Use Cases

Avihoo Amos
Hila Elbaz
Kfir Shalom
Mor Shay
Tamir Oshri Avisar
Yuval Roth

TABLE OF CONTENTS

SYSTEM

1) System startup

Actors: Administrator
Precondition:

1. The market system is being initialized according to an external configuration file that defines the initialization parameters.

Parameters: API keys, Object of user administrator
Postcondition:

2. The system will have an assigned administrator.
2. The market system has a connection to payment and delivery services.

Actions:

1. The administrator will run the application and the system will load.
2. After system bootup, the system will connect to the payment and shipment services.
3. Assignment of a system administrator.

Acceptance tests:

positive: system is booted up and there is an assigned administrator and the external services are connected
negative: System boots up but external services are failing to connect
alternative: wrong api keys or administrator credentials are entered and the system fails to load correctly, prompting a message to the administrator. After fixing the problem the system boots successfully

2) Adding / removing / changing an external service

Actors: System administrator
Precondition: The system is running normally, and logged in system administrator
Parameters: The details of the external service to be added to the system
Postcondition: The system supports an additional external service
Actions:

1. The administrator selects adding/removing/changing an external service
2. choose a type of external service to add
3. fill the required details

Acceptance tests:

positive: A new external service is added and works as expected
negative: A new external service is added with all the correct details but connection to the service fails for an unknown reason
negative: add an external service that already exist, and fails
negative: trying to delete the only payment service, and fail as long there is no at least another way to pay.
negative: change an external service with illegal details, the action fails and save the previous state
alternative: upon adding a new external service, wrong details of the external service were entered and connection to the service failed. Upon fixing the details the service connection was successful.

3) Payment

Actors: System, user
Precondition: There is an order that is pending payment
Parameters: The order details and payment method
Postcondition: The order is paid successfully
Actions:

1. A user initiates a payment for an order
2. The system gathers the order details from the user's shopping cart
3. The system gathers the payment method from the user's profile
4. The system contacts the appropriate payment service provider and charges the user based on the shopping cart.

Acceptance tests:

positive: A user successfully paid for an order
negative: A payment failed for an unknown reason at the external payment system
negative: The user close or not complete the payment process, the purchase fails
negative: The connection to the external payment system failed, the payment failed
negative: The price amount is 0 (because of coupon or gift) so the purchase done automatically without access to external payment system
alternative: A payment failed due to a problem with the details of the payment method. The user was prompted and fixed the details. Afterwards the payment was successful.

4) Supply

Actors: System, user
Precondition: There is an order that has been paid
Parameters: The order details, the user shipping details
Postcondition: The shipping details and the order details are sent to the shipping provider for further handling.
Actions:

1. The user paid for an order
2. The system gathers the order details from the user's shopping cart
3. The system gathers the shipping details from the user's profile
4. The system contacts the appropriate shipping service provider and sends it the shipping information and the order details, meaning the info about which items needs to be shipped.

Acceptance tests:

positive: Shipping provider confirms receipt of shipping information and begins processing the order for delivery.
negative: System fails to contact the shipping service provider due to network issues.
negative: The address is out of supplying area, the purchase fails
negative: The address is invalid (like street number doesn't exist), the purchase fails
alternative: User receives a notification about the error and corrects the shipping information. Upon correction, the system successfully contacts the shipping service provider and initiates the shipping process.

5) Real time notifications

A. Store owner gets a real time notification (new transaction occurred*)

Actors: system, store manager (passive actor), user (buyer)
Precondition: A new transaction took place
Parameters: none
Postcondition: The notification about the transaction has been sent to the store owner
Actions:

1. A buyer makes a purchase
2. The system finds the store where the transaction took place
3. The system sends a new real time notification to the owner of that store with the transaction details

* similar steps occur when the notification is about a store has been closed or when a store owner was removed from his position.

Acceptance tests:

Positive: There was a transaction. Real-time notification is sent to the store owner with transaction details.
negative: System fails to identify the store associated with the transaction due to a technical issue. Store owner does not receive the notification.
alternative: System identifies the store but encounters an error while sending the notification. Store owner does not receive the initial notification but receives a follow-up notification once the issue is resolved.

