INFO-C451
System Implementations
Connor Smith

Indiana University Southeast

Midterm Project Demo

Theater Ticket and Seating Manager

By Connor Smith

SP23: SYSTEM IMPLEMENTATION: 26578

March 10, 2023

INFO-C451
System Implementations
Connor Smith

Table of Contents

INFO-C451
System Implementations
Connor Smith

Customer Problem Statements

Problem Statement:

Post-pandemic, many theaters have begun to face issues for seating customers in regards to health, regulations, and efficient seating, especially as more and more people purchase their tickets online in advance rather than at the theater itself. These issues have begun to lead to some theaters selling tickets by seating location. As a customer, not being able to find a good seat for myself and my group's needs can sour the experience, and make me go to a more organized competitor or not go at all.

Glossary:

Identifier - A unique (per section) series of letters/numbers used to designate a specific seat.
Time Slot - A per-showing setting that designates the start and end times of a said showing.
Seating Modes:

Standard - Normal seating procedures, sit wherever you like. No identifiers.
Priority - Certain seats are marked with identifiers, others are first comes first serve.
Assigned - All seats are marked with identifiers. Customers may only sit in assigned seat.
Restricted - Assigned seating, but with the addition of extra space at the ends of a group. To be used with social distancing policies/other related reasons.

INFO-C451
System Implementations
Connor Smith

Requirements Specification

Stakeholders:

Movie/Theater companies (particularly managers at high-traffic locations), ticket aggregators

Actors and Goals:

Primary Actors:

- Customers: This actor can pay for and receive a ticket(s) for a specific showing at a theater.

Secondary Actors:

- Admin: This actor can designate specifications for a given showing at a given theater. All admins should also be Staff.
- Staff: This actor can view the current status of any showing at their location to determine current expected occupancy (for selling tickets to customers who have not per-purchased a ticket, seeing if a room should need more or less oversight, etc). Cannot designate specifications.
- System: This actor is responsible for maintaining the current status of any rooms in the theater, including occupancy (both numerical and positional), time slots, and updating available spots when a ticket is sold.

INFO-C451
System Implementations
Connor Smith

Use Cases:

System (Total: 11):

- Assign ticket: To create a ticket-to-seat pair with a unique identifier (3)
- Show available: To show what seats are currently not occupied for a given time slot (1)
- Show unavailable: To show what seats are currently occupied for a given time slot (2)
- Hide unavailable times: To hide rooms that are currently fully occupied for a given time slot (2)
- Set seating mode: To set the rules for what seats are/are not available (2)
- Remove past showings: To remove past showings from the current schedule (1)

Admin (Total: 14):

- Add room: To add a room to the database (1)
- Remove room: To remove a room from the database (1)
- Add showing: To add a showing to the current schedule (1)
- Remove showing: To remove a showing from the current schedule if canceled (3)
- Change seating mode: To change the rules for what seats are/are not available (2)
- Modify ticket prices: To change ticket prices for specific seats/rooms (2)
- Login/Logout: To allow access to administrative account (4)

Staff (Total: 4):

- View occupancy: To see how full a room is and what seats are left (2)
- Add physical ticket: To add a physically sold ticket to the system (2)

Customer (Total: 12 + ??):

- Purchase ticket: To purchase a unique ticket-to-seat pairing for a given showing (4)
- View occupancy: To see how full a room is and what seats are left (2)
- View seating arrangements: To see where, if anywhere, a specific seating arrangement can be found in the remaining unoccupied seats (2)
- Refund ticket: To refund a ticket for an upcoming showing based on the theater's refund policy (??, *would depend on the refund policy of the theater, may be ignored for the sake of the project, but still worth mentioning*)
- Use ticket: To use the ticket to enter (and subsequently remove it from the list of outstanding tickets) (2)

INFO-C451
System Implementations
Connor Smith

Functional Requirements:

No.	Priority	Description
REQ-1	HIGH	Ticket-to-Seat pairing, the any seat in a theater should be marked with an *identifier* tied to a specific ticket.
REQ-2	HIGH	Two tickets with the same *identifier* should not be issued in the same *time slot*.
REQ-3	MEDIUM	If a given *time slot* is fully occupied, customers should not see the option to select it **or** should be told that that *time slot* is not available.
REQ-4	HIGH	All issued tickets must identifty their associated *time slot* as well as their *identifier*.
REQ-5	HIGH	All showings should be in one of any of the *seating modes*.
REQ-6	MEDIUM	Customers should be able to purchase tickets online or at the location.
REQ-7	LOW	Customers should be able to search a given *time slot* to see what, if any, seating combinations match their needs.
REQ-8	MEDIUM	Vendors should be able to set prices by seat if desired (for use with special seats, such as the front row or upper seating location).
REQ-9	MEDIUM	Vendors should be able to view past and upcoming showings to see how many seats remained/are remaining, as well as which seats those were.

INFO-C451
System Implementations
Connor Smith

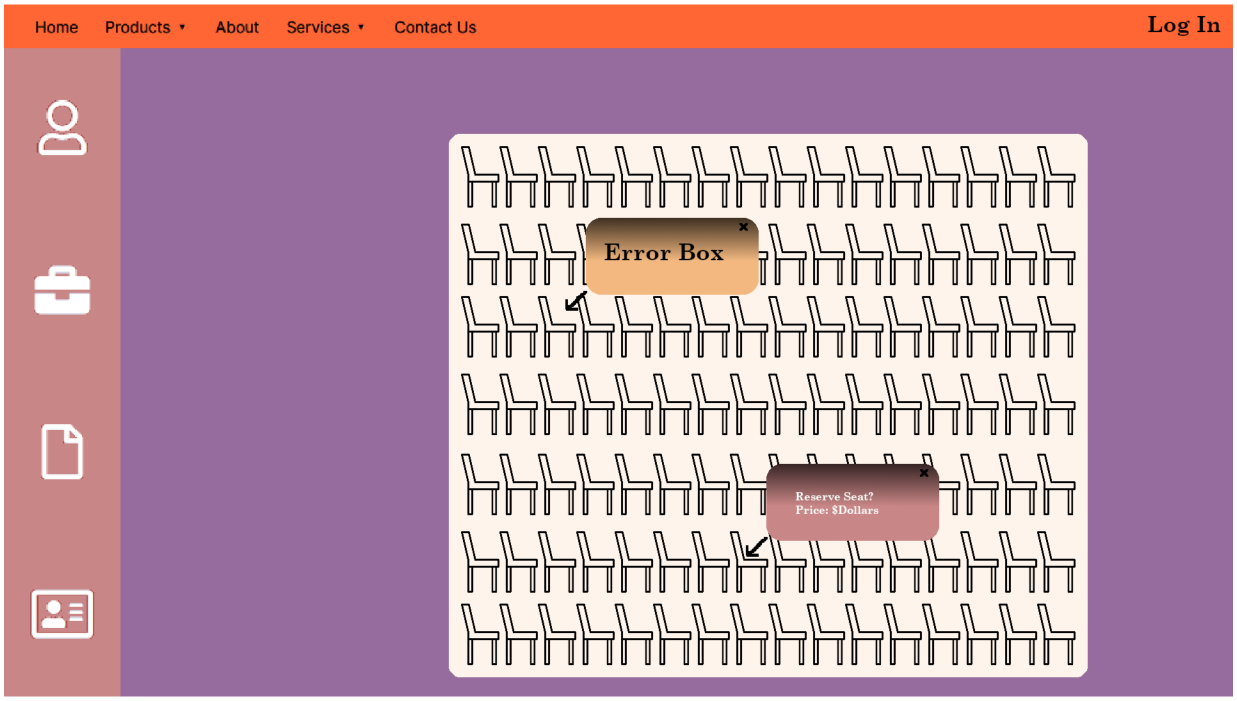
Nonfunctional Requirements:

1. Functionality
   - The system must be feature complete and secure (especially since it will be handling some sensitive information, such as credit cards).
2. Reliability
   - The system needs to avoid errors at all costs, as assigning two people to the same seat could cause issues, and is the service went down during a showing, it could be hard to figure out where everyone should be.
3. Supportability
   - The system needs to be adaptable for the uses of individual locations, as theaters in particular can vary wildly from one another.
4. Usability
   - UX is good, especially for the consumer end, but not as important at the above.
5. Performance
   - While system speed is nice, it is not a priority, as this is a system that sells unique items (id tickets), as as such, the traffic (relative to, say, an online ordering/delivery service) should be relatively slow. Additionally, once set up, little should need to change at a given location, unless major renovations occur.

INFO-C451
System Implementations
Connor Smith

User Interface Specification

Example UI #1, Reserving a Seat:



A very basic potential user-side view of a room.

Already reserved chairs marked (in this case, the orange ones)

Side bar with options related to the room (Account info [like already reserved tickets for this room], check seating arrangements [ex: find a spot with 5 seats in the 2nd row on the edge], change rooms, and check out)

Top navigation bar that should be present on all pages to go from section to section (incorrect text, just grabbed it from another older project, but it gets the point across)

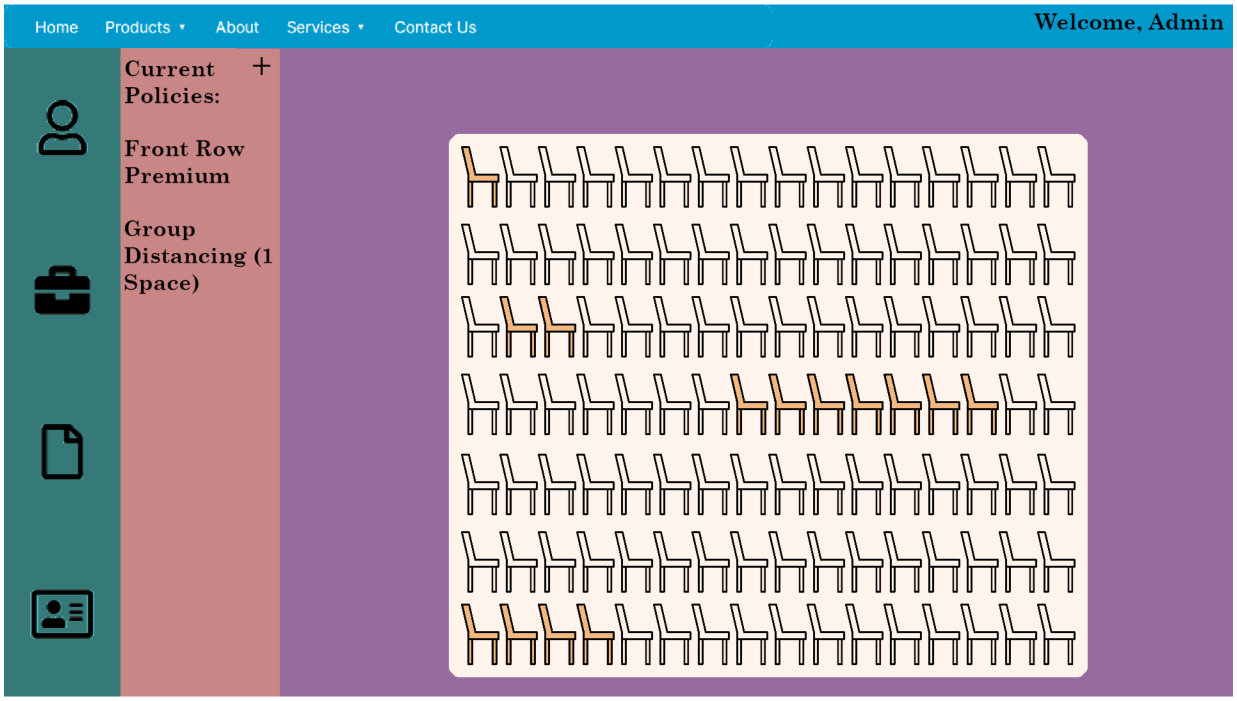
Small confirmation/error windows when a specific seat is clicked on

