Team Name: TeamJoshbutnotJosh

Team Members: Dobson Dunavant, Joshua Uys, Leonardo Georgeto, Robbie

Secure Ticketing System (PouncePass) Elaboration Document

Use Cases

Purpose

The College of Charleston requests the development of a secure application to facilitate the selling of event tickets, allowing users to search and purchase tickets to various events.

Use Case Specification

1. User Authentication

1.1 Secure Login

- **Use case name**: Secure Login
- **Scope**: Secure Ticketing System
- **Level**: User Goal
- **Primary Actor**: User
- **Stakeholders and Interests**:
    - **User**: Wants a secure and reliable method to login to the system.
    - **System Administrator**: Requires the system to maintain security and reliability.
- **Preconditions**: The user is registered and has valid login credentials.
- **Success Guarantee**: User is successfully logged in to the system.
- **Main Success Scenario**:
        - User navigates to the login page.
        - User enters their unique username and password.
        - System validates the credentials and grants access to the user.

- **Extensions**:
    - 3a. System displays an error message for incorrect credentials.
- **Special Requirements**: None.
- **Technology and Data Variations List**:
    - Utilization of secure and encrypted channels for data transmission.
- **Frequency of Occurrence**: Very High
- **Miscellaneous**: None

1.2 Unique Username

- **Use case name**: Unique Username Registration
- **Scope**: Secure Ticketing System
- **Level**: User Goal
- **Primary Actor**: User
- **Stakeholders and Interests**:
    - **User**: Wants to secure a unique identifier within the system.
    - **System Administrator**: Needs to maintain a database of unique identifiers for security and data integrity.
- **Preconditions**: The user is on the registration page, ready to create a new account.
- **Success Guarantee**: User successfully registers with a unique username.
- **Main Success Scenario**:
        - User inputs a desired username during the registration process.
        - System verifies the uniqueness of the username in the database.
        - User successfully registers with a unique username.
- **Extensions**:
    - 2a. The system prompts the user to choose another username if the chosen one is already taken.
- **Special Requirements**: None.
- **Technology and Data Variations List**:
    - Utilization of a robust database system to store and verify usernames.
- **Frequency of Occurrence**: High.
- **Miscellaneous**: Ensuring the privacy and security of user data is a top priority.

1.3 Secure Password

- **Use case name**: Secure Password Creation
- **Scope**: Secure Ticketing System
- **Level**: User Goal
- **Primary Actor**: User
- **Stakeholders and Interests**:
    - **User**: Wants to create a secure password that protects their account and personal information.
    - **System Administrator**: Requires users to create strong passwords to ensure system security and data protection.
- **Preconditions**: The user is on the registration page, creating a new account.
- **Success Guarantee**: User sets up a secure password adhering to the set criteria.
- **Main Success Scenario**:
        - User begins to create a password during the registration process.
        - User creates a password adhering to the criteria: a minimum of 15 characters, including lowercase and uppercase letters, numbers, and special characters.
        - System verifies the strength and criteria of the password.
        - User successfully creates a secure password.
- **Extensions**:
    - 2a. System prompts the user to create a password fulfilling the criteria if the entered password does not meet the requirements.
- **Special Requirements**: None.
- **Technology and Data Variations List**:
    - Implementation of encryption algorithms to store passwords securely.
- **Frequency of Occurrence**: High.
- **Miscellaneous**: Ensuring strong encryption and hashing algorithms are in place for password security.

2. Functions of System

2.1 Search for Events

- **Use case name**: Event Search Functionality
- **Scope**: Secure Ticketing System
- **Level**: User Goal
- **Primary Actor**: User

- **Stakeholders and Interests**:
    - **User**: Wants an efficient and user-friendly method to search for events.
    - **Event Organizer**: Interested in maximizing the visibility and attendance of events.
- **Preconditions**: The user is logged into the system.
- **Success Guarantee**: User successfully finds and views details of desired events.
- **Main Success Scenario**:
        - User navigates to the event search section.
        - User enters specific search criteria to find events.
        - System displays a list of events matching the criteria.
        - User views detailed information of selected events.
- **Extensions**:
    - 3a. System displays a message if no events match the search criteria.
- **Special Requirements**: None.
- **Technology and Data Variations List**:
    - Integration with a dynamic database to update and display event information in real-time.
- **Frequency of Occurrence**: High.
- **Miscellaneous**: Ensuring the accuracy and up-to-datedness of event information is critical.