B. A user gets notified of a new message or a request

Actors: user (passive), system
Precondition: user received a request or a message
Parameters: none
Postcondition: the user received the notification
Actions:

1. A user receives a new request / message
2. The system displays the notification in real time

Acceptance tests:

Positive: User receives a new message or request. System displays the notification to the user in real-time.
negative: User receives a new message or request. System fails to display the notification due to an uknown error.
alternative: User receives a new message or request. System displays the notification, but the user's device is offline, preventing immediate notification display. Upon device connectivity, the user receives the notification.

6) Delayed notifications

Actors: user, system
Precondition: A user received a delayed notification
Parameters: none
Postcondition: On the next login the user will receive the notification
Actions:

1. A user receives a new request / message
2. The system displays the notification the next time the user logs into the system

Acceptance tests:

positive: User receives a new message or request. System displays the notification during the next login.
negative: User receives a new message or request. System fails to display the notification during the next login due to an unknown issue.
alternative: User receives a new message or request. System displays the notification during the next login, but the user fails to retrieve the notification due to connectivity issues. Upon device connectivity, the user receives the delayed notification.

VISITOR ACTIONS REQUIREMENTS

1) Guest entry

Actors: Guest
Precondition: The user accesses the trading system's login page.
Parameters:
Postcondition:

1. The guest is logged into the trading system as a guest.
2. A shopping cart is assigned to the guest.
3. The guest can function as a buyer within the system.

Actions:

1. Guest accesses the trading system.
2. System automatically assigns a guest status to the user upon accessing the system.
3. A shopping cart is created and assigned to the guest.
4. Guest is now able to browse and interact with the system as a buyer.

Acceptance tests:

Positive: Unregistered user can search for products.
Positive: The guest user can add and remove products from the cart
Negative: The guest attempts to access the trading system and submits invalid parameters or missing required data. The system detects the invalid request and rejects it.
Alternative: The guest attempts to access the trading system. An error occurs during the process of assigning guest status. The guest is unable to browse or interact with the system as a buyer.

2) Guest exit

Actors: Guest
Precondition: The guest is currently logged into the trading system.
Parameters: none
Postcondition:

1. The guest leaves the trading system.
2. The guest loses their shopping cart.
3. The guest is no longer defined as a visitor in the system.

Actions:

1. Guest indicates the intention to leave the trading system.
2. System removes the guest's shopping cart.
3. Guest is logged out of the system.
4. Guest is no longer defined as a visitor within the system.

Acceptance tests:

Positive: Guest indicates the intention to leave the trading system. The system removes the guest's shopping cart and the guest is logged out of the system.
Negative: Guest tries to leave the system without indicating the intention to exit. The system does not remove the guest's shopping cart and the guest remains logged into the system.
Alternative: Guest indicates the intention to leave but decides to stay in the system.
System does not remove the guest's shopping cart immediately and guest remains logged into the system.

3) Registration

Actors: Guest, System
Precondition: The guest is accessing the registration page of the trading system.
Parameters: Unique identification details (e.g., email address, username, password)
Postcondition:

1. The guest successfully registers as a user in the system.
2. The user gains access to additional features and privileges.
3. To achieve subscriber status, the user must log in with their provided identification details.

Actions:

1. Guest requests to register.
2. Guest provides unique identification details such as email address, username, and password.
3. System validates the provided details for uniqueness and completeness.

- If the provided details are valid:
  System registers the guest as a user.
  Guest is prompted to log in with their provided identification details to achieve subscriber status.
- If the provided details are invalid or incomplete:
  System prompts the guest to correct the errors and resubmit the registration form.

Acceptance tests:

Positive: Guest provides unique identification details and the system successfully registers the guest as a subscriber.
Negative: Guest provides incomplete identification details. The system prompts the guest to correct the errors and resubmit the registration form.
Negative: A guest provides invalid details, like illegal username, the registration fails and the system asks to correct the details.
Negative: A guest provides details the system can't approve (like taken username, or too young age to use the system for example), the registration fails and the system asks to correct the details.
Alternative: Guest attempts to register with identification details already associated with an existing account. The system detects the duplicate details and prompts the guest that the details are already in use.

4) Login

Actors: User (Subscriber)
Precondition: The user has registered and has identification details.
Parameters: Username, Password
Postcondition: The user successfully logs in and is recognized as a visitor-subscriber in the market.
Actions:

2. User provides their username and password for logging in.
3. System validates the entered credentials.

- If the credentials are valid:
  User is logged in.
  User is recognized as a visitor-subscriber in the market.
- If the credentials are invalid:
  User is prompted with an error message indicating invalid credentials.
  User is given the option to retry entering their credentials or reset their password.

