MOVIE SHOW TIME

Schwifty

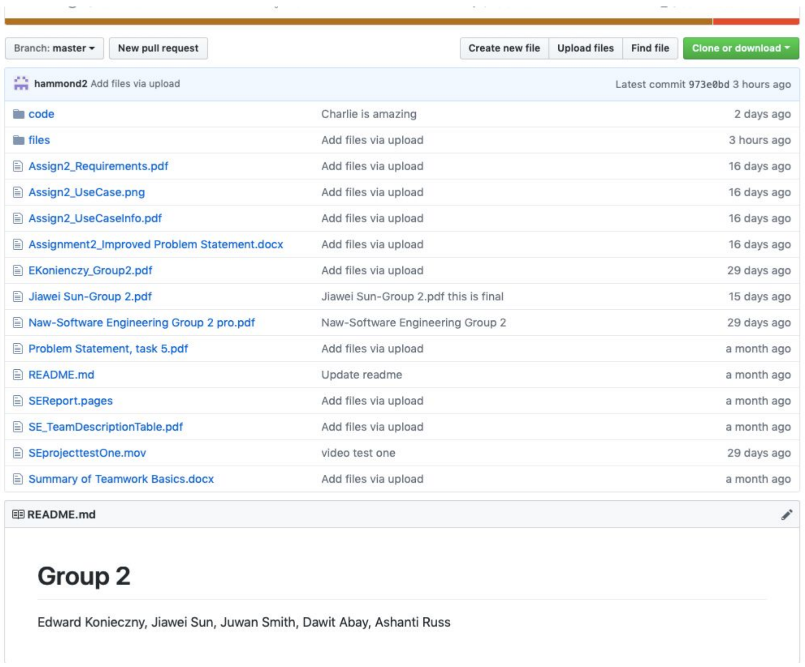
Ashanti Russ
Edward Konienczy
Dawit Abay
Jiawei Sun
Team coordinator: Juwan Smith

Spring 2019

TASK 1:
Planning Scheduling and Peer Evalua

Name	Email	Task	Hour	Dependency	Date	Evaluation
Juwan Smith(TC)	jsmith405@student.gsu.edu	Task 6 Testing	4h	Testing	3/3/19	97% did a great job Testing
Dawit Abay	Dabay1@student.gsu.edu	Task 1/2/7 planning/ Report	3h	Planning/ Report/Github	3/3/19	90% did a good job
Jiawei Sun	jsun8@student.gsu.edu	Task 3 Revise/ Refine	4h	Revise/ Refine	3/3/19	100% did a excellent job Revise/ Refine
Edward Konienczy	EKonienczy1@student.gsu.edu	Task 5 Implementation	3h	Implementation	3/3/19	88% did a good job
Ashanti Russ	aruss3@student.gsu.edu	Task 4 System Modeling	5h	System Modeling	3/3/19	100% did a excellent job System Modeling

TASK 2:
Communication and Collaboration
Github



Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

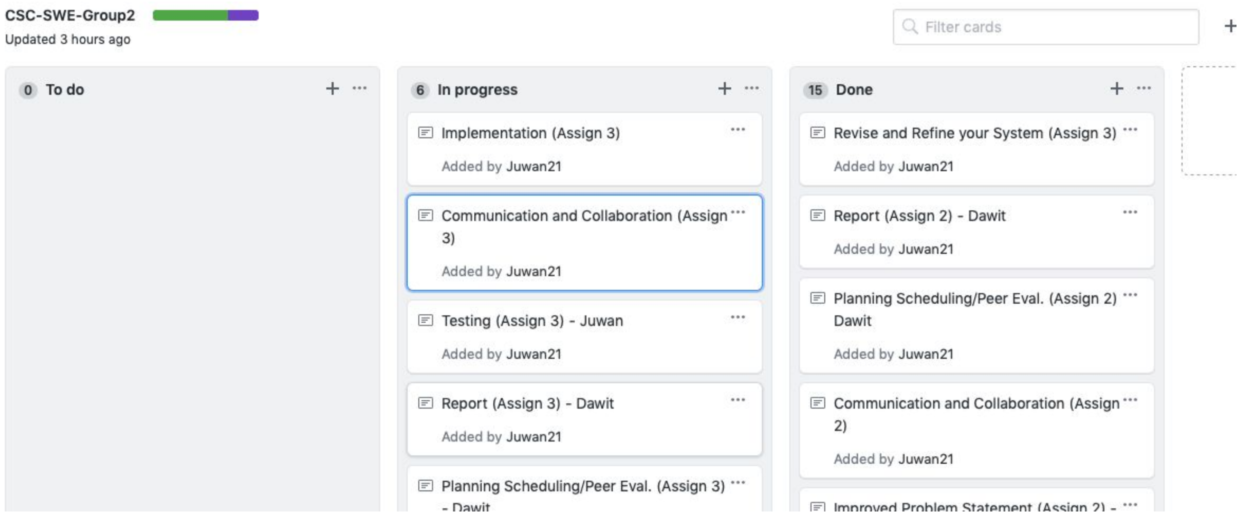
hammond2 Add files via upload — Latest commit 973e0bd 3 hours ago

