LPTS

Los Portales Theatre System

https://github.com/Streetplayr1/Los-Portales-Theatre-System

Trey Allen

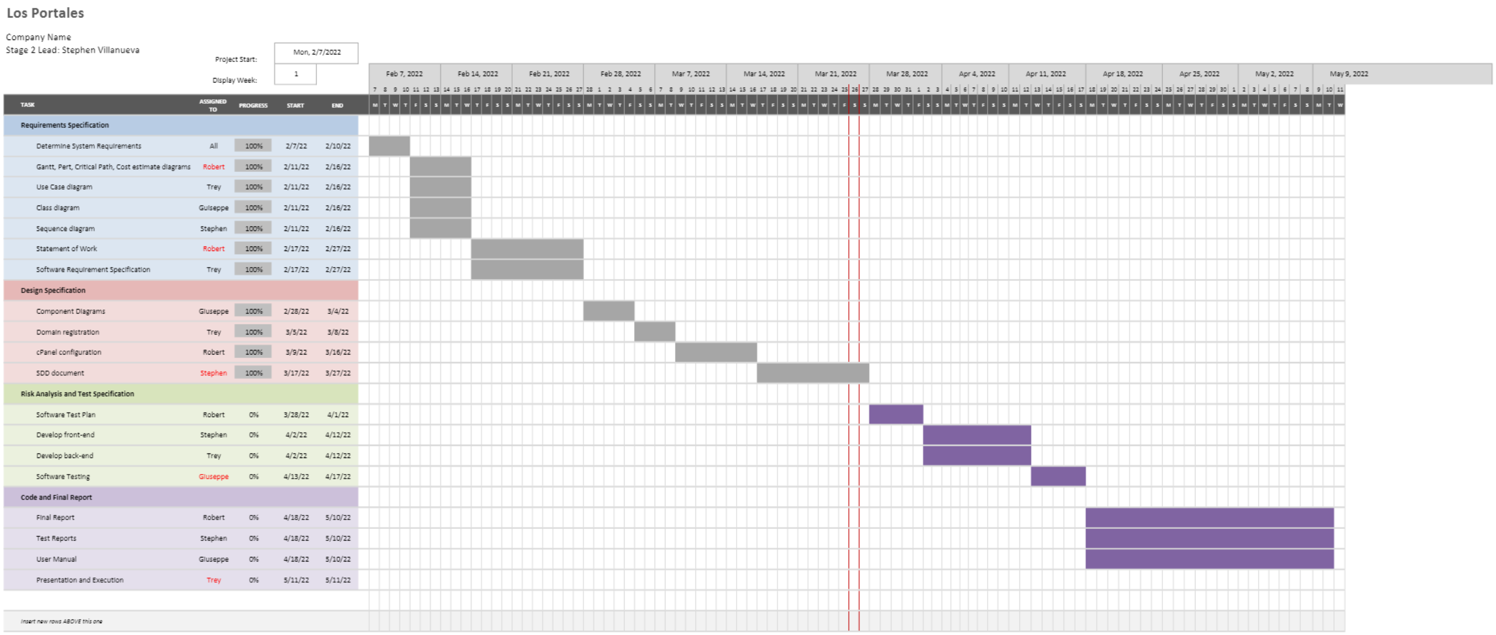
Stephen Villanueva

Robert Leal

Giuseppe Scalise

Table of Contents

Individual Contributions Breakdown



Los Portales

Company Name
Stage 2 Lead: Stephen Villanueva

Project Start: Mon, 2/7/2022
Display Week: 1

TASK	ASSIGNED TO	PROGRESS	START	END
Requirements Specification				
Determine System Requirements	All	100%	2/7/22	2/10/22
Gantt, Pert, Critical Path, Cost estimate diagrams	Robert	100%	2/11/22	2/16/22
Use Case diagram	Trey	100%	2/11/22	2/16/22
Class diagram	Guiseppe	100%	2/11/22	2/16/22
Sequence diagram	Stephen	100%	2/11/22	2/16/22
Statement of Work	Robert	100%	2/17/22	2/27/22
Software Requirement Specification	Trey	100%	2/17/22	2/27/22
Design Specification				
Component Diagrams	Giuseppe	100%	2/28/22	3/4/22
Domain registration	Trey	100%	3/5/22	3/8/22
cPanel configuration	Robert	100%	3/9/22	3/16/22
SDD document	Stephen	100%	3/17/22	3/27/22
Risk Analysis and Test Specification				
Software Test Plan	Robert	0%	3/28/22	4/1/22
Develop front-end	Stephen	0%	4/2/22	4/12/22
Develop back-end	Trey	0%	4/2/22	4/12/22
Software Testing	Giuseppe	0%	4/13/22	4/17/22
Code and Final Report				
Final Report	Robert	0%	4/18/22	5/10/22
Test Reports	Stephen	0%	4/18/22	5/10/22
User Manual	Giuseppe	0%	4/18/22	5/10/22
Presentation and Execution	Trey	0%	5/11/22	5/11/22

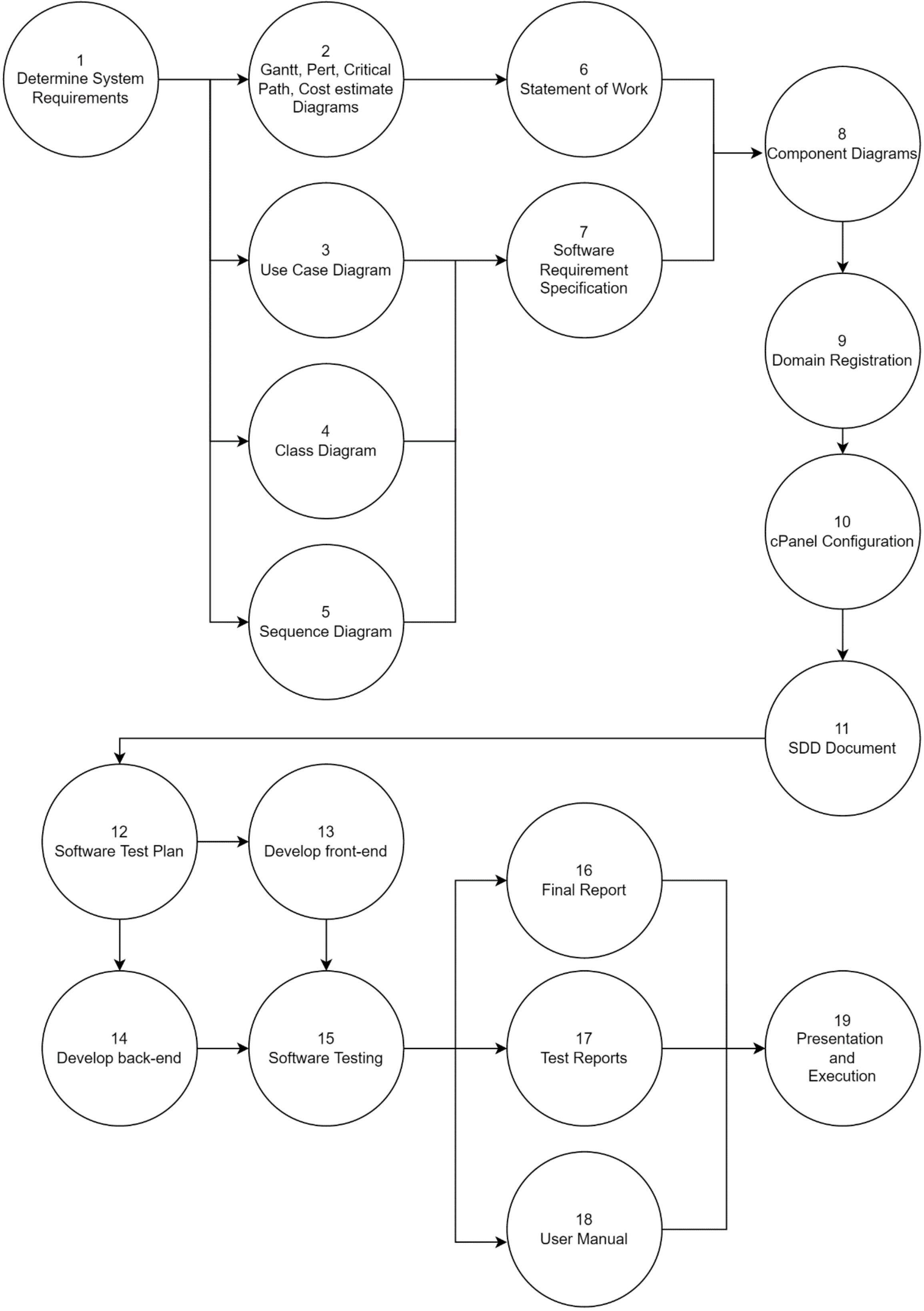
Insert new rows ABOVE this one

Summary of Changes

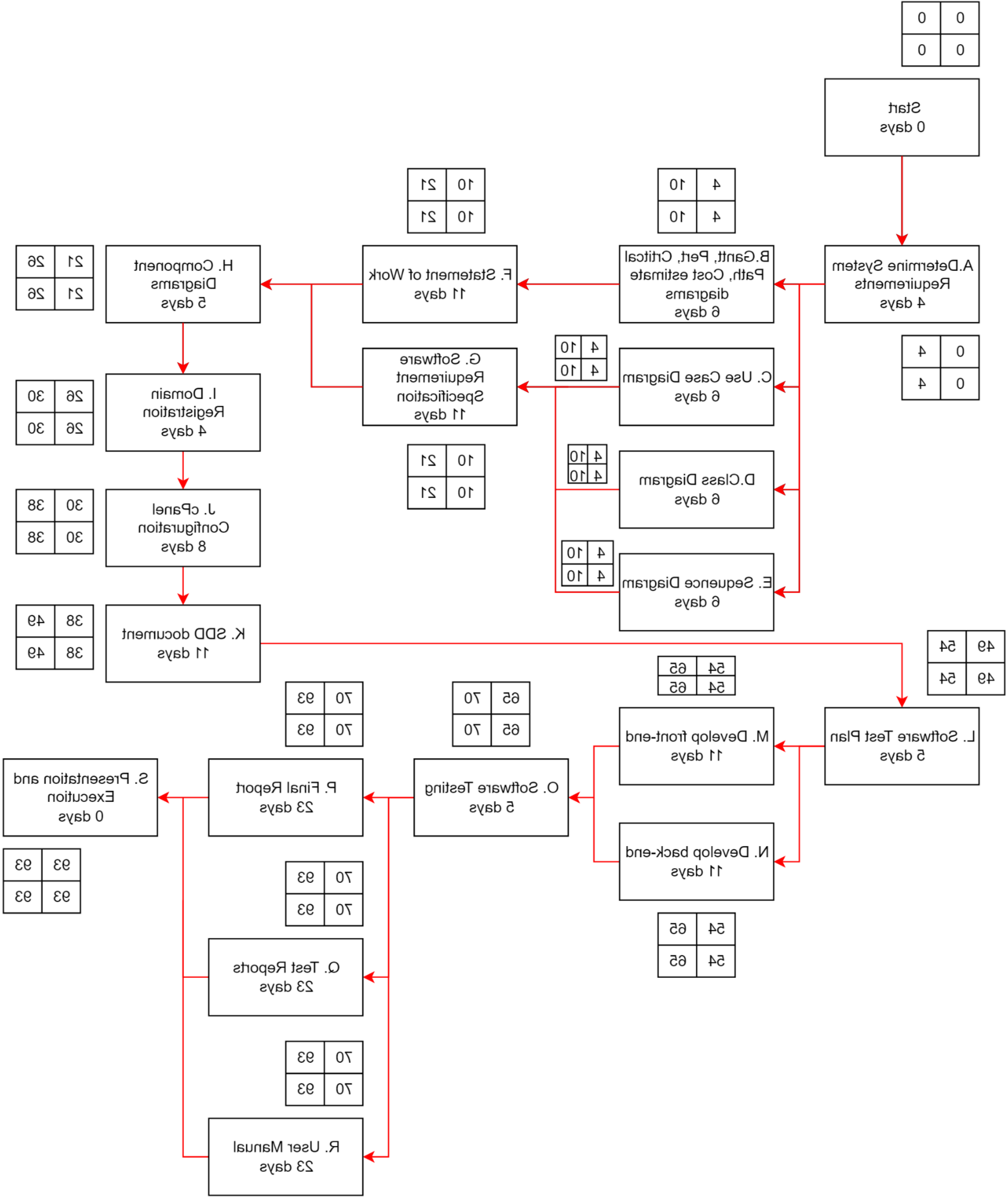
- Added use case tables
- Added security features
- Added functional and nonfunctional requirements
- Modified use-case table hierarchy
- Adjusted diagrams
- Modified Table of Contents for Matrix
- Assigned Requirement ID and Technical Requirement ID for Matrix Requirements
- Updated UML Diagram to match system
- Fulfilled tasks missing in Pert Diagram
- Added actor
- Implemented additional non-functional requirements
- Completed requirements matrix
- Added Elon Musk as a partner to the project

1 Customer Statement of Requirements

1.1 Pert Diagram



1.2 Critical Path Diagram



1.3 Scope, Tasks, and Cost Estimation

Scope:

Los Portales requires an app in which tickets for plays can be sold and sales can be managed. The project is meant to ease ticket buying for customers and simplify management of sales for administration of theater. This document will contain graphs and charts related to the progress of the project.