Pathing: Home → Get Ticket → Choose Your Theater → Choose Your Showing → See Rooms

Total inputs: Depends on if the users already set a theater as a default, if yes, only 4 (ticket, showing, room, select seat). If not, they need to add in searching for a theater, which is an extra click and several keystrokes (to enter the address).

INFO-C451
System Implementations
Connor Smith

Example UI #2, Admin Room Editing:



The same room as above, but from an admin's side.

Same basic functions as a user (minus reserving/checking out for real)

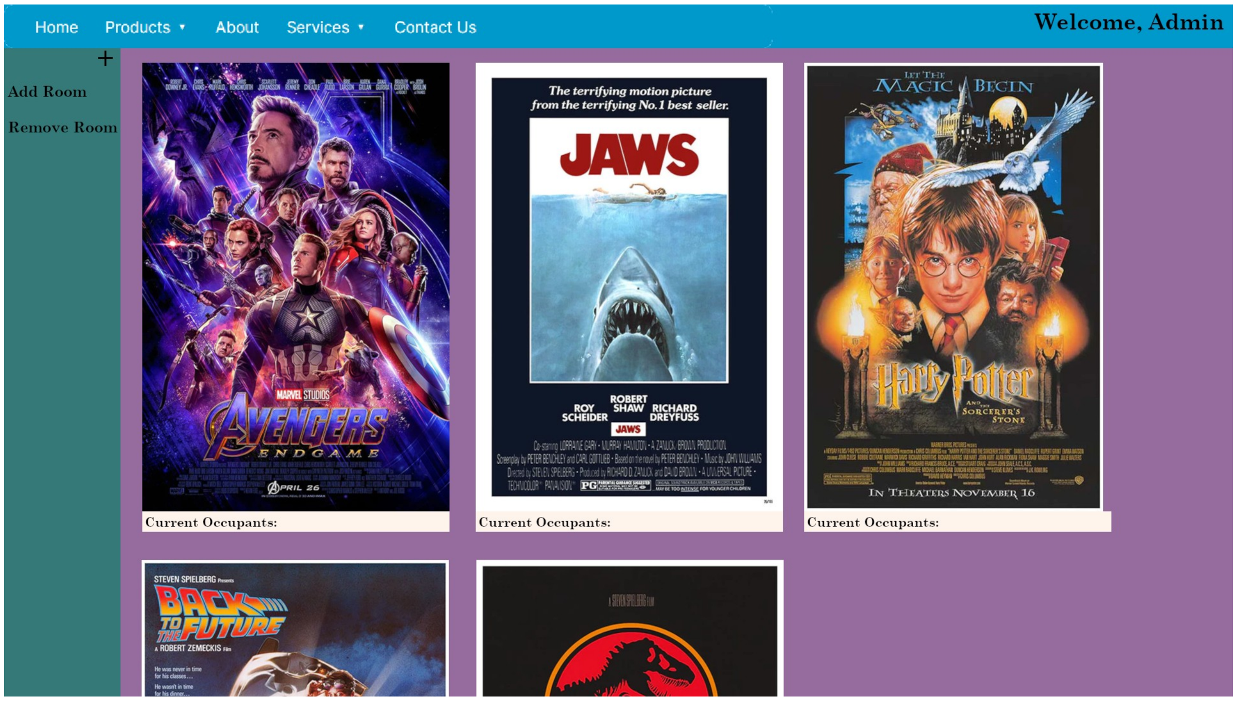
Access to setting room policies/prices.