code	Charlie is amazing	2 days ago
files	Add files via upload	3 hours ago
Assign2_Requirements.pdf	Add files via upload	16 days ago
Assign2_UseCase.png	Add files via upload	16 days ago
Assign2_UseCaseInfo.pdf	Add files via upload	16 days ago
Assignment2_Improved Problem Statement.docx	Add files via upload	16 days ago
EKonienczy_Group2.pdf	Add files via upload	29 days ago
Jiawei Sun-Group 2.pdf	Jiawei Sun-Group 2.pdf this is final	15 days ago
Naw-Software Engineering Group 2 pro.pdf	Naw-Software Engineering Group 2	29 days ago
Problem Statement, task 5.pdf	Add files via upload	a month ago
README.md	Update readme	a month ago
SEReport.pages	Add files via upload	a month ago
SE_TeamDescriptionTable.pdf	Add files via upload	a month ago
SEprojecttestOne.mov	video test one	29 days ago
Summary of Teamwork Basics.docx	Add files via upload	a month ago

README.md

Group 2

Edward Konieczny, Jiawei Sun, Juwan Smith, Dawit Abay, Ashanti Russ



CSC-SWE-Group2
Updated 3 hours ago

Filter cards

0 To do

6 In progress

Implementation (Assign 3)
Added by Juwan21

Communication and Collaboration (Assign 3)
Added by Juwan21

Testing (Assign 3) - Juwan
Added by Juwan21

Report (Assign 3) - Dawit
Added by Juwan21

Planning Scheduling/Peer Eval. (Assign 3) - Dawit

15 Done

Revise and Refine your System (Assign 3)
Added by Juwan21

Report (Assign 2) - Dawit
Added by Juwan21

Planning Scheduling/Peer Eval. (Assign 2) Dawit
Added by Juwan21

Communication and Collaboration (Assign 2)
Added by Juwan21

Improved Problem Statement (Assign 2) -

TASK 3:
Revise and Refine your System

1. Refined problem statement

1) What is your product, on a high level?
Our product provides a service of making showtimes of movies being presented at local movie theatres readily available for potential consumers of entertainment.

2) Whom is it for?
This product is available for any consumer that is in want of an entertaining experience that is within a relatively close environment of where they live and the necessary information, we provide to make that possible. Especially for movie fans who wants to watch the coming movie in ideal place for him or her.

3) What problem does it solve?
It provides readily available information for anyone who does not intuitively know or have access to information regarding the closest proximity movie theatre or showing times for movies in his or her area.

4) What alternatives are available?
Applications such as “Fandango” and “cinema” provide a relative competitive result to what we provide. Also, the more tedious methods are readily available such as finding theatres and showtimes manually through search engines.

5) Why is this project compelling and worth developing?
This project allows for some initial experience with being an individual within a development team and understand the quirks and intricacies required to work as a cohesive unit to achieve some common goal. Along with the previous statement, this provides an opportunity to test our current technical skills gained as students and find areas that we need to improve and hone these skills further.

6) Describe top-level objectives, differentiators, target customers, and scope of your product.
Top-level objectives: Provide easily accessible and free upcoming movie information including movie show time and locations to potential movie consumers.
Differentiators: Our product is limited to the relative Atlanta area so the amount of traffic would be less of an issue opposed to our competitors. And it costs less resources.

Target customers: Customers within the Metropolitan and greater Atlanta areas who interested in movies.
Scope of your product: The scope is relatively consistent with in the realm of movie Entertainment.