Acceptance tests:

Positive: User enters valid username and password and the system successfully authenticates the user.
Negative: User enters an invalid username or password and the system fails to authenticate the user.
Alternative: User enters incorrect password multiple times, exceeding the allowed attempts. The system locks the user's account temporarily for security reasons.

PURCHASE ACTIONS OF A VISITOR

1) View Stores and Products (Guest)

Actors: Guest
Precondition: The visitor is logged into the trading system as a guest.
Parameters:
Postcondition: The visitor obtains information about stores and products available in the market.
Actions: A guest can get information about stores and products available in the market.
Acceptance tests:

Positive: Visitor can get information about relevant stores and products.
Negative: There is no yet stores or available products on some store. The market give empty stores list or product list of a store
Negative: System fails to provide stores due to technical issues and the visitor cannot view any stores or products and get an error.
Alternative: Visitor selects a store to view its products. System displays the store but there are no products listed due to a database error or store configuration issue, the system provides an error message instead.

2) Search products

a. Actors: Guest
Precondition: The visitor is logged into the trading system as a guest.
Parameters: productName, category ,keywords
Postcondition: The list is sorted
Actions:

1. The system asks you to select the type of search you want - product name, category, keywords
2. User chooses
3. The system verifies that he has chosen a valid option and returns him the search results if he was successful. otherwise throws an error.
4. The system offers a sorting option by price range, product rating, category, store rating
5. The user chooses
6. The system displays according to the user's choice, with the default being sorting by price range

Acceptance Test:

Positive: the user insert correct parameters so the system should show the search result
Negative: the user inserts incorrect parameters (like text parameter instead of numeric, etc.) so the system should throw an error
Negative: the user inserts conflicting parameters (like price>100 and price<50), the system should return an empty result
Alternative: The user provides parameters that return an empty or short list, the system throw error.

b. The same as before, except that now the option to sort by store rating will not be given.

3) Save products

Actors: Guest
Precondition: The visitor is logged into the trading system as a guest.
Parameters: shopId, list(product)
Postcondition: Products are kept in the cart
Actions:

1. The system asks the user for the store details, and the products they wish to keep in the basket
2. The user enters his choice
3. The system checks that everything is correct, if not correct - does not save and gives a note that it was not saved

Acceptance Test:

Positive: the user inserts correct paramaters so the system should save the products
Negative: the user inserts incorrect paramaters so the system should throw an error
Negative: the user inserts amount that is bigger than the store supply, the system should return an error
Alternative: the user choose product that out of stock, so the system should not save.

4) Checking shopping cart

Actors: Guest
Precondition: The visitor is logged into the trading system as a guest.
Parameters:
Postcondition: Products are kept in the cart
Actions:

1. The system shows the user the shopping cart
2. The system allows the user to change the products in the shopping cart - add/delete
3. If the user has made changes, the system will show him an updated list

Acceptance Test:

Positive: The user chooses to see the latest shopping cart, so the system will have to show him the latest list
Negative: The user will try to change a product improperly, the system will throw an error
Alternative: The user will delete a product from the shopping cart, so the system will save the change and display an up-to-date cart

5) Buy products of cart

Actors: Guest
Precondition: The visitor is logged into the trading system as a guest.
Parameters: payment details
Postcondition: the cart is saved in system – The purchase is saved to the system
Actions:

1. The system will check the compliance of each cart with the store policyThe system will calculate the discounts
2. The system will allow the user to pay the final price
3. The user will enter payment information
4. The system will check whether the purchase can be completed according to the policy
    a. If a purchase can be completed, it will complete the purchase successfully
    b. Otherwise it will update the user that some of the products cannot be purchased

Acceptance Test:

Positive: The products in the cart are in stock, and the discounts were calculated, so the system will complete a purchase successfully
Negative: There are products that cannot be purchased because of the policy (like selling alcohol to 18+), therefore the system will not allow purchasing and will exit the operation
Negative: The payment system returned an error by some reason, the system will return an error
Alternative: The user wants to buy something that is not in stock, the system will not allow to buy

PURCHASE ACTIONS OF A SUBSCRIBER

0.1) View Stores and Products (Subscriber)