2.2 Purchase Tickets

- **Use case name**: Ticket Purchase Process
- **Scope**: Secure Ticketing System
- **Level**: User Goal
- **Primary Actor**: User
- **Stakeholders and Interests**:
    - **User**: Aims to purchase tickets for events conveniently and securely.
    - **Event Organizer**: Interested in selling tickets efficiently and monitoring ticket sales dynamically.
    - **System Administrator**: Ensures smooth and secure transactions, maintaining the integrity of ticket sales data.
- **Preconditions**: The user has found an event they are interested in and are logged into the system.
- **Success Guarantee**: User successfully purchases tickets and receives a confirmation message.
- **Main Success Scenario**:

        - User selects the desired event.
        - User chooses the number of tickets to purchase.
        - User proceeds to the payment section.
        - User completes the mock payment process.
        - System verifies the payment and displays a success message.
        - System updates the ticket inventory accordingly.
- **Extensions**:
    - 4a. System displays an error message if there is an issue during the payment process.
    - 6a. System notifies the user if the tickets are sold out during the process.
- **Special Requirements**: Integration with a reliable mock payment gateway.
- **Technology and Data Variations List**:
    - Integration with a dynamic database to update and monitor ticket inventory in real-time.
- **Frequency of Occurrence**: Moderate.
- **Miscellaneous**: Maintaining a reliable and secure payment process is crucial, along with real-time updates to ticket inventory.

2.3 Mock Process Payment

- **Use case name**: Mock Process Payment
- **Scope**: Secure Ticketing System
- **Level**: User Goal
- **Primary Actor**: User
- **Stakeholders and Interests**:
    - **User**: Expects a smooth and secure payment process.
    - **System Administrator**: Ensures the system simulates a secure and smooth payment process without real transaction taking place.
- **Preconditions**: The user has selected tickets for purchase and is in the payment section.
- **Success Guarantee**: User successfully completes the mock payment process.
- **Main Success Scenario**:
        - User enters payment details in the provided fields.
        - System validates the entered payment details.
        - System processes the mock payment successfully.
- **Extensions**: 3a. System shows an error message if the mock payment process fails.

- **Special Requirements**: Integration with a mock payment gateway to simulate the payment process.
- **Technology and Data Variations List**: N/A
- **Frequency of Occurrence**: High
- **Miscellaneous**: The mock payment gateway should resemble a real payment gateway to give users a realistic experience.

2.4 Display Message of Payment Success

- **Use case name**: Display Message of Payment Success
- **Scope**: Secure Ticketing System
- **Level**: User Goal
- **Primary Actor**: System
- **Stakeholders and Interests**:
    - **User**: Expects a clear confirmation that the payment was successful.
    - **System Administrator**: Needs to ensure the system accurately communicates the status of the payment to the user.
- **Preconditions**: The mock payment process has been successfully completed.
- **Success Guarantee**: User sees a clear confirmation message indicating successful payment.
- **Main Success Scenario**:
    - System verifies the success of the mock payment process.
    - System displays a success message to the user.
- **Extensions**: N/A
- **Special Requirements**: Clear and user-friendly interface to convey success messages.
- **Technology and Data Variations List**: N/A
- **Frequency of Occurrence**: High
- **Miscellaneous**: The message should be concise and clearly indicate the success of the transaction.