7) What are the competitors and what is novel in your approach?
Our competitors include “Fandango” and “cinema”, and other movie information websites. We provide useful information exclusively for the Atlanta area.

8) Make it clear that the system can be built, making good use of the available resources and technology.
There are many frameworks to build this system. For our project, we will Spring Boot as our framework, and use HTML, JavaScript and CSS to create our front-end, Java for back-end, then PostgreSQL for database.

9) What is interesting about this project from a technical point of view?
This project allows for our team to experience the use of full stack development in a single process by needing to create and environment to display information from databases that we create and then port said information over to a visual format to be used by potential consumers.

2. Refined requirements

(1) System Requirements

1) Check Movie Times
Actors: Customers, Developers, Movie Theatres, Database System
Description: The website contains a variety of showing times for films being shown at local theatres. This provides a free service to customers in need of pertinent information.
Alternate Paths: You could directly search for a specific film you believe to be in theatres and all the required info will be available along with, of course, the show times.
Pre-Condition: The local movie theatres have accurate information regarding the show times for their films. Also, our development team is consistent with the updating accurate information on the website.

2) Search Movies
Actors: Customers, Developers, Database System, Movie Theatres
Description: The website allows the ability to search for current available films that are being shown at local theatres if they’re indeed in our system.
Alternate Paths: You could visit your local theatre directly to find a specific film.
Pre-Condition: Information from the local theatres is crucial for it be accurate and displayed for customers.

3) Buy Tickets.
Actors: Customers, Develops, Movie Theatres

Description: The website contains the ability to purchase tickets electronically for showings at local theatres.
Alternate Paths: You could visit your local theatre and pay for tickets upon arrival.
Pre-Condition: The local movies theatres provide information and available tickets to be sold to customers, that is facilitated through our platform.

4) Watch Trailers
Actors: Customers, Developers Description: The website provides film trailers for current films being shown at local theatres.
Alternate Paths: None.
Pre-Condition: Developers are staying up to date on current films being shown and then acquiring the necessary resources to apply the trailers and info onto the platform. Group 2 CSC 4350

5) Development Team Information
Actors: Developers, Customers, Movie Theaters
Description: The website contains the necessary information on the development such as names, experience, background information etc. This is available to the public.
Alternate Paths: Going to the development team directly in order to gather the desired information.
Pre-Condition: The development continues to keep and update accurate and true information regarding the team and perceived qualifications and experience.

6) Find Theatre Locations
Actors: Customers, Developers, Movie Theatres, Database System
Description: The website contains a variety of information on local theatres that allow you to enjoy a safe, comfortable film experience.
Alternate Paths: None.
Pre-Condition: The developers keep up to date information on the locations of certain theatres and what films are currently being shown at those aforementioned theatres.

(2) User Requirements
1) Retrieve Movie Times
Introduction: This requirement is related to the need of displaying pertinent data for the films on the system.
Inputs: A name or date regarding a specific film you are trying to watch at a local theatre.
Requirements Description: A robust amount of data for films that are being shown at local theatres to be entered into a database system.
Outputs: The system will then display all results that are possibly related to the input and thus showing a schedule of films being run at a theatre.

2) Retrieve Specific Movie

Introduction: This requirement is related to the need of finding specific information in regards to a specified film.
Inputs: A name or date regarding a specific film you are trying to watch at a local theatre.
Requirements Description: Data from a multitude of different films being shown and input into a database to be displayed.
Outputs: The system will return a list of films that are related to the input and then give a certain amount of information on the searched film.

3) Ticket Retrieval
Introduction: This requirement is related to the need of providing an electronic version of a ticket to gain access toward attending a film at a theatre.
Inputs: A credit or debit card of some kind is needed in order to perform an epurchase of a ticket for a film at a local theatre.
Requirements Description: A secure encrypted system that allows a customer to make and charge a payment online.
Outputs: An e-ticket will be sent to a device of a customer that can be used to attend a film at a local theatre.

4) Stream Trailer
Introduction: This requirement is related to the need of providing a short clip displaying an introduction to films being shown at local theatres.
Inputs: A name of the film you are searching for.
Requirements Description: A robust amount of short video files that are uploaded to a database system in order to be displayed on the system.
Outputs: Shorts clips showing scenes from films that are being shown at local theatres.