Actors: Subscriber
Precondition: The visitor is logged into the trading system as a subscriber.
Parameters:
Postcondition: The visitor obtains information about stores and products available in the market.
Actions: A subscriber user can get information about stores and products available in the market.
Acceptance tests:

Positive: Visitor can get information about relevant stores and products.
Negative: There is no yet stores or available products on some store. The market give empty stores list or product list of a store
Negative: System fails to provide stores due to technical issues and the visitor cannot view any stores or products and get an error.
Alternative: Visitor selects a store to view its products. System displays the store but there are no products listed due to a database error or store configuration issue, the system provides an error message instead.

0.2) Search products

Actors: subscriber
Precondition: The visitor is logged into the trading system as a subscriber.
Parameters: productName, category ,keywords
Postcondition: The list is sorted
Actions:

1. The system asks you to select the type of search you want - product name, category, keywords
2. User chooses
3. The system verifies that he has chosen a valid option and returns him the search results if he was successful. otherwise throws an error.
4. The system offers a sorting option by price range, product rating, category, store rating
5. The user chooses
6. The system displays according to the user's choice, with the default being sorting by price range

Acceptance Test:

Positive: the user insert correct parameters so the system should show the search result
Negative: the user inserts incorrect parameters (like text parameter instead of numeric, etc.) so the system should throw an error
Negative: the user inserts conflicting parameters (like price>100 and price<50), the system should return an empty result
Alternative: The user provides parameters that return an empty or short list, the system offers similar results

b. The same as before, except that now the option to sort by store rating will not be given.

0.3) Save products

Actors: subscriber
Precondition: The visitor is logged into the trading system as a subscriber.
Parameters: shopId, list(product)
Postcondition: Products are kept in the cart
Actions:

1. The system asks the user for the store details, and the products they wish to keep in the basket
2. The user enters his choice
3. The system checks that everything is correct, if not correct - does not save and gives a note that it was not saved

Acceptance Test:

Positive: the user inserts correct paramaters so the system should save the products
Negative: the user inserts incorrect paramaters so the system should throw an error
Negative: the user inserts amount that is bigger than the store supply, the system should return an error
Alternative: the user choose product that out of stock, so the system should not save.

0.4) Checking shopping cart

Actors: subscriber
Precondition: The visitor is logged into the trading system as a subscriber.
Parameters:
Postcondition: Products are kept in the cart
Actions:

1. The system shows the user the shopping cart
2. The system allows the user to change the products in the shopping cart - add/delete
3. If the user has made changes, the system will show him an updated list

Acceptance Test:

Positive: The user chooses to see the latest shopping cart, so the system will have to show him the latest list
Negative: The user will try to change a product improperly, the system will have to throw an error
Alternative: The user will delete a product from the shopping cart, so the system will save the change and show an up-to-date cart

0.5) Buy products of cart

Actors: subscriber
Precondition: The visitor is logged into the trading system as a subscriber.
Parameters: payment details
Postcondition: the cart is saved in system – The purchase is saved to the system
Actions:

1. The system will check the compliance of each cart with the store policyThe system will calculate the discounts
2. The system will allow the user to pay the final price
3. The user will enter payment information
4. The system will check whether the purchase can be completed according to the policy
5. If a purchase can be completed, she will complete the purchase successfully
6. Otherwise you will update that some of the products cannot be purchased

Acceptance Test:

Positive: The products in the cart are in stock, and the discounts were calculated, so the system will complete a purchase successfully
Negative: There are products that cannot be purchased because of the policy (like selling alcohol to 18+), therefore the system will not allow purchasing and will exit the operation
Negative: The payment system returned an error by some reason, the system will return an error
Alternative: The user wants to buy something that is not in stock, the system will not allow to buy

1) Logout

Actors: Subscribed user
Precondition: The user is logged into the trading system as a guest.
Parameters: userId
Postcondition: the cart is saved in system
Actions:

1. The system will allow the user to log out
2. The user will be disconnected from the system
3. The system will remove him from the system as a subscribed user
4. The system will save his shopping cart for next time

Acceptance Test:

Positive: The user chose to disconnect, so the system disconnects him and saves his shopping cart
Negative: The user's username is incorrect for disconnection, so the system should throw an exception
Alternative: None