The main objective of the final product is for customers to be able to choose seating and purchase tickets for those seats. Theater administration shall be able to add plays, showtimes, and pricing for seating.

The project will be small to moderate in terms of breadth and shall have few limitations.

Contractor shall work on deciding software requirements which may take up to approximately 4 days. Design requirements will then be decided and may take up to 28 days. A test plan will then be made and may take up to 5 days. The Pert diagram shown in Figure 1 shows a flow of the tasks that are being planned. The Critical Path diagram shown in Figure 2 shows duration for each task.

Tasks:

The security for this system software includes role-based access and encrypted credential information. Furthermore, the login fields will be sanitized to prevent malicious code from being run on the server.

There are no foreseeable travel requirements.

There are no foreseeable unique requirements.

Performance will be at the Contractor’s facility.

Period estimated to complete project starts 02-07-2022 and ends 05-11-2022

Estimated Cost:

Estimated cost of this software is approximately $56,000. Please refer to SOW diagram for breakdown.

2 Glossary of Terms

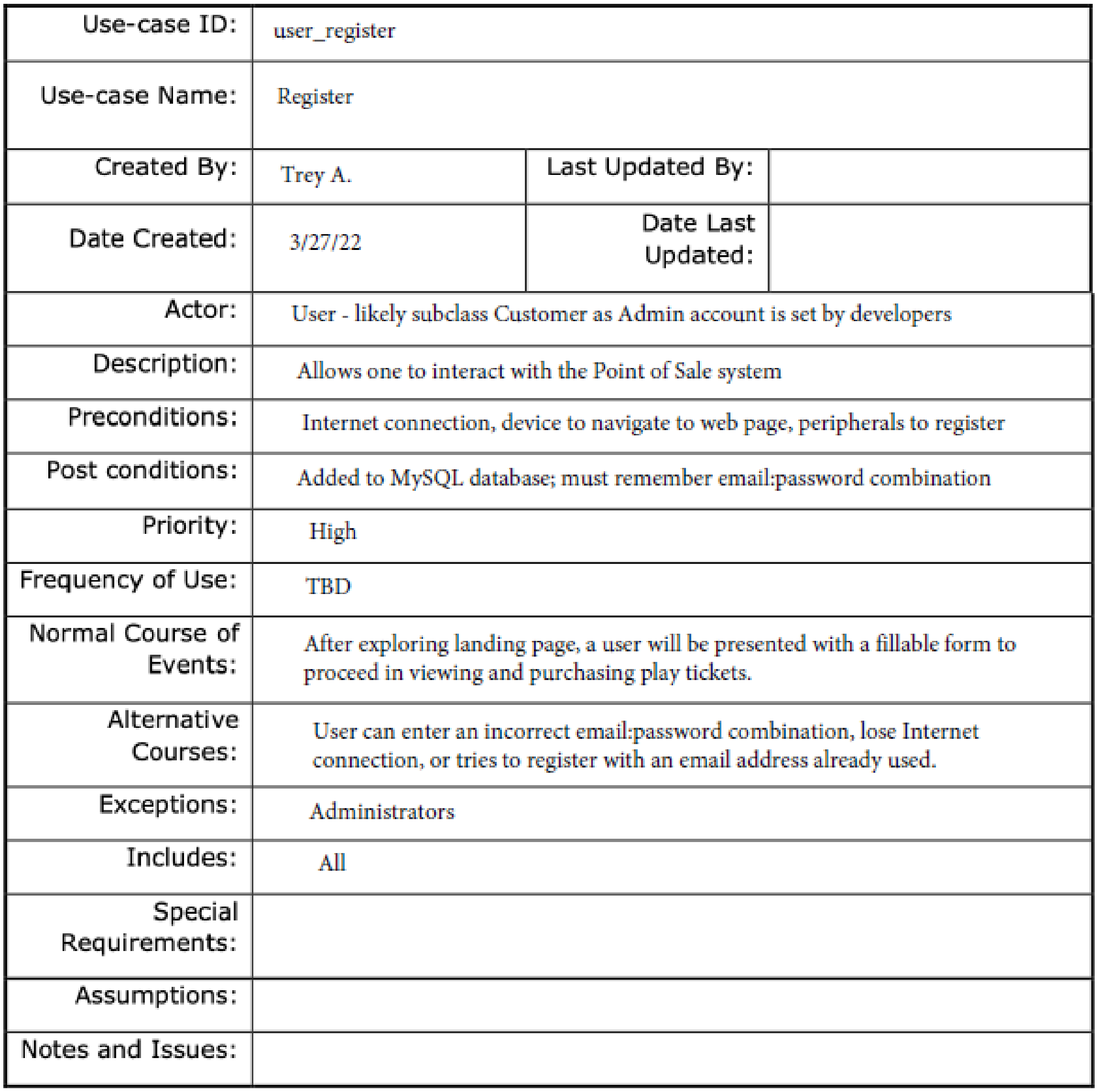
Term	Definition
Administrator	A person responsible for running the system software.
Business Continuity	Plan to deal with difficult situations; continuing to function with as little disruption as possible.
Customer Module	Key activities for the registered customer to add and remove seats, add and remove to cart, change from different plays, and purchase tickets.
Database	Collection of all the information monitored by this system.
Developers	A person that is responsible for creating the (working) system.
Field	A cell within a form.
Graphical User Interface	A visual way of interacting with a computer.
Maintenance Module	Key activities for the administrator to create, manage, and delete fields.
Registered Customer	Basic user information stored in the database that grants the user access to Customer Module
Report	Database object that displays or distributes a summary of data.
Software Requirements Specification	A document that completely describes all the functions of a proposed system and the constraints under which it must operate. For example, this document.
Stakeholder	Any person with an interest in the project who is not a developer.
User	Customer or Administrator

3 Software Requirements Specification

3.1 Use Case Breakdown

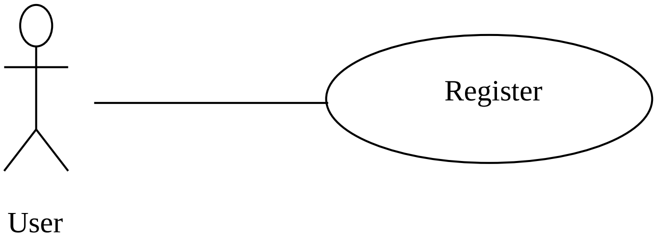
Use case: **Register [T1]**

Table:



Use-case ID:	user_register		
Use-case Name:	Register		
Created By:	Trey A.	Last Updated By:	
Date Created:	3/27/22	Date Last Updated:	
Actor:	User - likely subclass Customer as Admin account is set by developers		
Description:	Allows one to interact with the Point of Sale system		
Preconditions:	Internet connection, device to navigate to web page, peripherals to register		
Post conditions:	Added to MySQL database; must remember email:password combination		
Priority:	High		
Frequency of Use:	TBD		
Normal Course of Events:	After exploring landing page, a user will be presented with a fillable form to proceed in viewing and purchasing play tickets.		
Alternative Courses:	User can enter an incorrect email:password combination, lose Internet connection, or tries to register with an email address already used.		
Exceptions:	Administrators		
Includes:	All		
Special Requirements:			
Assumptions:			
Notes and Issues:			

Diagram:



Brief Description

The User's only use case to access system functionality is to register to the database.

Initial Step-By-Step Description

Before this use case can be initiated, the User must access the web application software.

1. The User chooses the Register button.
2. The system displays fillable fields to the User.

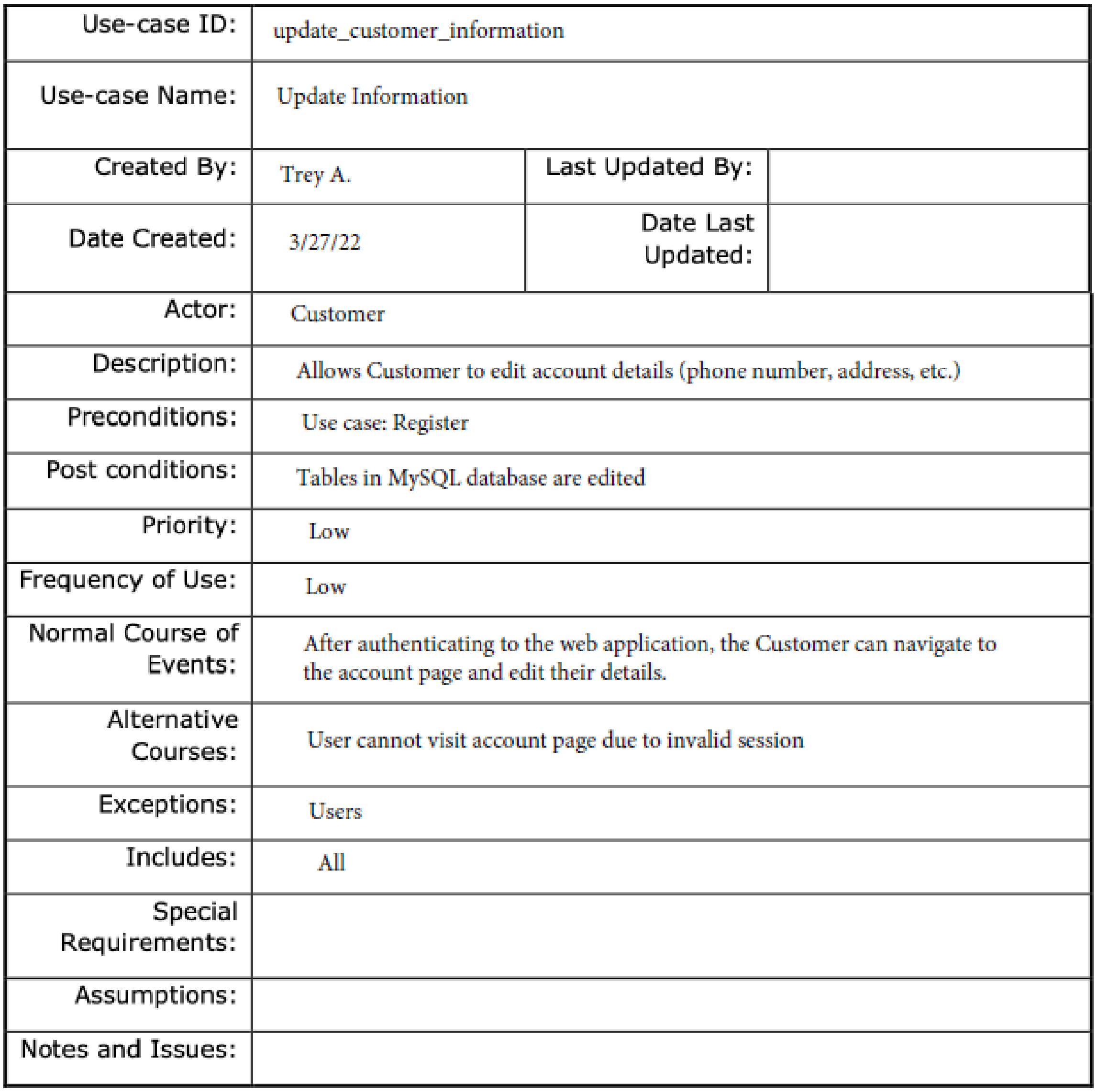
3. The User provides their name, age, address, telephone number, and email address.
4. The system confirms the registration was successful.
5. The User chooses the Login button.
6. The system displays the corresponding Module depending on credentials.