5) Display Development Team Info
Introduction: This requirement is related to the need of providing information on the development team of the system.
Inputs: No input required. Simply click relevant menu option.
Requirements Description: Information that amounts to a synopsis on the relevant members with in the development team to be uploaded onto the database.
Outputs: Documents that display the abilities and skills of the development team members.

6) Show Theatre Locations
Introduction: This requirement is related to the need of providing locations and information of local theatres you would want to visit.
Inputs: A name of a film you believe to be currently shown at a local.

Requirements Description: Information in regard to films being shown currently, also operating hours and locations of local theatres that have said films available.
Outputs: Times relating to the input of the film being searched and local theatres that are currently showing the film.

3. Refined Use Case Diagram
In this part, we use buy tickets as our use case to show class diagram, which is shown below.

Use case: Buy Tickets
Actors: Customers, Database, Payment Service
Description: The website contains the ability to purchase tickets electronically for showings at local theatres. As customer goes to the website, he or she would use UI to browse the recommendations for movies, then select movie time and pay for tickets.
Alternate Paths: You could visit your local theatre and pay for tickets upon arrival.
Pre-Condition: The local movies theatres provide information and available tickets to be sold to customers, that is facilitated through our platform.
Class diagram:

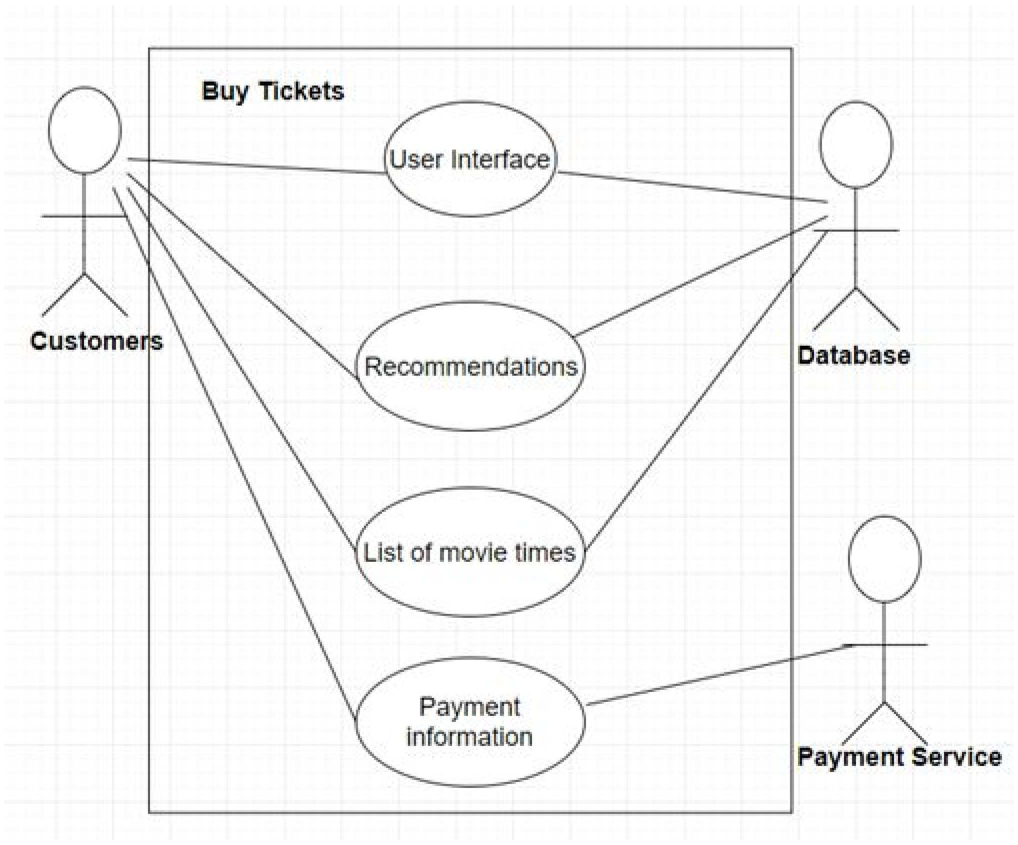
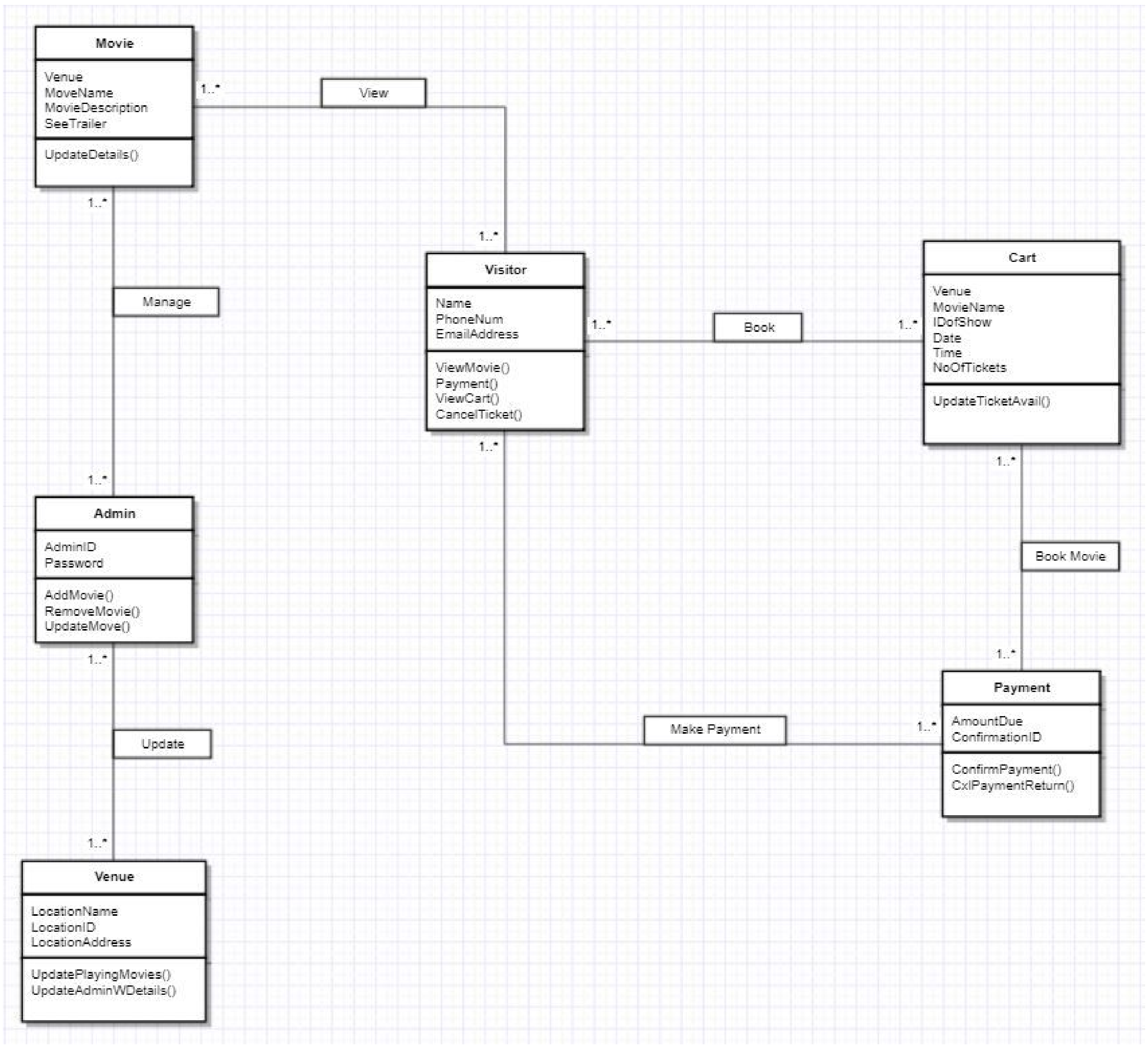


Figure: Class Diagram

4. Refined class diagram

Class Diagram



Class Diagram Analysis				
Objects	Associations	Multiplicity (Instances)	Attribute	Defined Operations

Venue	Admin	Update Admin: One or more	Location Name
			Location ID
			LocationAddress
				UpdatePlayingMovies()
				UpdateAdminWDetails()

Admin	Venue
	Movie
		Update by Venue:One or more instances
		Manage Movie: One or more
			AdminID
			Password
				AddMovie()
				RemoveMovie()
				UpdateMovie()