2) Creating new store

Actors: user
Precondition: A subscribed logged in user
Parameters: storeName
Postcondition: New store created. The user becomes the founder of the store
Actions: A registered user creates a new store and becomes its founder
Acceptance Test:

Positive: A subscribed user tries to open a store with a unique name in the market and succeeds
Negative: A guest user tries to open a store with a unique name but fails without registration
Negative: A subscribed user tries to open a store with invalid parameters (For example a name that consists only of emojis), the operation fails
Alternative: A subscribed user tries to open a store with an occupied name, the system will suggest a similar unique name

ACTIONS OF A STORE-OWNER

1) Inventory management

Actors: subscribed user
Precondition: The user is a store manager with editing permissions or a store owner
Parameters: productName, productDetails
Postcondition: The product added/removed/changed
Actions: A store manager adds/removes/changes product in his store
Acceptance Test:

Positive: A subscribed user with editing permissions for a store tries to edit its inventory legally and succeeds
Negative: A guest user tries to edit a store inventory but fails
Negative: A subscribed user with editing permission for a store tries to change the amount of some product in this store to negative number, but fails
Negative: A subscribed user with editing permission for a store tries to edit another store inventory, and fails
Negative: A subscribed user with only view permission for a store tries to edit its inventory, and fails
Alternative: A subscribed user without editing permissions for a store tries to edit its inventory and fails, but can ask for permission from the owners' team

2) Policy/discount change

Actors: subscribed user
Precondition: The user is a store manager with editing permissions or a store owner
Parameters: discountPolicy
Postcondition: The policy/discount added/removed/changed
Actions: A store manager adds/removes/changes policy/discount in his store
Acceptance Test:

Positive: A subscribed user with policy editing permissions for a store tries to edit its discount or policy and succeeds
Negative: A guest user tries to edit a store policy or discount but fails
Negative: A subscribed user with policy editing permissions for a store tries to add a discount for a product has already a discount, and fails until he will delete the exist discount, or suggests him edit the discount instead.
Negative: A subscribed user with policy editing permissions for a store tries to add a discount is greater than 100 or less than 1, and fails
Negative: Trying to add a policy that conflicts with another policy, and failure
Alternative: A subscribed user without editing permissions for a store tries to edit its discount or policy and fails, but can ask for permission from the owners' team

3) Appointing store owner

Actors: Store owner, subscribed user
Precondition: The subscribed user is not the store owner
Parameters: storeName
Postcondition: The subscribed user is a store owner
Actions:

1. A store manager appoints new registered user to the store owners' team
2. The user approves the appointment

Acceptance Test:

Positive: A store owner appoints new subscribed user to the store owners' team, the user approves the request, and successfully becomes one of the owners

Negative: A store owner appoints another user is already owner of that store, and fails

Negative: A guest or a subscribed user that is not a store owner trying to perform the action, and fails

Alternative: A store owner appoints a new subscribed user to the store owners' team, the user doesn’t approve the request, and the appointment failed. Allows the owner to send him or another user a repeat request

6) Appointing store manager

Actors: Store owner, subscribed user
Precondition: The subscribed user does not manage the store
Parameters: storeName, permissionType
Postcondition: The subscribed user manages the store
Actions:

1. A store owner appoints new user to the store management team with some permissions
2. The user approves the appointment

Acceptance Test:

Positive: A store owner appoints new subscribed user to the store managers' team, the user approves the request, and successfully becomes one of the managers

Negative: A store owner appoints another user is already owner or manager of that store, and fails

Negative: A store owner appoints a username doesn't exist in the system, and fails

Alternative: A store owner appoints a new subscribed user to the store managers' team, the user doesn’t approve the request, and the appointment failed. Allows the owner to send him or another user a repeat request

7) Change Permissions

Actors: Store manager, store owner
Precondition: the store owner appointed another store manager
Parameters: storeName
Postcondition: the store manager has it's permissions changed
Actions: store owner changes the permissions of the store manger he appointed.
Positive: The store owner successfully changes the permissions of the store manager. The system updates the permissions and confirms the update to the store owner.
Negative: The store owner attempts to change permissions for a non-existent manager, the system does not change any permissions and alerts the store owner that the manager could not be found.
Alternative: The store owner tries to assign permissions exceeding their own level, the system prevents the change and notifies the owner of the permission limitation.