2.2.2 Registered Customer Use Case [A2]

Depending on credentials of the user, this case covers a customer’s role-based access.

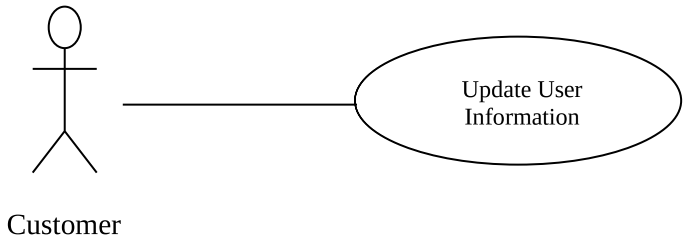
Use case: Update User Information [T2]

Table:



Use-case ID:	update_customer_information		
Use-case Name:	Update Information		
Created By:	Trey A.	Last Updated By:	
Date Created:	3/27/22	Date Last Updated:	
Actor:	Customer		
Description:	Allows Customer to edit account details (phone number, address, etc.)		
Preconditions:	Use case: Register		
Post conditions:	Tables in MySQL database are edited		
Priority:	Low		
Frequency of Use:	Low		
Normal Course of Events:	After authenticating to the web application, the Customer can navigate to the account page and edit their details.		
Alternative Courses:	User cannot visit account page due to invalid session		
Exceptions:	Users		
Includes:	All		
Special Requirements:			
Assumptions:			
Notes and Issues:			

Diagram:



Brief Description

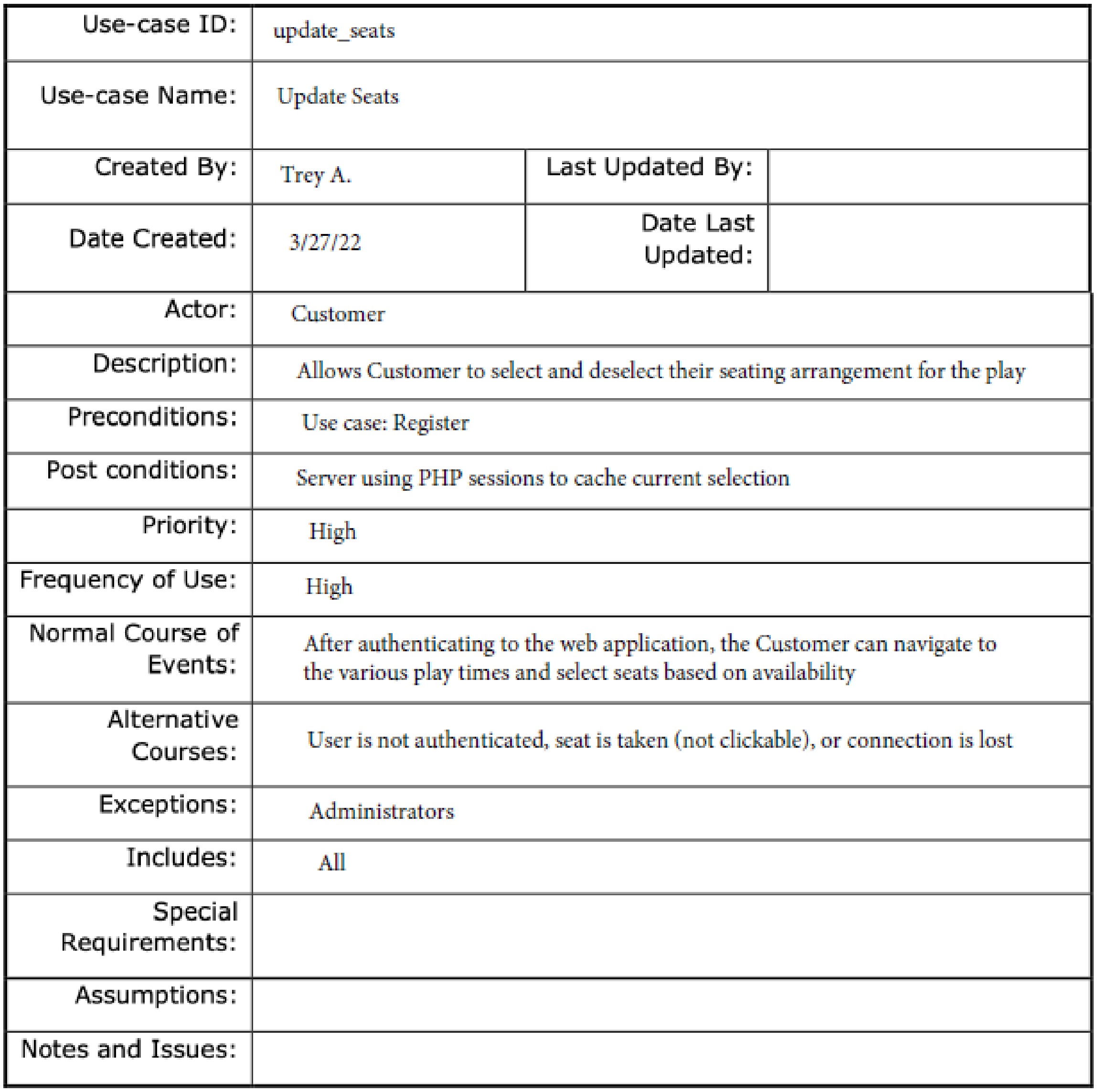
The registered customer can edit the data submitted during the registration process.

Initial Step-By-Step Description

Before this use case can be initiated, the User must access the web application software and login using the credentials created in the Register (2.2.1) use case.

1. The Registered Customer chooses the *Edit Profile* button.
2. The system presents the registered customer with the registration fields.
3. The Registered Customer will change the fields as needed and select *Save.*
4. The System will update the database.

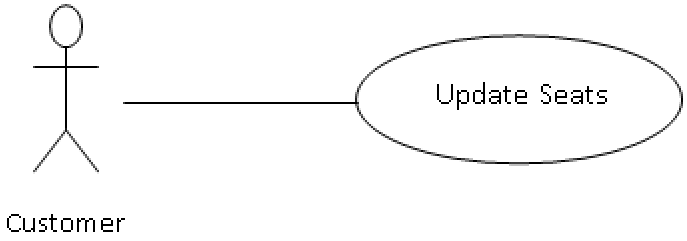
Use case: Update Seats **[T3]**



Use-case ID:	update_seats		
Use-case Name:	Update Seats		
Created By:	Trey A.	Last Updated By:	
Date Created:	3/27/22	Date Last Updated:	
Actor:	Customer		
Description:	Allows Customer to select and deselect their seating arrangement for the play		
Preconditions:	Use case: Register		
Post conditions:	Server using PHP sessions to cache current selection		
Priority:	High		
Frequency of Use:	High		
Normal Course of Events:	After authenticating to the web application, the Customer can navigate to the various play times and select seats based on availability		
Alternative Courses:	User is not authenticated, seat is taken (not clickable), or connection is lost		
Exceptions:	Administrators		
Includes:	All		
Special Requirements:			
Assumptions:			
Notes and Issues:			

Table:

Diagram:

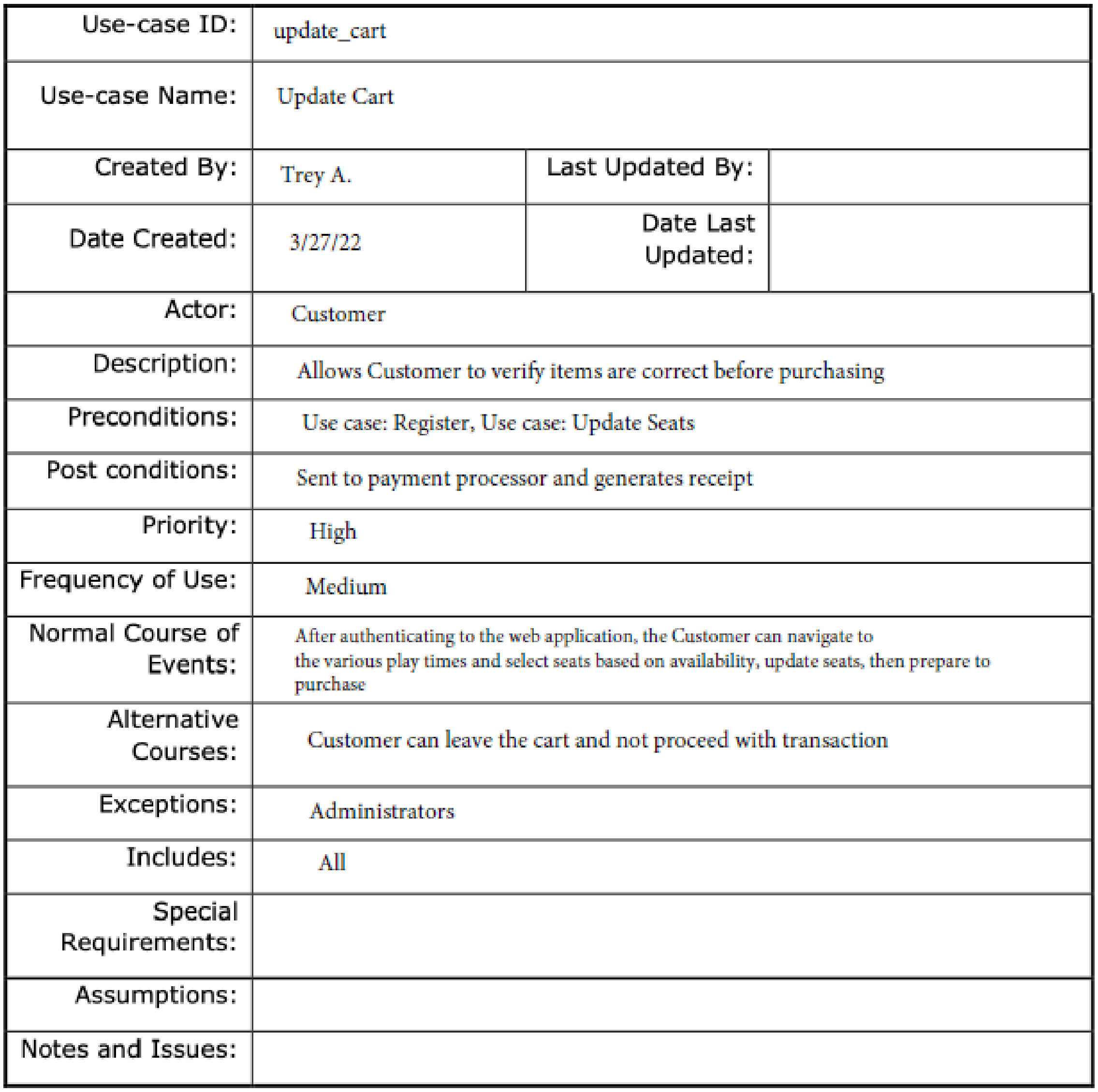


Brief Description
The registered customer can their currently selected seats (add/remove).

Initial Step-By-Step Description
Before this use case can be initiated, the User must access the web application software and login using the credentials created in the Register (2.2.1) use case.

1. The system will present a list of available seats in green for a specific showing.
2. The Registered Customer can select an available seat for any available showing.
3. If the Registered Customer needs a seat from another play, the RC will select *Plays.*
4. The system will present all showings configured by the administrator.
5. The Registered Customer will repeat steps 1-3 until they have fulfilled their desire.