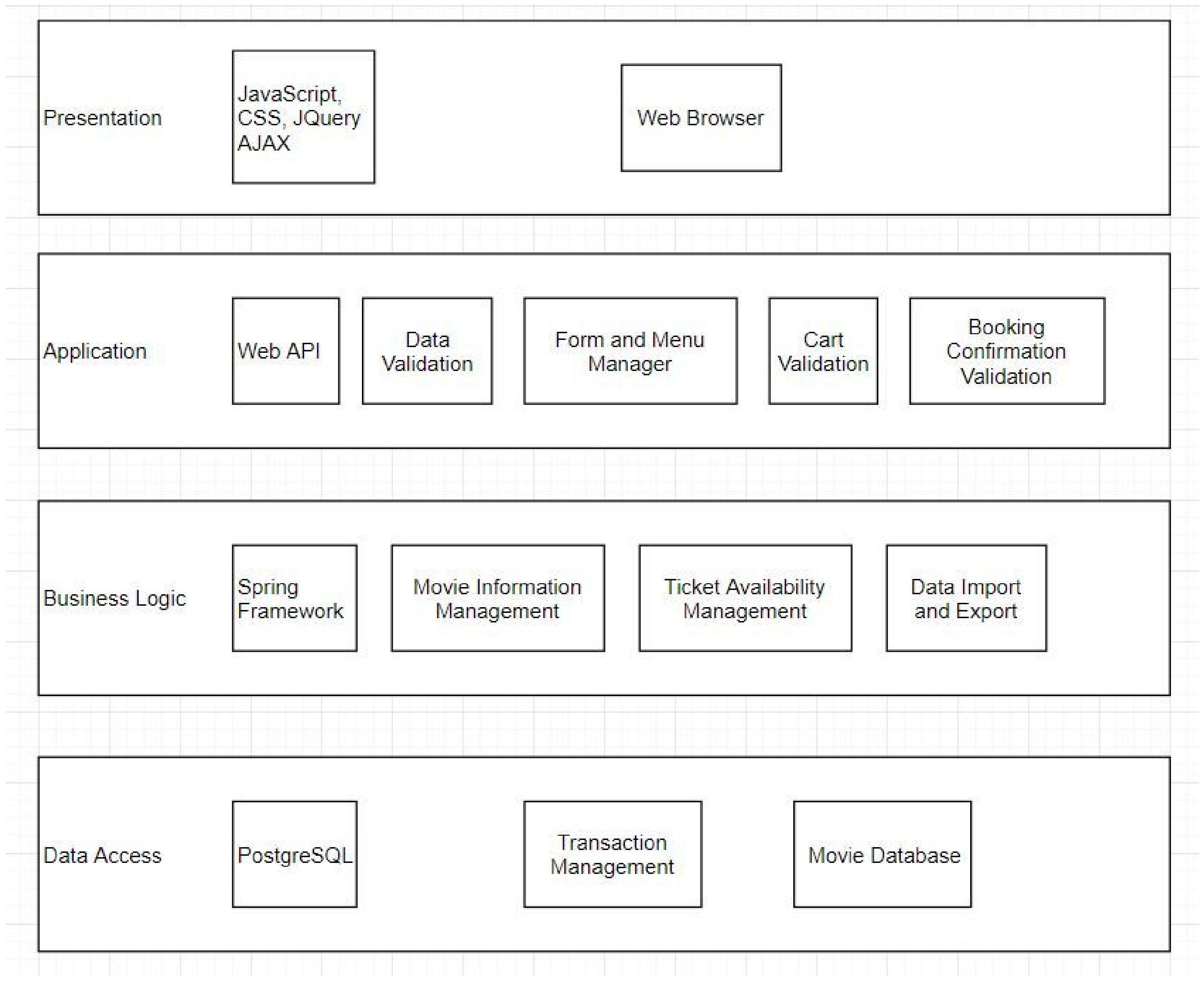
Movie	Admin Visitor	Manage by Admin: One or more View by Visitor: One or more	MovieName MovieShow Venue SeeTrailer	UpdateDetails()

Visitor	Movie
	Cart
	Payment
		View Movie: One or more
		Book via Cart: One or more
		Make a Payment: One or more
			FullName
			PhoneNum
			EmailAddress
				ViewMovie()
				Payment()
				ViewCart()
				CancelTicket()