2.5 Update Ticket Inventory

- **Use case name**: Update Ticket Inventory
- **Scope**: Secure Ticketing System
- **Level**: System Operation
- **Primary Actor**: System
- **Stakeholders and Interests**:
    - **Event Organizer**: Requires an accurate and real-time update of the ticket inventory.

    - **System Administrator**: Needs to maintain accurate data on ticket inventory for management and reporting.

- **Preconditions**: Tickets have been successfully purchased through the system.
- **Success Guarantee**: The ticket inventory is updated accurately to reflect the new status.
- **Main Success Scenario**:
    - System acknowledges the successful transaction.
    - System updates the ticket inventory to reflect the purchase.
    - System maintains an accurate count of available tickets for the event.
- **Extensions**:
    - 3a. System sends a notification to the event organizer if the ticket inventory reaches a low threshold.
- **Special Requirements**: Integration with a robust database system to maintain and update ticket inventory in real-time.
- **Technology and Data Variations List**: N/A
- **Frequency of Occurrence**: Moderate
- **Miscellaneous**: The system should be able to handle high traffic and multiple concurrent updates to the ticket inventory without errors or delays.

3. Stretch Goals

3.1 Google Calendar Integration

(To follow the same template as above, for Google Calendar Integration)

3.2 Seat Selection

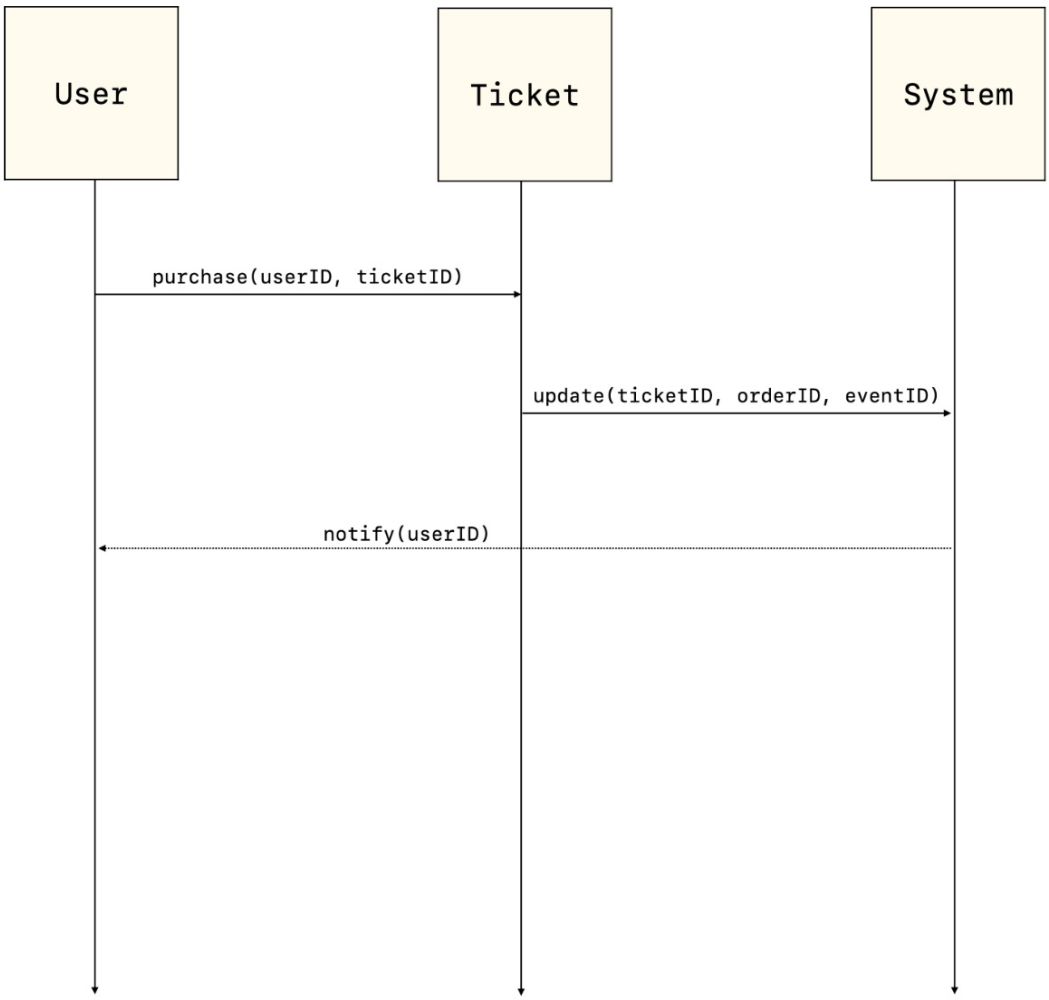
(To follow the same template as above, for Seat Selection)