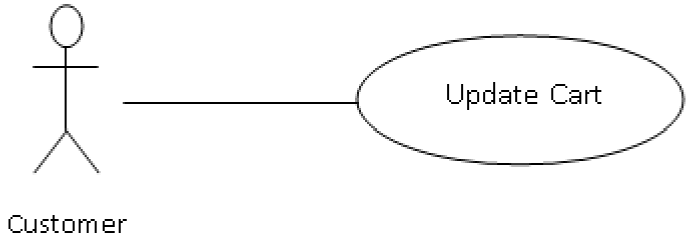
Use case: Update Cart **[T4]**



Use-case ID:	update_cart		
Use-case Name:	Update Cart		
Created By:	Trey A.	Last Updated By:	
Date Created:	3/27/22	Date Last Updated:	
Actor:	Customer		
Description:	Allows Customer to verify items are correct before purchasing		
Preconditions:	Use case: Register, Use case: Update Seats		
Post conditions:	Sent to payment processor and generates receipt		
Priority:	High		
Frequency of Use:	Medium		
Normal Course of Events:	After authenticating to the web application, the Customer can navigate to the various play times and select seats based on availability, update seats, then prepare to purchase		
Alternative Courses:	Customer can leave the cart and not proceed with transaction		
Exceptions:	Administrators		
Includes:	All		
Special Requirements:			
Assumptions:			
Notes and Issues:			

Table:

Diagram:



Brief Description

The registered customer can add or remove their selected seats (Update Seats use case) to their shopping cart to prepare for a purchase.

Initial Step-By-Step Description

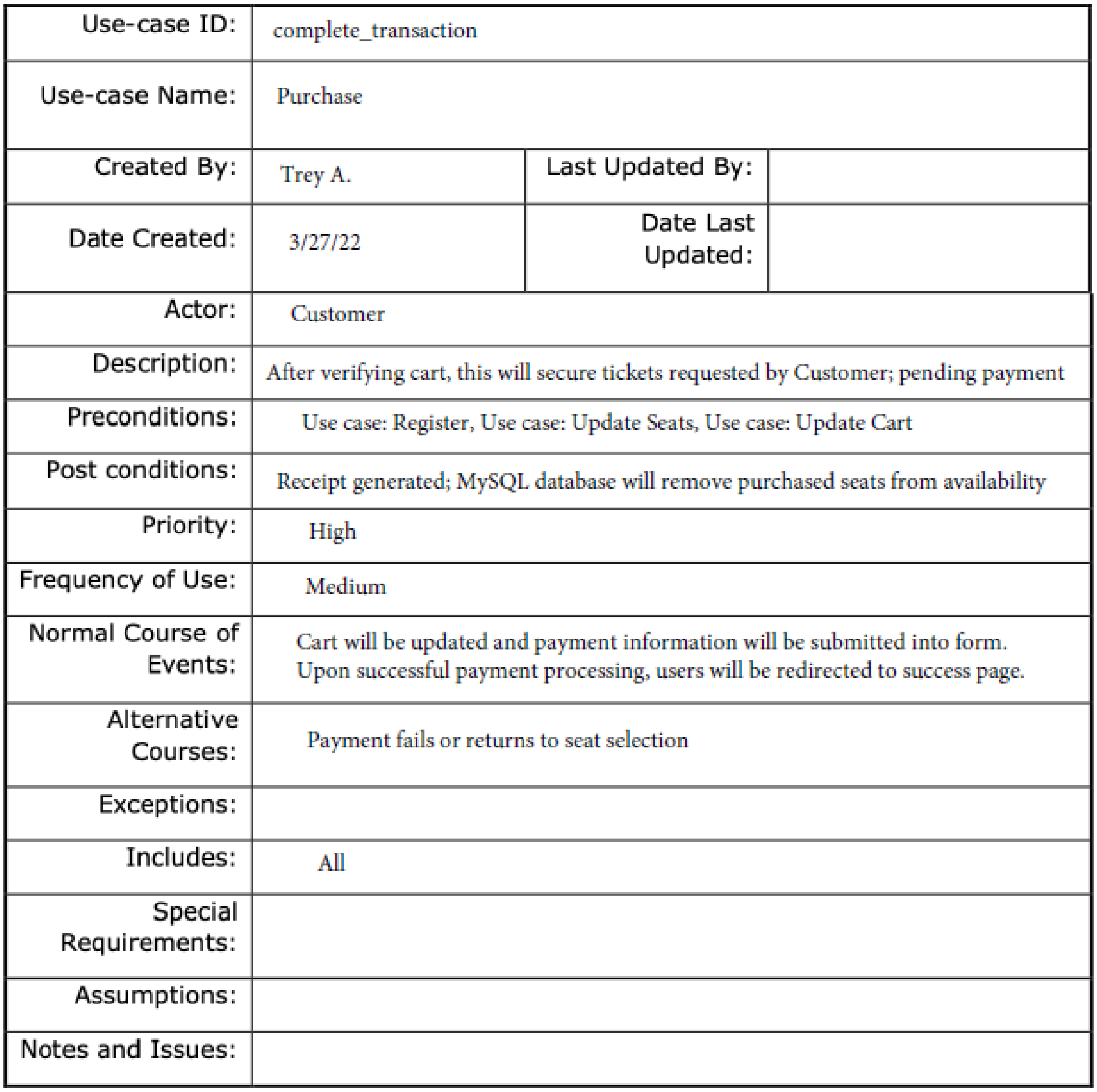
Before this use case can be initiated, the User must access the web application software and login using the credentials created in the Register (2.2.1) use case. Then, have seats selected from the Update Seat use case.

1. The Registered Customer chooses the *Add to Cart* button.
2. The system will store their selected seats into a database.
3. The Registered Customer chooses *View Cart* to see tickets that area ready for purchase.

4. The system will display icons for adding or removing items to the cart and update DB.

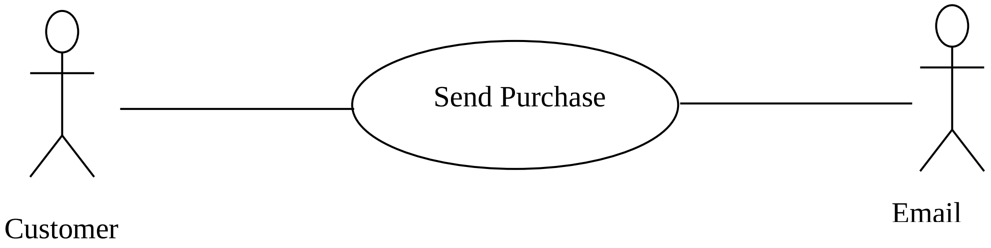
Use case: Send Purchase [T5]

Table:



Use-case ID:	complete_transaction		
Use-case Name:	Purchase		
Created By:	Trey A.	Last Updated By:	
Date Created:	3/27/22	Date Last Updated:	
Actor:	Customer		
Description:	After verifying cart, this will secure tickets requested by Customer; pending payment		
Preconditions:	Use case: Register, Use case: Update Seats, Use case: Update Cart		
Post conditions:	Receipt generated; MySQL database will remove purchased seats from availability		
Priority:	High		
Frequency of Use:	Medium		
Normal Course of Events:	Cart will be updated and payment information will be submitted into form. Upon successful payment processing, users will be redirected to success page.		
Alternative Courses:	Payment fails or returns to seat selection		
Exceptions:			
Includes:	All		
Special Requirements:			
Assumptions:			
Notes and Issues:			

Diagram:



Brief Description

The user can submit their purchase after providing credit card information to the database. The registered costumer will receive an email confirmation of their order.

Initial Step-By-Step Description

Before this use case can be initiated, the User must access the web application software and login using the credentials created in the Register (2.2.1) use case. Then, have seats selected from the Update Seat use case and have items in their cart (Update Cart).

1. The Registered Customer chooses the *View Cart* button.
2. The system will display all tickets added to the RC’s cart and display a *Checkout* option.
3. The Registered Customer chooses *Checkout*.
4. The system will display fields for credit card information and a *Submit* button.
5. The system will display a success or error message and display the output accordingly.

2.2.3 Administrator Use Cases [A3]

The Administrator has the following sets of use cases:

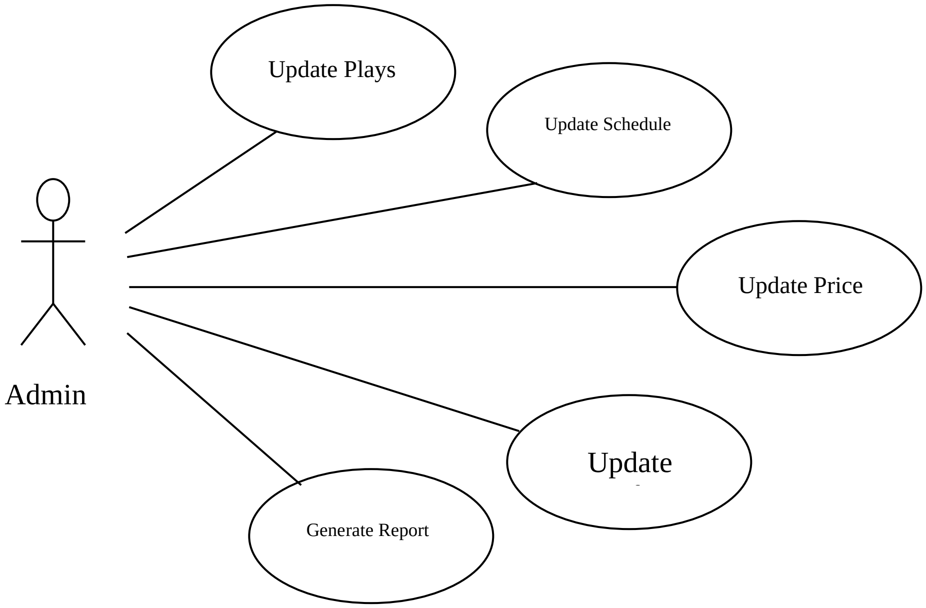


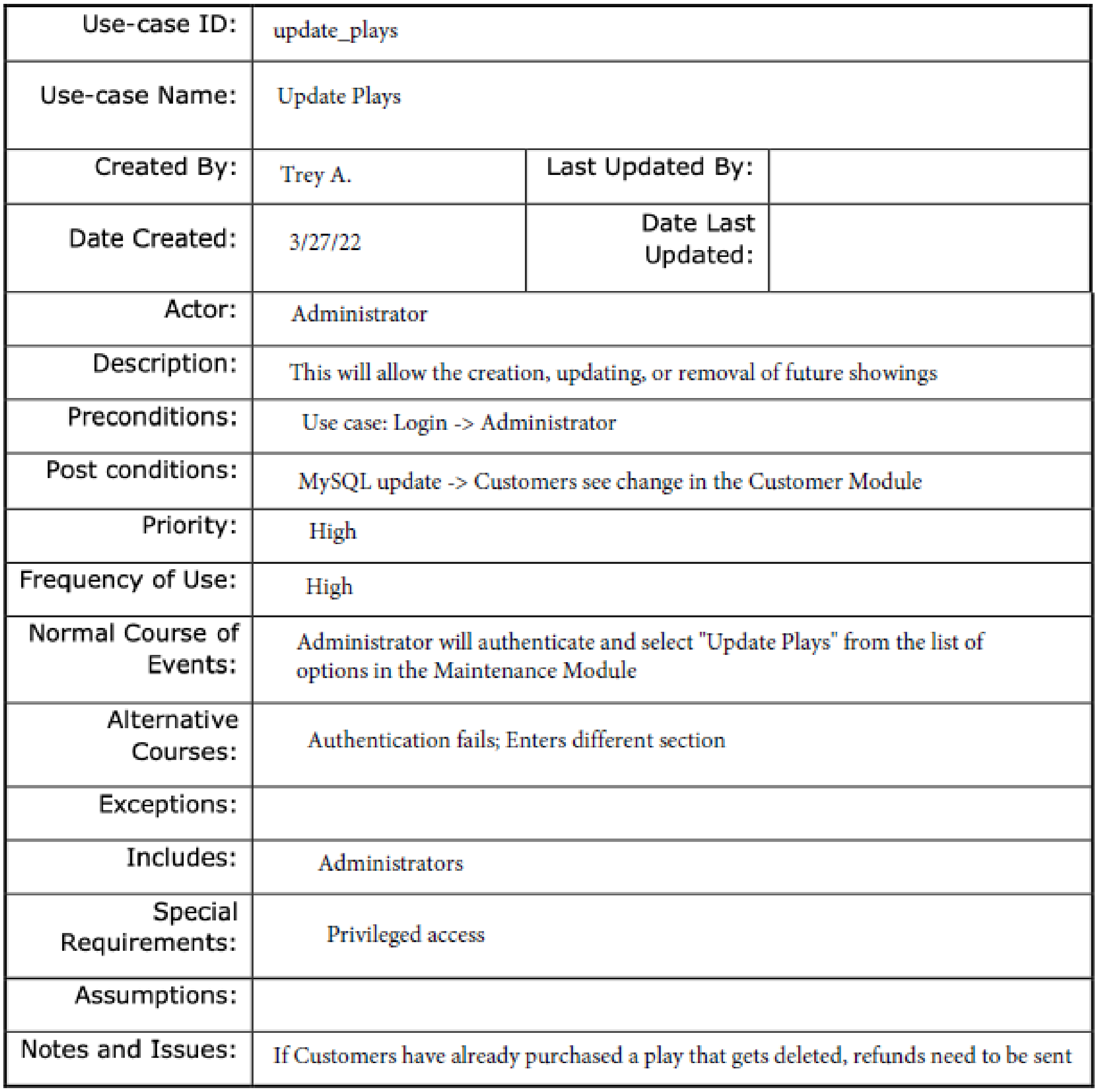
Figure 2 - Administrator Use Cases

Use case: Update Information [T6]