Cart	Visitor Payment	Book for Visitor: One or more Book Payment: One or more	Venue MovieName IDofShow Date Time NoOfTicketsAvailable	UpdateTicketAvail()

Payment	Visitor
	Cart
		Make Payment via Visitor: One or more
		Book Movie via Cart: One or more
			AmountDue
			ConfirmationID
				ConfirmPayment()
				CxlPaymentReturn()

TASK 4:
System Modeling (Analysis)
Architectural design pattern: Layered



Due to it easily distinguishing and distributing the responsibilities of the web application, a layered architecture is used. As a reminder, the system maintains and manages information on movies, and their ticket availability, that are showing in select venues, and as tickets are purchased, the remaining accessible tickets are updated through the system. Below, is given a detailed description for each layer of the architecture pattern used.

(1.) Top Layer (Presentation Layer)

- The product is a browser-based user interface (BUI) that's presented within a web browser.

(2.) Second Layer (Application Layer)

- The components included within this section provides the user interface functionality given by the web browser. This includes the access for users to browse the system's menus, view updated and validated information that is presented (such as movie showtimes and locations), and to have access to selected tickets within their carts, as well as the booking confirmation provided after making a purchase.

(3.) Third Layer (Business Logic Layer)

- Within this layer,the functionality of the system is implemented as it also provides components that implements the management of movies and their data, the ticket availability of the provided movies, and exchanges (import/export) of movie and ticket data from the system's database to and from venues' databases.

(4.) Lowest Layer (Data Access Layer)

- Transaction management and the databases of movies are provided as they are built using a commercial database management system.

Behavioral (Dynamic) Modeling

The given sequence diagram below provides the scenario of a visitor purchasing tickets. Two major use cases shown are: adding a movie's showtime (which originates from the venue) and the purchasing of a ticket (which originates from the visitor). Additional details concerning the components of the diagram is provided below.

Lifelines:

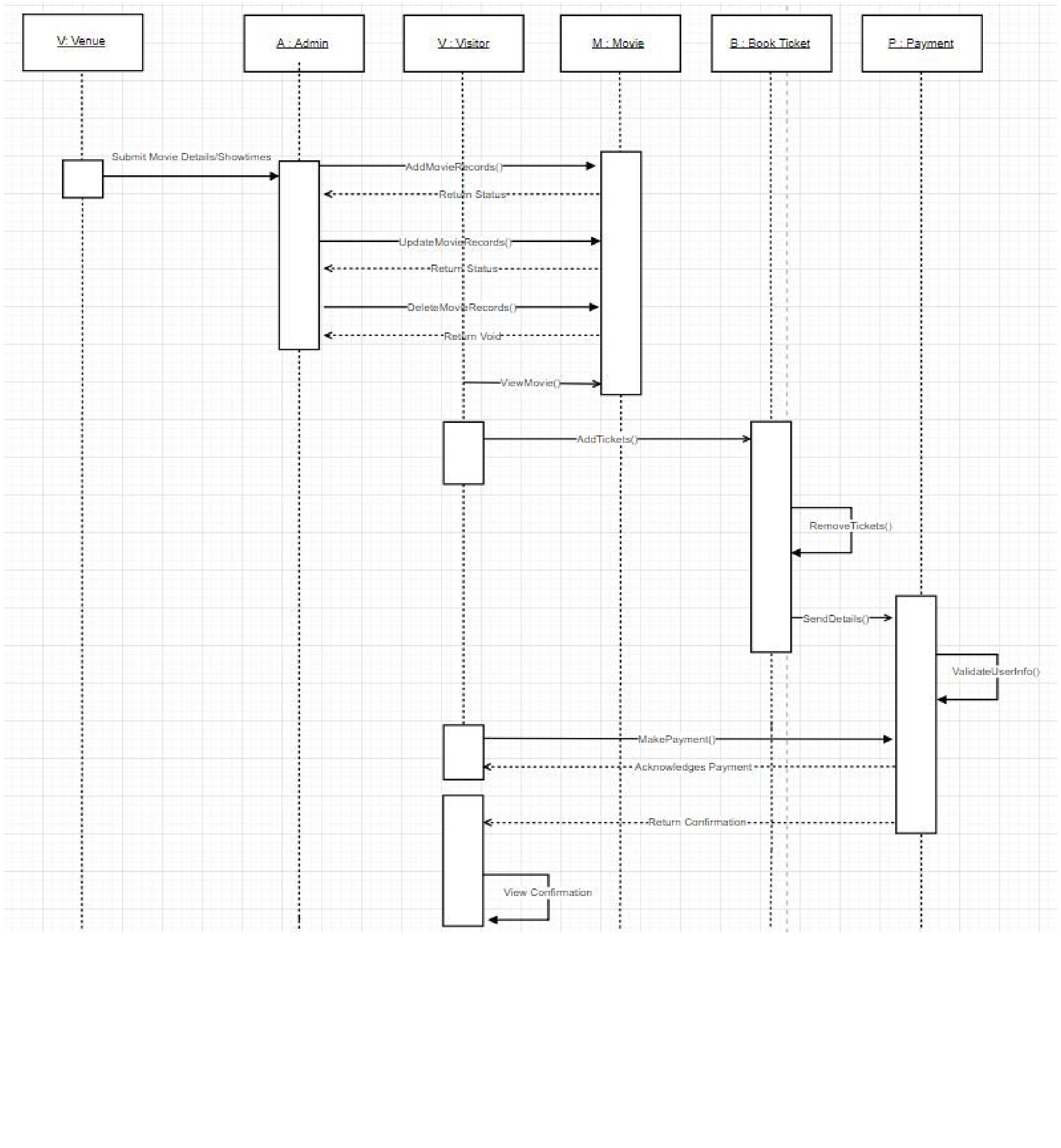
- Admin
- Visitor
- Movie
- Book Ticket
- Payment
- Venue