Conclusion

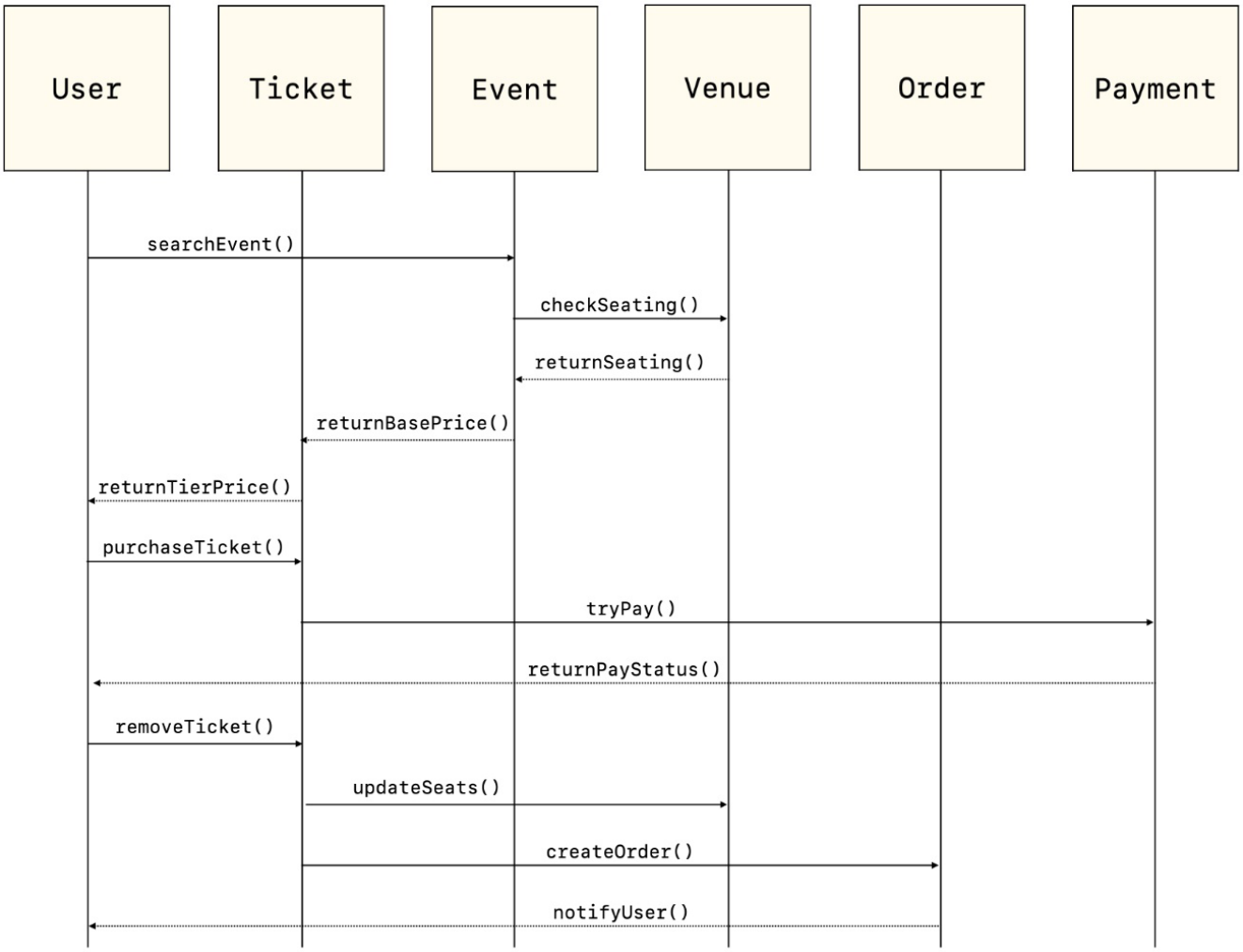
This document outlines the primary use cases for the Secure Ticketing System to be developed for the College of Charleston. Each use case provides a structured overview of the different functionalities that will be implemented, fulfilling the stakeholder interests and requirements.