Xref: Section 2.2.2, Update User Information

Note
This use case will work in the same manner as referenced above. The different modules (Customer and Maintenance) will not affect this functionality.

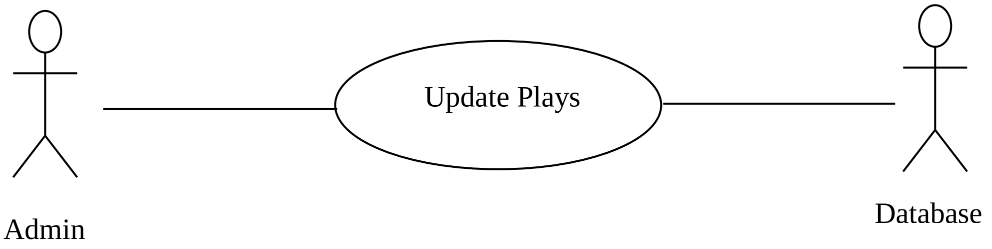
Use case: Update Plays [T7]



Use-case ID:	update_plays		
Use-case Name:	Update Plays		
Created By:	Trey A.	Last Updated By:	
Date Created:	3/27/22	Date Last Updated:	
Actor:	Administrator		
Description:	This will allow the creation, updating, or removal of future showings		
Preconditions:	Use case: Login -> Administrator		
Post conditions:	MySQL update -> Customers see change in the Customer Module		
Priority:	High		
Frequency of Use:	High		
Normal Course of Events:	Administrator will authenticate and select "Update Plays" from the list of options in the Maintenance Module		
Alternative Courses:	Authentication fails; Enters different section		
Exceptions:			
Includes:	Administrators		
Special Requirements:	Privileged access		
Assumptions:			
Notes and Issues:	If Customers have already purchased a play that gets deleted, refunds need to be sent		

Table:

Diagram:



Brief Description

The administrator creates, updates, or removes a play/showing.

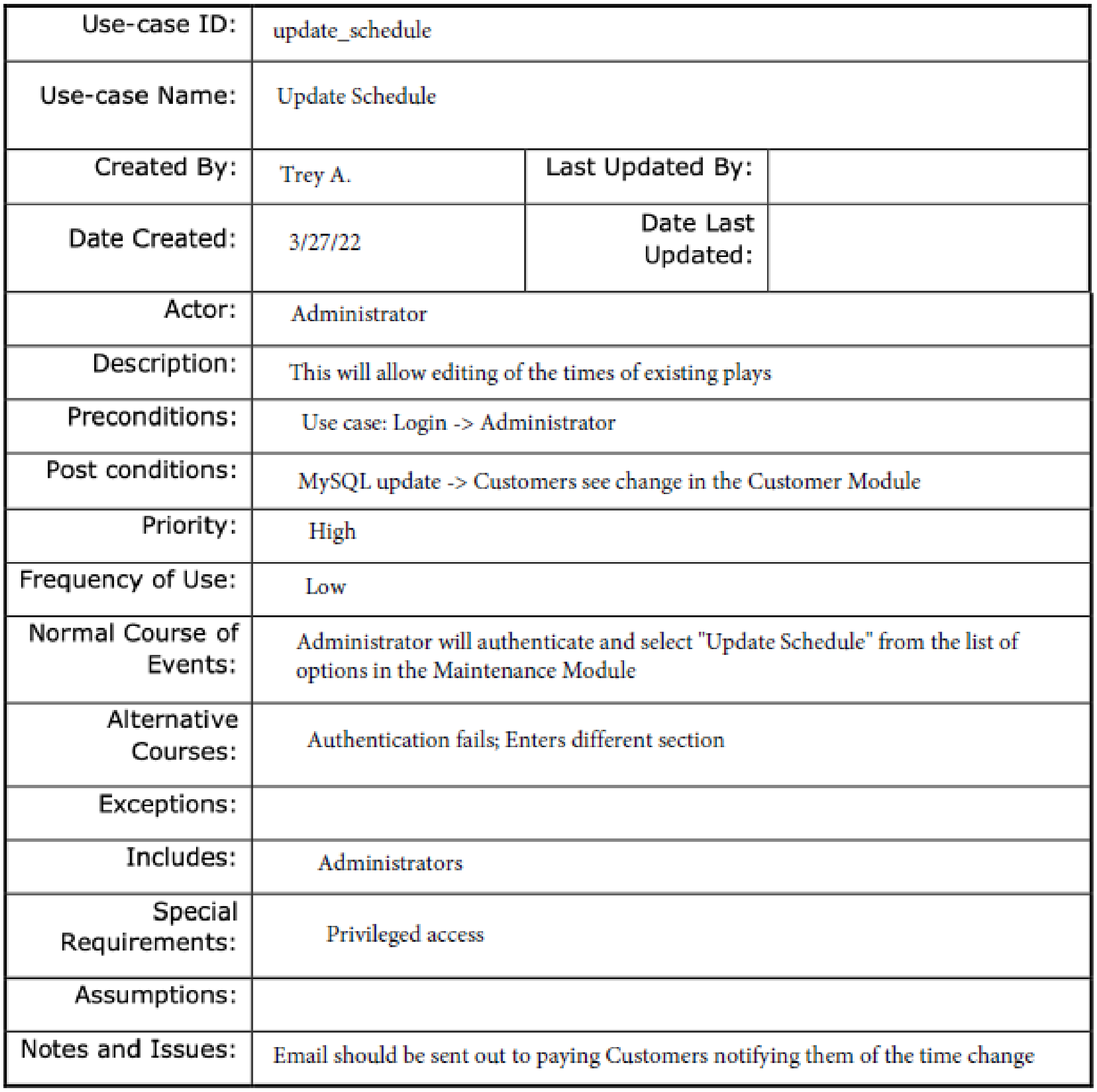
Initial Step-By-Step Description

Before this can be initiated, the administrator must access the web application software and authenticate accordingly. Then, the administrator must navigate to the Play section of the Maintenance module.

1. The Administrator selects *Plays* from the Maintenance module.

2. The system presents a choice of adding, updating, or removing.
3. The Administrator chooses to add, update, or remove.
4. The system links to the existing database.
5. Depending on the choice, administrator will be prompted according to Figure 4.
6. The Editor fills in the information and submits the form.
7. The system verifies the information and updates the database.

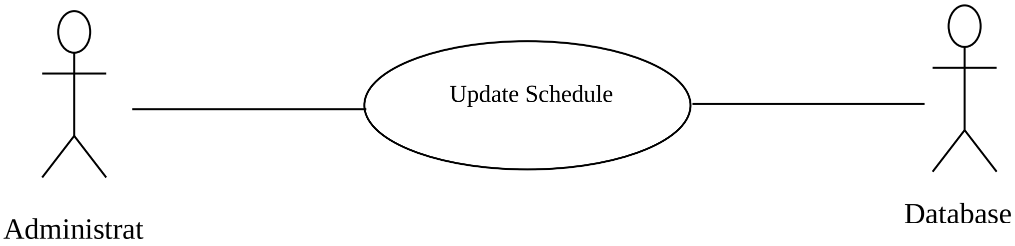
Use case: Update Schedule **[T8]**



Use-case ID:	update_schedule		
Use-case Name:	Update Schedule		
Created By:	Trey A.	Last Updated By:	
Date Created:	3/27/22	Date Last Updated:	
Actor:	Administrator		
Description:	This will allow editing of the times of existing plays		
Preconditions:	Use case: Login -> Administrator		
Post conditions:	MySQL update -> Customers see change in the Customer Module		
Priority:	High		
Frequency of Use:	Low		
Normal Course of Events:	Administrator will authenticate and select "Update Schedule" from the list of options in the Maintenance Module		
Alternative Courses:	Authentication fails; Enters different section		
Exceptions:			
Includes:	Administrators		
Special Requirements:	Privileged access		
Assumptions:			
Notes and Issues:	Email should be sent out to paying Customers notifying them of the time change		

Table:

Diagram:



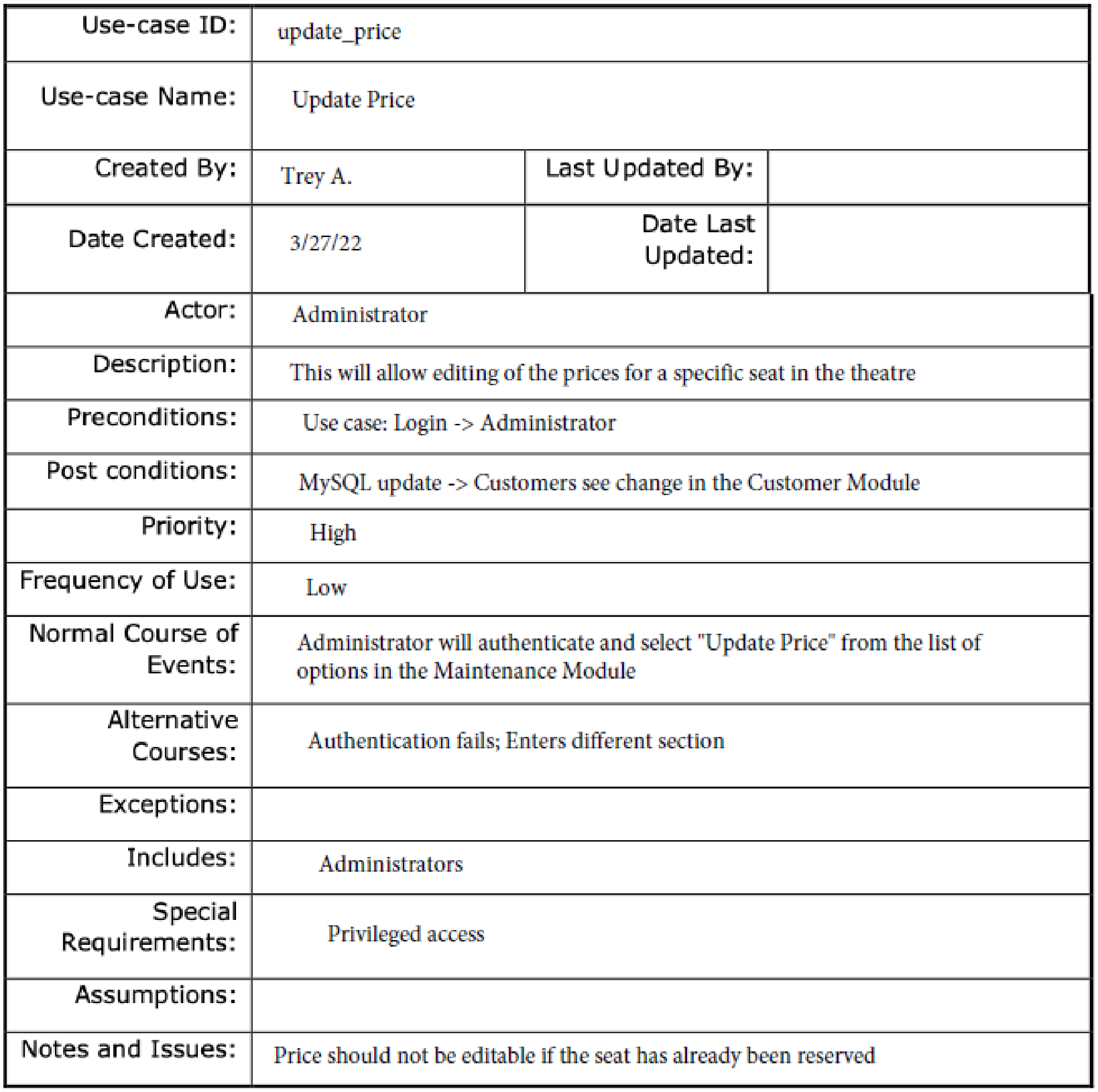
Brief Description

The administrator can manage the scheduling of plays.