9) closing store

Actors: store founder, store owner, store manager.
Precondition: store founder opened a store. Store founder is logged in to the system.
Parameters: storeName
Postcondition: the store becomes non active. Store manager and store owner gets notified that the store got closed.
Actions: store founder closes the store he founded. Store manager and store owner gets notified that the store closed
Positive: The store founder successfully closes the store, the store is marked as non-active, and both the store manager and store owner are notified.
Negative: The store founder tries to close an already non-active store. The system does nothing and informs the founder that the store is already closed.
Alternative: The store founder is not logged in but tries to close the store, the system denies the action and requires the founder to log in.

11) View workers and permissions

Actors: Store manager, Store owner
Precondition: at least one store manager exists.
Parameters: storeName, storeOwner
Postcondition: none
Actions: store owner views the other store manager and their permissions
Store owner views the store workers.
Positive: The store owner views the permissions of store managers, the system shows the permissions of each manager associated with the store.
Negative: No store managers are currently appointed, the system displays a message that no managers are available.
Alternative: The store owner requests to see permissions of workers other than managers. The system displays the permissions for all roles the owner is authorized to view.

13) purchase history

Actors: user
Precondition: account logged in
at least one purchase has been made with the logged in account
Parameters: username, storeName
Postcondition: none
Actions: user views his purchase history in the given store
Positive: A user views their purchase history successfully. The system displays the purchase history for that store.
Negative: A user with no purchase history tries to view their purchases, the system informs them that no purchase history is available.
Alternative: A user tries to view another user's purchase history. The system denies access and informs the user of the lack of permission.

ACTIONS OF A STORE-MANAGER

1) permission check

Actors: Store manager
Precondition: the store manager is logged in to the system and has at least one permission.
Parameters: none
Postcondition: none
Actions: the store manager does some things that require permission given to him by the store owner.
Positive: Store manager performs an action within their permissions. The action completes successfully.
Negative: Store manager attempts to perform an action without the necessary permissions. The system prevents the action and alerts the manager of the permission violation.
Alternative: Store manager requests a review of their permissions. The system provides a summary of current permissions and offers guidance on how to request changes if needed.

ACTIONS OF A MARKET ADMINISTRATOR

4) manager views purchase history

Actors: market administrator
Precondition: market administrator is logged in to the system
Parameters: storeName, customerName
Postcondition: none
Actions: market administrator views the purchase history of a given store or a given customer
Positive: An administrator views a user's purchase history successfully. The system displays the purchase history for that store.
Negative: A user with no administrator permissions tries to view a user's purchases. The system informs them that they have no permissions to view history.
Negative: An administrator tries to see a purchase history of a store with no purchases yet, and receive an empty list.
Alternative: A administrator tries to view another user's purchase history but the user hasn't made any purchases , the system shows an error informs the manager that the user has no purchases.

SYSTEM RESILIENCE AND STABILITY

1) Unexpected user input

Actors: user
Precondition: The user is on a form page within the application.
Parameters: Form fields that expect specific data types.
Postcondition:

1. The system remains stable and responsive.
2. The user receives feedback on invalid input.

Actions:

1. User inputs invalid data in a form field (e.g., text in a numeric field).
2. The system validates the input data type before processing.
3. If the input is invalid, the system displays an error message.
4. The system prompts the user to correct the input without crashing.

Acceptance tests:

- Positive: The user inputs valid data (e.g., numbers in a numeric field). The form submits successfully. The system processes the valid input without issues.
- Negative: User inputs text in a numeric field. The form does not submit, and an error message is displayed. The system validates the input, detects the invalid data type, and prompts the user to correct the input.
- Alternative: User inputs special characters in a numeric field. The form does not submit, and an error message is displayed. The system performs input validation, identifies the invalid characters, and prompts the user to correct the input.

2) Managing High Traffic Load

Actors: User

Precondition: The system is configured to handle varying levels of traffic.

Parameters: Traffic load levels (normal, high).

Postcondition:

1. The system effectively manages high traffic loads.
2. Continuous availability is ensured.

Actions:

1. A promotional campaign causes a surge in user activity on the website.
2. The system automatically scales resources or employs load balancing.
3. The system continues to function smoothly without crashing.
4. Users experience a responsive application.