Synchronous messages:

- AddMovieRecords()
- UpdateMovieRecords()
- DeleteMovieRecords()
- MakePayment()
- ValidateUserInfo()
- View Confirmation
- Submit Movie Details/Showtimes

Asynchronous messages:

- AddTickets()
- ViewMovie()
- RemoveTickets()
- SendDetails()


V: Venue
A : Admin
V : Visitor
M : Movie
B : Book Ticket
P : Payment
Submit Movie Details/Showtimes
AddMovieRecords()
Return Status
UpdateMovieRecords()
Return Status
DeleteMovieRecords()
Return Void
ViewMovie()
AddTickets()
RemoveTickets()
SendDetails()
ValidateUserInfo()
MakePayment()
Acknowledges Payment
Return Confirmation
View Confirmation

TASK 5:
Implementation

The code is compiled and run using an included gradle wrapper. All tasks criteria for this task have been completed as per specifications.

TASK 6:
Testing

Use Cases being tested

• **Watch Trailers**: The website provides film trailers for current films being shown at local theatres.

• **Search Movies**: The website allows the ability to search for current available films that are being shown at local theatres if they're indeed in our system.

Testing Search Movies:

• **Input Partitions: SearchMovie (string some_mov)**
This is the representative function used for the Search Movies Use Case.

- some_mov < A string of zero length
- some_mov = A string of zero length
- some_mov > A string of zero length

• **Test Specification:**

A string that contains multiple characters should be inserted into the Search Movie function. This string should be greater than zero in length and have the string be comprised of any combination of letters, digits, special characters, and spaces. There is no maximum limit to the number of characters being used but there is a minimum length of at least one character.

Description: **some_mov = string > 0; string includes [A-Z, a-z, 0-9, spaces, special char]**

• **Test Cases**

- Expected Input: some_mov = "Ant-Man"
- Expected Output: A link to a page containing the necessary background info, time, and ticket info for the searched movie.
- Expected Input: some_mov = ""
- Expected Output: No available link since nothing was inserted into the search.

Testing Watch Trailers: 3/3/19 Group 2

• **Input Partitions:** A subtest of Search Movies so all the necessary inputs that correspond to the previous stated inputs are used here.

This is the representative function used for the Search Movies Use Case.

- some_mov < A string of zero length
- some_mov = A string of zero length

- some_mov > A string of zero length

• Test Specification:

A link to a movie description and information page is selected and, on this page, should be an embedded video link to a trailer for the searched movie. You should click the video and assuming all external factors, such as a strong connection to the internet and developers linked the correct corresponding video, are correctly met then it should play.

Description: **Mouse input needed for embedded video link.**

• **Test Cases:**

- Expected Input: Mouse Input
- Expected Output: Video link plays embedded video