Initial Step-By-Step Description
Before this can be initiated, the administrator must access the web application software and authenticate accordingly. Then, the administrator must navigate to the Schedule section of the Maintenance module.

1. The Administrator selects *Schedule* from the Maintenance module.
2. The system presents a list of current Plays.
3. The Administrator chooses a play to schedule (or reschedule).
4. The system links to the existing database.
5. The system provides fields relevant to the Play's showtimes.
6. The Administrator fills in the information and submits the form.
7. The system verifies the information and updates the database.

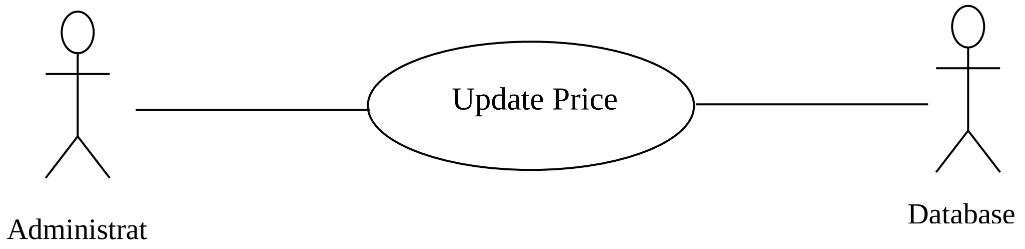
Use case: Update Price **[T9]**



Use-case ID:	update_price		
Use-case Name:	Update Price		
Created By:	Trey A.	Last Updated By:	
Date Created:	3/27/22	Date Last Updated:	
Actor:	Administrator		
Description:	This will allow editing of the prices for a specific seat in the theatre		
Preconditions:	Use case: Login -> Administrator		
Post conditions:	MySQL update -> Customers see change in the Customer Module		
Priority:	High		
Frequency of Use:	Low		
Normal Course of Events:	Administrator will authenticate and select "Update Price" from the list of options in the Maintenance Module		
Alternative Courses:	Authentication fails; Enters different section		
Exceptions:			
Includes:	Administrators		
Special Requirements:	Privileged access		
Assumptions:			
Notes and Issues:	Price should not be editable if the seat has already been reserved		

Table:

Diagram:



Brief Description

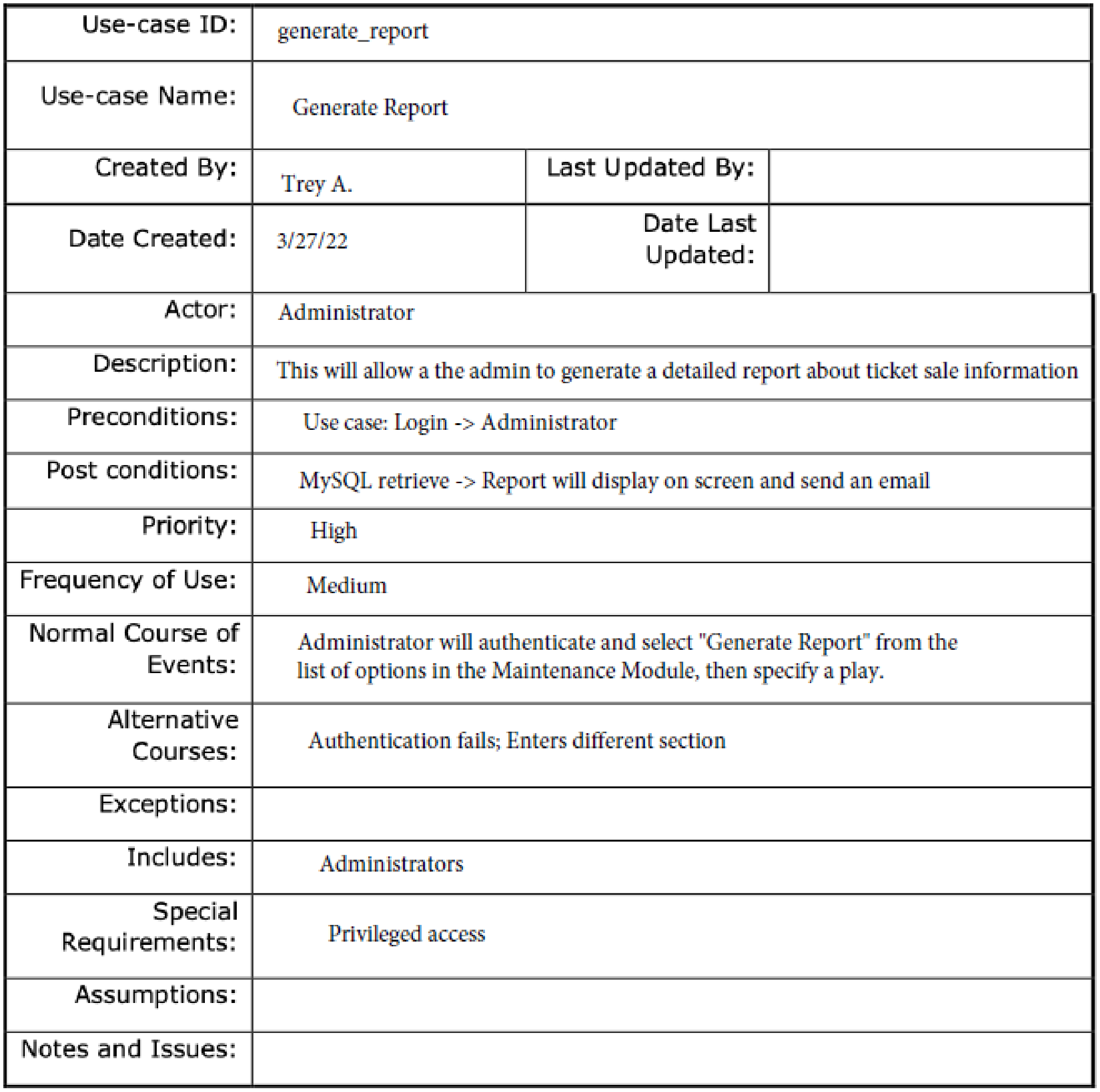
The administrator can manage the price of the tickets.

Initial Step-By-Step Description

Before this can be initiated, the administrator must access the web application software and authenticate accordingly. Then, the administrator must navigate to the Price section of the Maintenance module.

1. The Administrator selects *Price* from the Maintenance module.
2. The system presents a list of current Plays.
3. The Administrator chooses a play to manage seat prices.
4. The system links to the existing database.
5. The system provides fields relevant to the pricing of tickets.
6. The Administrator fills in the information and submits the form.
7. The system verifies the information and updates the database.

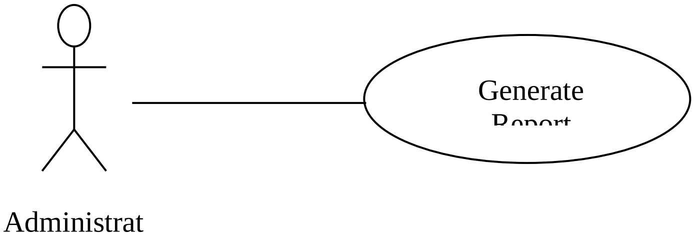
Use case: Generate Report **[T10]**



Use-case ID:	generate_report		
Use-case Name:	Generate Report		
Created By:	Trey A.	Last Updated By:	
Date Created:	3/27/22	Date Last Updated:	
Actor:	Administrator		
Description:	This will allow a the admin to generate a detailed report about ticket sale information		
Preconditions:	Use case: Login -> Administrator		
Post conditions:	MySQL retrieve -> Report will display on screen and send an email		
Priority:	High		
Frequency of Use:	Medium		
Normal Course of Events:	Administrator will authenticate and select "Generate Report" from the list of options in the Maintenance Module, then specify a play.		
Alternative Courses:	Authentication fails; Enters different section		
Exceptions:			
Includes:	Administrators		
Special Requirements:	Privileged access		
Assumptions:			
Notes and Issues:			

Table:

Diagram:



Brief Description
The administrator can generate a report of ticket sales.

Initial Step-By-Step Description
Before this can be initiated, the administrator must access the web application software and authenticate accordingly. Then, the administrator must navigate to the Reports section of the Maintenance module.

1. The Administrator selects *Reports* from the Maintenance module.
2. The system presents a list of Plays.
3. The Administrator chooses a play to see statistics on its sales.
4. The system links to the existing database.
5. The system generates a PDF report.
6. The system emails the report to the administrator.

Priority 5

3.2 Non-Functional Requirements

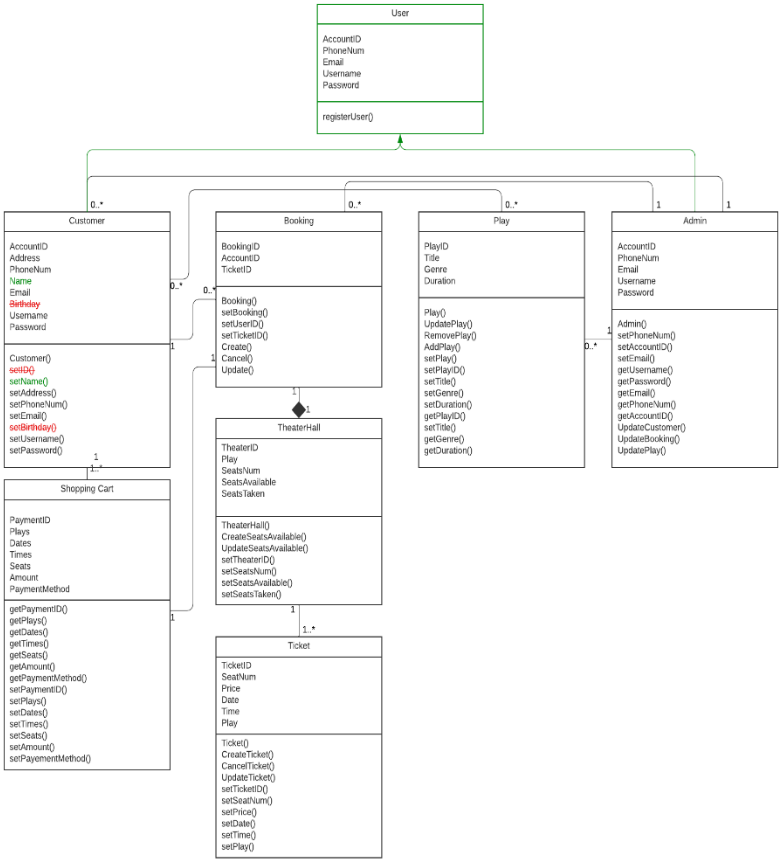
The Theatre Booking System will run on a server provided by contracting staff members. The server is expected to run on a high-speed internet connection along with the database. The workstation and/or tablet to be used by the Administrator and Customers will be provided by Los Portales Theatre. The workstation is expected to have internet capability. Workstation is bilanguage friendly - languages supported: English, Spanish. GUI will translate instantaneously with a click of a button.