System Sequence Diagram

Higher Level View:



Lower Level View:



Operation Contracts

Name	Content
Operation	successfulPurchaseTicket(int: ticketID, int: userID, object: paymentDetails)
Cross References	• **ticketID**: The unique identifier of the ticket to be purchased. • **userID**: The unique identifier of the user making the purchase. • **paymentDetails**: An object containing payment information such as card number, expiration date, etc.
Preconditions	1. The `ticketID` corresponds to a valid and existing ticket in the system. 2. The `userID` corresponds to a valid and registered user in the system. 3. The ticket associated with the `ticketID` is available for purchase (not already sold or reserved). 4. The `paymentDetails` provided are valid and the payment method has sufficient funds/credit to cover the ticket price. 5. The user has agreed to any terms and conditions required for the ticket purchase.
Postconditions	1. The ticket status is updated to "sold" or "reserved". 2. The ticket's ownership is transferred to the user associated with the `userID`. 3. The payment is processed successfully, and the payment details are recorded in the system. 4. A purchase confirmation is sent to the user (e.g., via email or in-app notification). 5. The user's purchase history in the system is updated to include the details of this transaction. 6. The inventory of available tickets in the system is reduced by one, or that specific ticketID is updated as sold
Operation	successfullogin(int: userID, string: userPassword)
Cross References	• **userID:** The username provided by the user to login • **userPassword:** The password created by the user to login.
Preconditions	1. The `userID` corresponds to a valid and registered user in the system. 2. The `userPassword` corresponds to the password associated with the given `userID`.
Postconditions	1. If login credentials are valid the user gains access to their account. 2. If unsuccessful an error message is displayed
Operation	userNameRegistration(int: userID, string: userPassword, string: name, string: email)
Cross References	• **userID**: The unique identifier of the user. • **userPassword**: password for the given user. • **name:** The full name of the user. • **email:** email provided by the user.
Preconditions	1. The user registration process is initiated by a new user. 2. The system is accessible and operational.

Aa Name	Content
Postconditions	1. If successful, the user's account is created. The user can login 2. If unsuccessful, an error message is displayed. Registration fails
Operation	createPassword(int: userID, string: userPassword)
Cross References	• **userID:** The username provided by the user to login • **userPassword:** The password provided by the user to login.
Preconditions	1. The `userPassword` is a length of 8-16 characters, contains two symbols, and one lower and uppercase character.
Postconditions	1. The `userPassword` is generated for the user.
Operation	searchEvents(object: events, dateRange: date, string: location, string: category)
Preconditions	1. Database includes events 2. User has permission to search `events` 3. Search criteria is defined for users.
Postconditions	The operation provides the user with a list of appropriate events
Operation	processPayment(int: paymentAmount, paymentMethod: string)
Cross References	• **paymentAmount:** The amount owed • **paymentMethod:** Paypal, Visa, etc.
Preconditions	1. The user initiating the payment has a valid account or payment method registered. 2. Payment details, including the amount and recipient, are provided.
Postconditions	1. If successful, payment is processed and confirmed 2. if unsuccessful an error message is displayed,