Acceptance Tests:

- **Positive Case:** The system experiences normal traffic. The system operates smoothly. The system handles the traffic without any performance issues.
- **Negative Case:** The system experiences a sudden spike in traffic without sufficient resources. The system becomes slow or unresponsive. The system employs load balancing and auto-scaling to manage the traffic spike, ensuring it remains responsive.
- **Alternative Case:** The system experiences a moderate increase in traffic. The system operates smoothly with slightly increased resource usage. The system adjusts resources dynamically to handle the moderate increase in traffic efficiently.

3) Handling Database Connection Failure

Actors: User

Precondition: The system has a caching mechanism and error-handling logic for database connection failures.

Parameters: Database connection status (connected, failed).

Postcondition:

1. Users are informed about the issue.
2. Essential information remains accessible.
3. The system remains operational.

Actions:

1. A network issue or database server downtime disrupts the connection.
2. The system detects the connection failure.
3. The system switches to a read-only mode, providing cached data where possible.
4. The system displays a message to the user about the temporary issue.

Acceptance Tests:

- **Positive Case:** Database connection is stable. The system operates normally. The system continues to interact with the database without issues.

- **Negative Case:** Database connection fails. The system switches to read-only mode and displays an error message. The system detects the failure, uses cached data, and informs the user of the temporary issue.
- **Alternative Case:** Intermittent database connectivity. The system handles occasional read/write errors gracefully. The system retries the failed operations and uses cached data when necessary, keeping the user informed.

4) Handling Unexpected API Response

Actors: System

Precondition: The system integrates with external APIs and includes error-handling for unexpected responses.

Parameters: API response status (normal, unexpected).

Postcondition:

1. The system remains stable.
2. Users receive appropriate feedback.

Actions:

1. The system makes a request to an external API.
2. The API returns an unexpected response or times out.
3. The system handles the unexpected response gracefully.
4. The system logs the issue and retries the request or uses fallback data.
5. The system informs the user of the temporary problem.

Acceptance Tests:

- **Positive Case:** The API returns a valid response. The system processes the response correctly. The system uses the API response as expected without issues.
- **Negative Case:** The API returns an unexpected response. The system logs the issue, retries the request, or uses fallback data. The system detects the unexpected response, logs it, and retries or uses fallback data, informing the user of the temporary issue.
- **Alternative Case:** The API times out. The system logs the timeout and retries the request. The system handles the timeout by logging it and retrying the request after a short delay, informing the user if the issue persists.

5) Preventing Unauthorized Access Attempt

Actors: User, System

Precondition: The system has role-based access control and logging mechanisms in place.

Parameters: Access control rules (authorized, unauthorized).

Postcondition:

1. Unauthorized access attempts are blocked.
2. The system remains stable.

Actions:

1. An unauthorized access attempt is detected by the system.
2. The system prevents the unauthorized action.
3. The system logs the attempt.
4. The system displays a message to the user about insufficient permissions.

Acceptance Tests:

- **Positive Case:** Authorized user attempts a permitted action. The action is allowed and executed. The system verifies permissions and allows the action.
- **Negative Case:** Unauthorized user attempts a restricted action. The action is blocked, and the user is informed. The system prevents the action, logs the attempt, and displays an insufficient permissions message to the user.
- **Alternative Case:** A user with expired permissions attempts an action. The action is blocked, and the user is informed about the expired permissions. The system checks the permissions validity, blocks the action, logs the attempt, and informs the user about the need to renew permissions.

6) Handling Data Integrity Issues

Actors: User, System

Precondition: The system includes mechanisms for data validation and transaction management.

Parameters: Data integrity status (valid, corrupted).

Postcondition:

1. Data integrity is maintained.
2. The system remains operational.

Actions:

1. The system detects data corruption or integrity issues during a transaction.
2. The system rolls back the transaction.

3. The system logs the issue.
4. The system informs the user of the error and prompts for retry or corrective action.

Acceptance Tests:

- **Positive Case:** Data is valid. The transaction is completed successfully. The system processes the transaction without issues.
- **Negative Case:** Data corruption is detected. The transaction is rolled back, and the user is informed. The system detects the corruption, rolls back the transaction, logs the issue, and informs the user to retry or correct the data.
- **Alternative Case:** Partial data corruption detected. The system handles the corrupted part, rolls back the transaction, and prompts the user. The system detects the partial corruption, rolls back the transaction, logs the issue, and informs the user to correct the data before retrying.