The system will operate in a user-friendly, non-burdensome, and optimized interaction (user guide to be provided). The GUI will work seamlessly for any device, including personal mobile phones or tablets (menus adjust to scale). Furthermore, updates and requests will run smoothly and efficiently to avoid disruption and/or User frustration. Administrators will have database access in a secure manner. Refer to 3.3.2 (Security).

3.3 Requirements Specification

This system software will require a third-party vendor for processing payments. Robert Leal’s contracting group does not provide a credit card processing system; however, the system software is able to integrate with any processing software.

The *Credit Card Processing System* will communicate with a server using the Hypertext Transfer Protocol Secure (HTTPS) protocol. All transactions will be secure.



3.3.1 Logical Structure of the Data

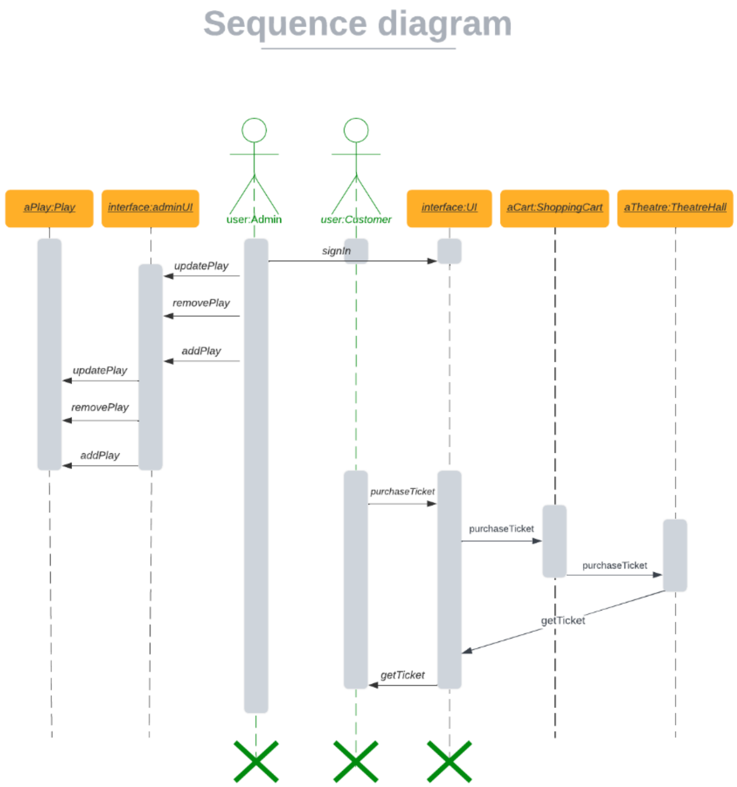


Figure 1 – Sequence Diagram

3.3.2 Security

As addressed above, the server will communicate via HTTPS protocol for secure transmission of data. Furthermore, the server's DNS will route through Cloudflare, a leading provider that ensures traffic is protected from Distributed Denial of Service (DDoS) attacks. The forms' input will be sanitized to prevent code from executing.

The system will run auto-lockout scripts, disable the cache, provide pattern-hiding displays for passwords, only provide the last four digits of a credit card number, and encrypt the databases that are only accessible to administrators of the system software.

As the system software will allow credit card retention with the consent of the Customer, the system will abide by Amazon Web Services (AWS) Service License Agreement (SLA) and safely store information on a private, dedicated host node. These nodes are backed up regularly (once an hour) and are run in server farms for redundancy. Uptime remains

99.9% as of March 27th, 2022. Plenty of storage is provided but can be expanded at the request of the client to ensure business continuity (for extra cost).

The system requires Users (of both subclasses) to create a password that meets minimum password complexity requirements: [passwords stored as BCrypt Hash]

- Minimum of 8 characters
- Use of a capital letter
- Use of a lowercase letter
- Use of a number
- Use of a special character/symbol

The system will accept payment in the form of cryptocurrency secured by the blockchain. Cryptocurrency allows for confidentiality and the inability to obtain transaction information by an unintended party. Third-party payment processors will be responsible for secure transactions, other than the SSL encryption mentioned above.

Product testing to take place to ensure system works efficiently. For example, offering only one theatre room at a single given time to avoid confusion, or attempting to break the system software by simulating peak-hour traffic or malicious users. If the system software can withstand these “penetration” tests, the prototype will be ready to publish for the clients’ viewing.

4. Software Design Document

4.1 Introduction

This design will detail the implementation of the requirements as defined in the Software Requirements Specification for Theatre Booking System.

This system will be an online interface used for purchasing tickets. The system will have multiple types of users, those being administrators and customers. A user must go through a registration process before performing any actions. A customer user will be capable of purchasing tickets, and an administrator will be capable of uploading, removing, and modifying plays. These plays will have a limited number of seats, and consequently, a limited number of tickets available. The system will be able to identify the current capacity of a play, the remaining number of seats, the price of a ticket, the duration of a play, and several more characteristics, including genre and title. The system will be able to process transactions regarding the purchase of tickets, as well as potential refunds. The system will utilize PHP and JavaScript, with a database in SQL.

Contained in this Software Design Document, the reader will find a description of the system architecture, as well as the design and motivation for methodology of the system's components and methods. The database schema, W.I.P. user interface, and requirements matrix can also be found.

Definitions and Acronyms PHPMyAdmin - 3rd party database software MySQL - 3rd party database software

4.2 Design Considerations

As there will be two different types of users, there will be one parent class called "User", which is inherited by two subclasses, "Administrator" and "Customer". These users will have common shared functions, like functions used in the retrieval of user information, identification, and login.

Assumptions The physical theatre location will have internet access, as well as a connection to the webserver from which our system is hosted on.

The system will be contained in a webserver, which is also where the database is contained. The system will primarily be written in PHP and JavaScript, and will be accessed from the client at the workplace. The database used to store data will be MySQL and PHPMyAdmin. Employees at the workplace will connect to the webserver via desktop computers.

The system will be storing sensitive information, including credit-card numbers and addresses. Although the system has implemented protection against database infiltration in several ways, including input sanitization, it is worth noting that sensitive customer information and data are being stored – and furthermore, protected.

4.3 Architecture, Database, and Requirements Matrix

The system's architecture follows a client-server model. The system's architecture can be accurately described by it's use-case diagram, found on page 3. The user first connects to our webserver, which houses our database. Our database interacts with our existing codebase using PHPMyAdmin and MySQL. Upon connection to the webserver, the user will login as either a customer or administrator – both of which are subclasses of parent class User. Depending on which type a user is logged in as, they will be directed to different modules. These modules are Customer Module and the Maintenance module. From here, customers can select plays, and administrators can manage plays. Additionally, both types of users can choose to update their account information -- interacting with our database. Upon customer interaction with plays, they can choose to purchase tickets by updating their cart. Administrators can choose to update available plays, update prices, update play scheduling, and possibly refund users. Both types of users will interface with a file called “logout.php”, which will allow the users to log out. The major responsibility of both users is to inevitably interact with our database. The Customer user achieves this by updating their account information, viewing a list of updated seats, and by updating their cart and making purchases. An Administrator user achieves this by updating their account information, and by managing plays.

The customer will be inherited from the parent User class. As mentioned in 3.1, the Customer will be able to edit their account information, select plays to view, and purchase tickets.

The administrator will be inherited from the parent User class. As mentioned in 3.1, the Administrator will be able to edit their account information, select plays to modify, and potentially refund tickets.

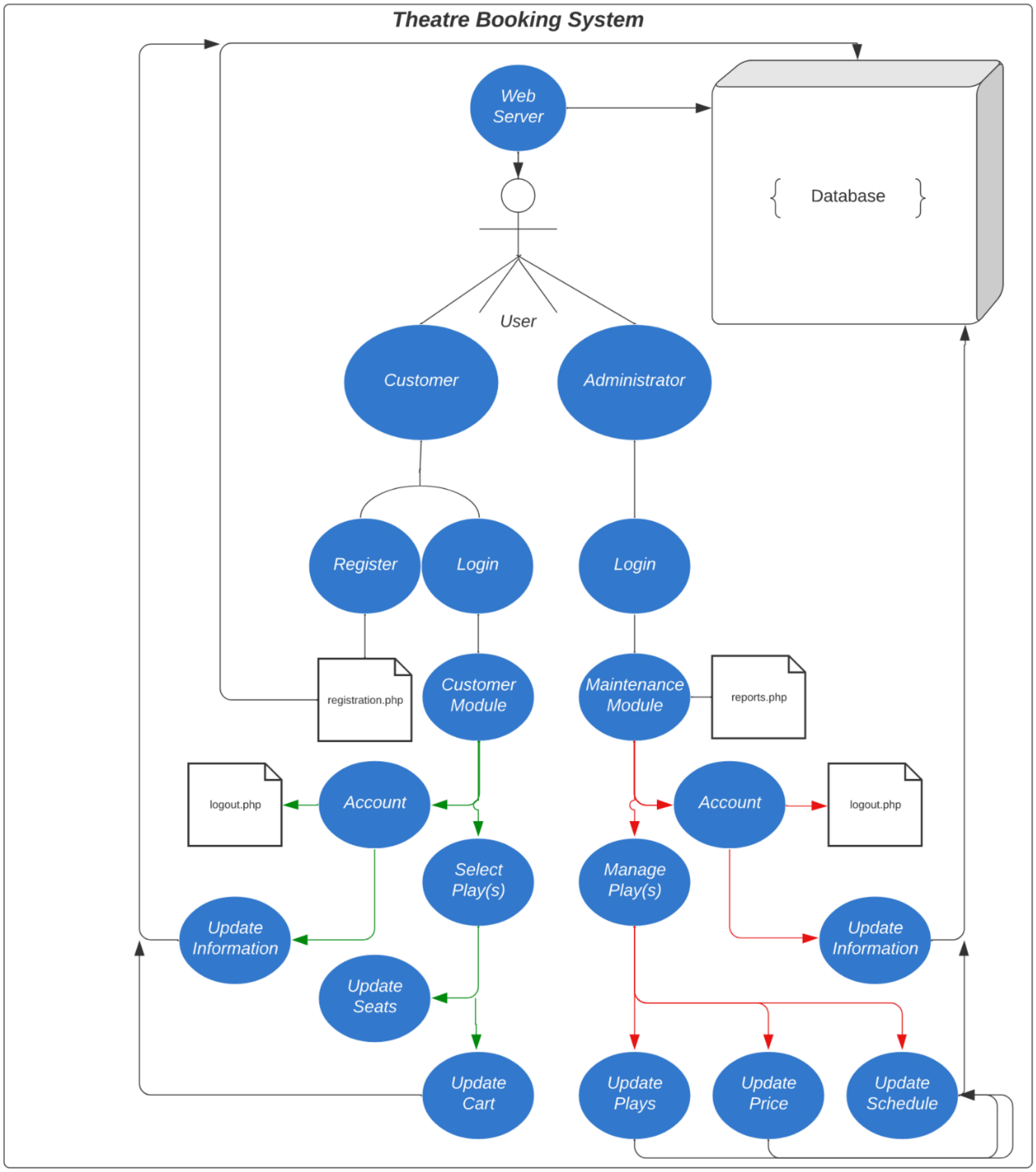
Plays will belong to their own class, where they will be given attributes such as genre, title, length, and ticket prices. Administrator users will perform actions on these plays, such as updating their ticket price or availability.
.

The shopping cart will belong to its own class, where Customer users will be able to perform actions on it, including adding tickets to the cart, removing tickets to the cart, clearing the card, and purchasing.

