance to halve a monster's strength, with durability consumption proportional to the staff's strength.

6.2 State diagram

The Guild's equipment system now tracks weapons through four distinct states that affect combat performance. In the new state (100% durability), weapons provide their full power bonus with a 10% "first use" bonus damage effect. After first use, weapons transition to the used state (50-99% durability), maintaining standard performance with their regular power boost. With continued use, weapons enter a damaged state (1-49% durability), with their power boost reduced to 75%. When durability reaches zero, weapons transition to broken state, becoming unusable. The next time a warrior attempts a quest, they try to repair their weapons by spending experience points proportional to each of weapons' power -– if successful, the weapon returns to the new state.

6.3 Scenarios to model with sequence diagrams

6.3.1 Scenario 1: Successful Quest Completion

When an adventurer attempts a quest, they first equip appropriate weapons. During monster fights, the adventurer uses their strongest non-broken weapon, dealing damage based on calculated power against monster armor. As the adventurer defeats monsters, they gain experience that increases rank and base power by 20% when experience reaches threshold values. After defeating all monsters in the quest's list, the quest is marked as complete, and the adventurer recovers full health.

6.3.2 Scenario 2: Equipment Failure

During challenging quests against high-health monsters, weapons sustain damage with each use, decreasing durability. As durability falls below 50%, the weapon enters the damaged state, reducing power by 25%. If durability reaches zero during combat, the weapon provides no power boost. This sudden power reduction may leave the adventurer unable to overcome the monster's armor value, causing them to take damage without dealing any in return. If defeated, the quest attempt fails, and the adventurer must recover before attempting another quest.

7 Borderland Survival

7.1 Additional functionality

Psychological tensions and loyalties are introduced into the Borderland to create room for player agency and strategic play.

Every character has a behaviour type – Rational, Aggressive, Cautious, or Unstable – that determines the manner in which he or she responds to fear and game type. Each influences them behind the scenes to influence their decision-making, fear gain, and play. For example, Cautious characters avoid dangerous games, and Aggressive characters ignore fear but utilize stamina.

The system also has trust-based relations that depreciate over time without being solidified by mutual success or card trades. Betrayals invoke fear and alter the behaviour of the characters.

7.2 State diagram

Design a UML state machine that starts every character in the Rational state and allows transitions to Aggressive, Cautious, or Unstable based on simple triggers: fear spikes push Rational to Cautious and Cautious to Unstable, betrayal drives Rational/Cautious to Aggressive, prolonged aggression with low stamina turns Aggressive to Unstable, while rest or trust-building can move Aggressive/Cautious back toward Rational and Unstable to Cautious to Rational.

7.3 Scenarios to model with sequence diagrams

7.3.1 Scenario 1: Typical game

Draw a sequence diagram for the “Start Game” interaction: the Borderland Manager instructs the Game Master to begin the next round; the Game Master randomly selects a Game from the Arena’s queue, notifies the Arena to set itself up, and then broadcasts a “game start” message to all Players, who each respond by checking their relevant stat (mental or physical) against the Game’s difficulty and returning either “win” or “lose”; the Game Master tallies results, updates each Player’s fear level, and awards cards to winners before reporting back to the Borderland Manager.

7.3.2 Scenario 2: Victory!

Create a sequence diagram depicting the moment a Player wins their last needed card: the Game object notifies the Game Master of the Player’s victory, the Game Master updates the Player's card collection, discovers the set now contains all 52 unique cards, and immediately informs the Borderland Manager. The Manager validates the full deck, broadcasts an “exit granted” message, orders the Arena to open an exit path, and logs the event for spectators while the victorious Player sends a final “leave borderland” confirmation back to both the Game Master and Manager.

8 Wildlife Simulation

8.1 Additional functionality

To introduce more unpredictability of the wild, the simulation includes animal behavior and ranger response patterns to handle high-risk encounters and medical urgency.

Every animal now has a temperament profile — Passive, Defensive, Curious, or Aggressive — which will influence the animal's reaction when approached. Aggressive animals attack or flee and stress surrounding animals. Defensive animals only defend themselves when cornered, and Curious animals will approach rangers uninvited. Temperament influences how rangers must prepare and interact.

The system also tracks ranger stress, which builds up while conducting dangerous rescues or poacher encounters. If stress becomes too great, the ranger will slow down or make more dangerous choices. Trucks now also have tranquilizer kits or mobile medic units as optional equipment during an emergency, depending on truck type.

8.2 State diagram

The state diagram should represent the behavior of the animal throughout the interaction with a ranger. The animals begin in a Calm state. As the ranger approaches, depending on their disposition (temperament profile), they might enter an Alerted, Defensive, Curious, or Aggressive state. While threatened or injured, they might enter Fleeing or Attacking states. Successfully administered treatment brings the animal to a Stabilized state, but severe injury or increasing stress can bring it to a Critical state that requires immediate attention.

8.3 Scenarios to model with sequence diagrams

8.3.1 Scenario 1: Hostile Animal Encounter

A ranger attempts to administer treatment to an injured animal with an Aggressive profile. The animal bites when the ranger gets close, so the ranger retreats and calls for help. Stress is increased, and the habitat is shut down as unsafe for a short period of time.

8.3.2 Scenario 2: Emergency Treatment in the Field

An attack by poachers leaves an animal critically injured. A ranger comes in a vehicle equipped with a med unit and speeds to the animal's rescue right away. The animal is stabilized right there and is driven directly to the rehabilitation zone by the vehicle.

9 Racing Tournament Simulation

9.1 Additional functionality

The race competition has also been supplemented with track modifiers, environmental hazards, and adaptive driving logic for realistic high-stakes simulation.

Weather conditions such as rain and fog affect vehicle performance. Rain decreases grip and affects effective speed and gas mileage. Fog decreases visibility, so it decreases speed even further than rain. Fog also necessitates a pit stop.

A new vehicle condition system adds optimal, worn, and critical. As durability degrades, performance decay — optimal cars operate with all stats, worn cars experience moderate loss of speed, and critical cars misfire and require emergency fix.

9.2 State diagram

The state diagram will have to include the life cycle of a race car during the race. At the start, the car is in Optimal state. Depending on track wear and weather during the course of the race, the car advances to Worn state. The car depletes further until it is at Critical state where slowness and misfires occur. If the pit-stop is performed successfully, it returns the car to Optimal; otherwise, it may remain Critical and run the risk of engine failure. Engine failure sets the car into a Retired state, losing the race for that car.

9.3 Scenarios to model with sequence diagrams

9.3.1 Scenario 1: Driver Completing a Lap

Model the sequence of events when a driver completes a lap during a race. Show the interaction between the Driver, their Car, and the RaceTrack. Include the calculation of effective speed, fuel consumption, and durability reduction based on the car's state, driver's skill, preferred model bonus (if applicable), track difficulty, and current weather conditions.

9.3.2 Scenario 2: Pit Stop

Illustrate the process of a driver taking a pit stop to repair their car when its durability state becomes Critical. Show the Driver initiating the pit stop, interacting with the assigned Mechanic, the Mechanic performing repairs on the Car, and the Car's state potentially improving back to "Optimal".