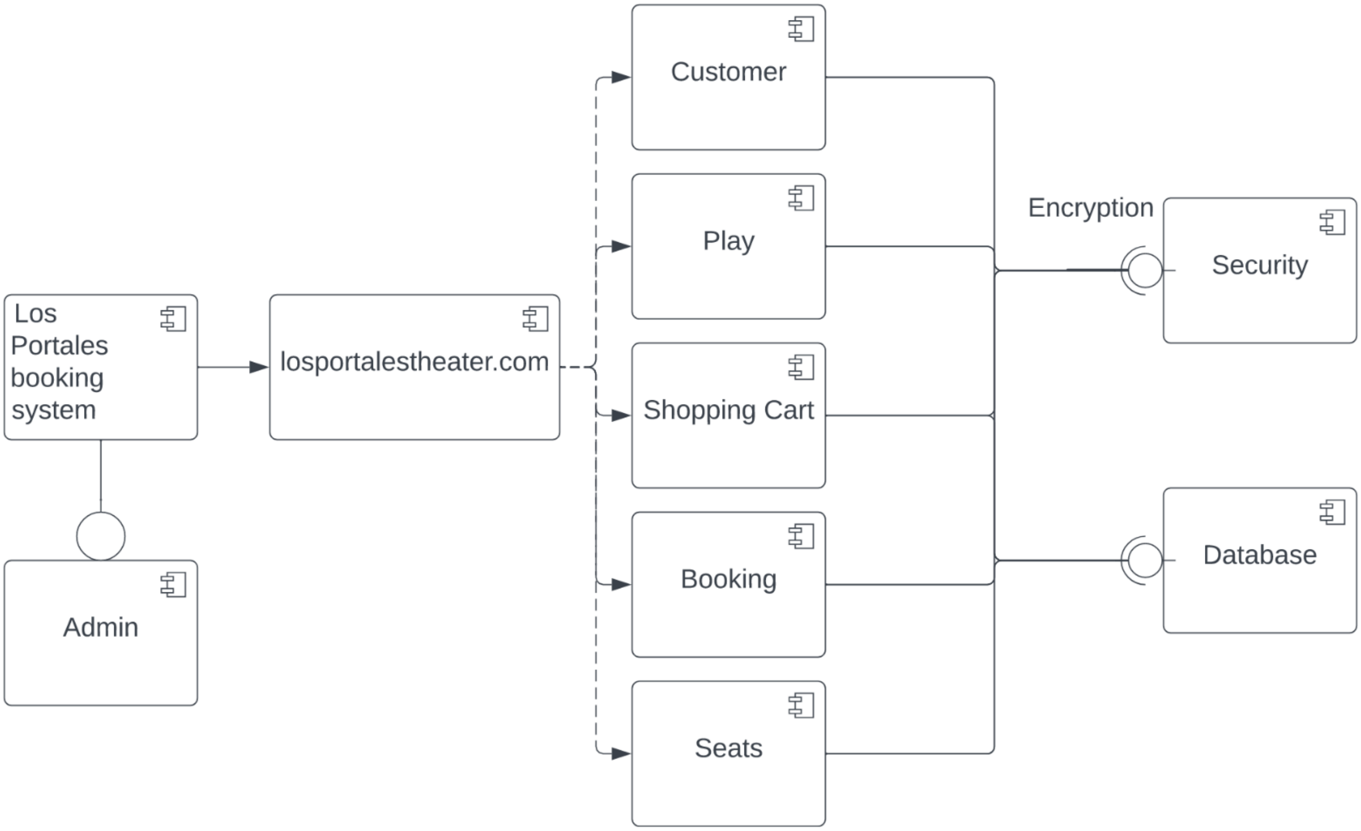
Tickets will belong to their own class. Objects of this class will be assigned to Customers upon purchasing a ticket.

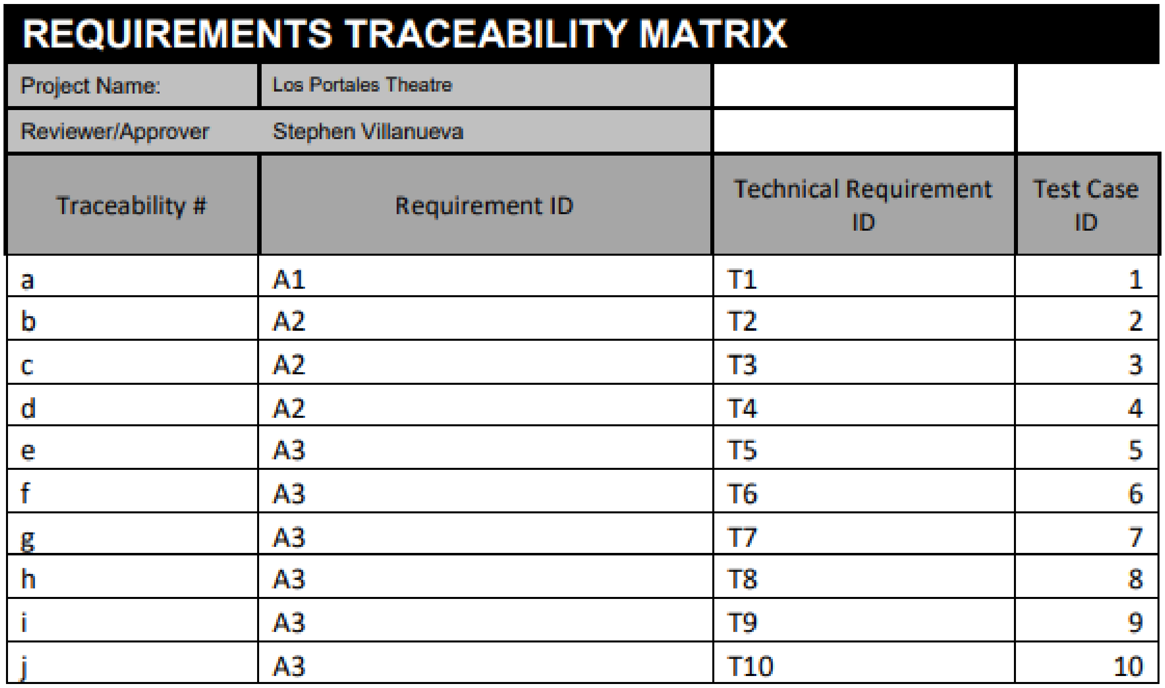
For the system, an object-oriented approach is being utilized. This appeared to be the most effective approach, as many of the system’s functional requirements are similar between different components.

Our tentative list of tables includes USERS, TRANSACTIONS, PLAYS. The USERS table will have fields UserID, Admin, CardNumber. The TRANSACTIONS table will have fields UserID, PlayID, TicketID, Date, TicketPrice The PLAYS table will have the fields PlayID, Genre, Title, Length, Tickets, TicketPrice. The USERS and TRANSACTIONS tables will share the UserID field. The TRANSACTIONS and PLAYS tables will share the PlayID and TicketPrice fields.



Theatre Booking System





REQUIREMENTS TRACEABILITY MATRIX

Project Name:	Los Portales Theatre		
Reviewer/Approver	Stephen Villanueva		
Traceability #	Requirement ID	Technical Requirement ID	Test Case ID
a	A1	T1	1
b	A2	T2	2
c	A2	T3	3
d	A2	T4	4
e	A3	T5	5
f	A3	T6	6
g	A3	T7	7
h	A3	T8	8
i	A3	T9	9
j	A3	T10	10

5 Software Test Plan

5.1 Strategy and Approach

Testing Strategy

Testing will include testing of all functionalities that are in scope (Refer Functional Scope Section) identified. System testing activities will include the testing of new functionalities, modified functionalities, functionality access, and testing of databases.

System Testing Entrance Criteria To start system testing: • Complete testable code is available • Requirements set by client have been met • Ability to obtain or simulate test data • Test environment has been set-up

Testing Types
Software Inspection A review of the code will be done by the test team in order to help fix any noticeable issues or bugs before the testing process begins. This will help mitigate the need for extra test cycles.
Functional Testing The test team will ensure that the functional requirements set by the client are met: • Documentation updates reflect the changes made to the website itself • New users must register • All card types are accepted • No special users • User information requirements (age, location, name, email, and phone number) • User payment option retention • Secure accounts and transactions Page 3
Development Testing The testing done in this stage will be done during and after development by the development team. They will document focused testing in areas the development team is concerned about.
Use-Case Testing The Use-Case* testing will be done by the testing team and will include the following: • Register – user • Update user information - customer • Update seats – customer • Update cart - customer • Send purchase - customer • Update information - administrator • Update schedule - administrator • Update price – administrator • Generate report – administrator
Scenario Testing The test team will include five different scenarios for testing and will act as a customer: Scenario 1: Customer registration – test team will satisfy several test cases** • All combinations of valid and invalid name, email, password, address, phone • Capital letter • Lowercase letter • Number • Special character/symbol Scenario 2: Login • All combinations of valid and invalid email and password • Forgot password? • Remember me: checked and unchecked Scenario 3: Seats • Select and deselect seat/seats • Add seats to cart Scenario 4: Cart • Remove seats from cart • Proceed to checkout Scenario 5: Payment Functionality • System accepts several types of payment • Automated send receipt.

User Testing All user testing will be simulated though the scenarios previously described. 3.4 Suspension Criteria and Resumption Requirements Suspension Criteria If testing discovers a critical error in the website that is causing a loss of functionality, we will suspend testing until the issue is resolved. Resumption Requirements The error will be retested in the same manor it was discovered. If the issue does not reoccur, the test team will resume with the scheduled testing.

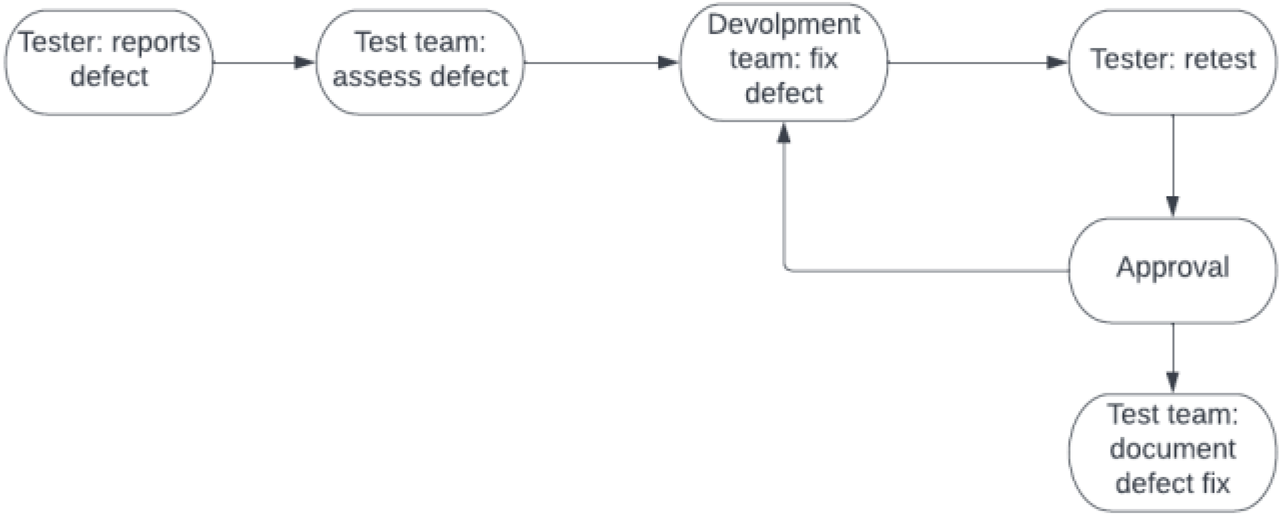
Test Data The test data will be derived from the test cases that come from our test scenarios. All the test data will be organized and documented using a browser extension called Selenium IDE.

5.2 Execution Plan

The test team will conduct each scenario as an actor playing the role of a customer. Each test case will be covered by inputting the test data and recording the result via the Selenium IDE.

5.3 Defect Reporting

Selenium IDE will be used for defect tracking.



Each defect discovered through testing will be categorized and sent to the development team to be corrected. When the defect is corrected it will be documented and marked as complete by the test team when it is retested and causes no error.

- Critical – any defect that causes a major loss of functionality and renders the website inoperable.
- Moderate – any defect that causes some loss of functionality without total loss of operation.
- Minimal – primarily cosmetic and design defects

6 Test Reports