Pathing: Home → Log-In (for first actions as admin for a session) → Edit Rooms → Select Room

Total inputs: 3 (edit, select, action). 4 if needing to log in.

INFO-C451
System Implementations
Connor Smith

Example UI #3, Admin Room Adding/Removing:



Menu where admins can add/remove/edit rooms as a whole, such as adding a new showing, changing the timeframes of the showings, etc.

Offer option of showing in poster (large) format or by text (list) format. Poster used above since it is more illustrative.

Shows current occupants for a given screening.

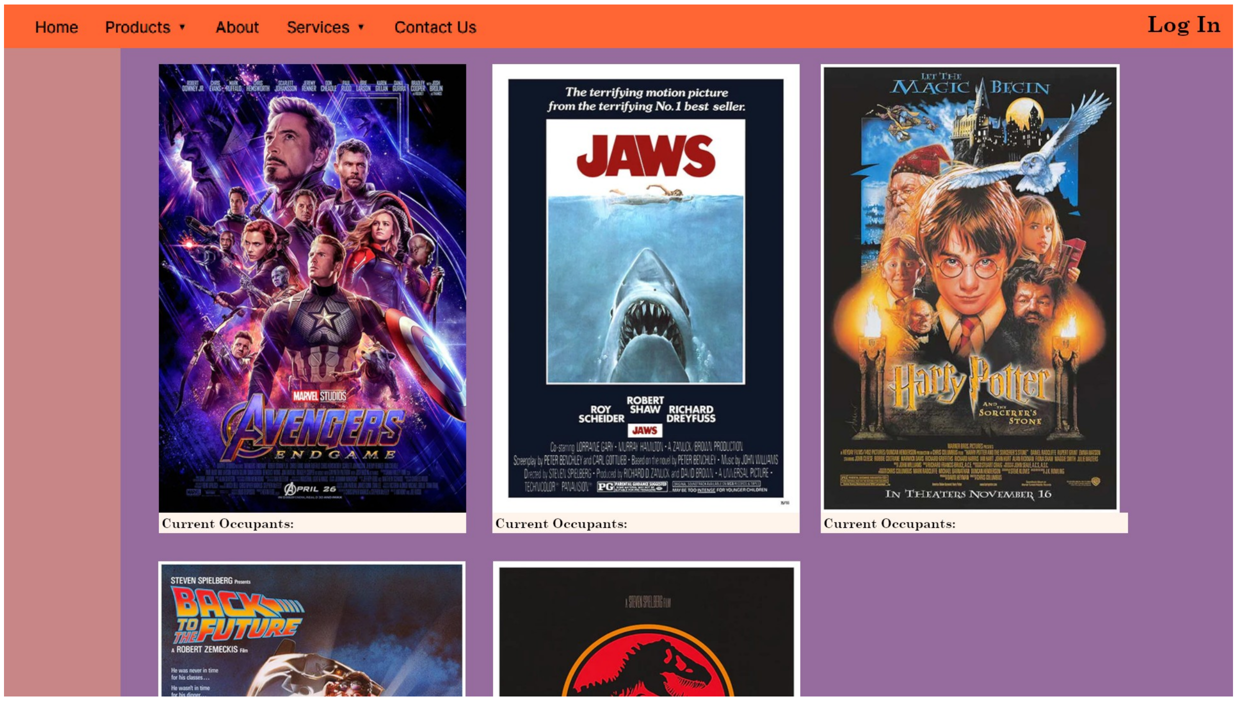
(Should) Warn admins about editing a room that has >0 occupants.

Pathing: Home → View Rooms → Edit

2 Clicks from Home. Actual editing process is more involved and total clicks/keystrokes depends on task.

INFO-C451
System Implementations
Connor Smith

Example UI #4, User Room Selection:



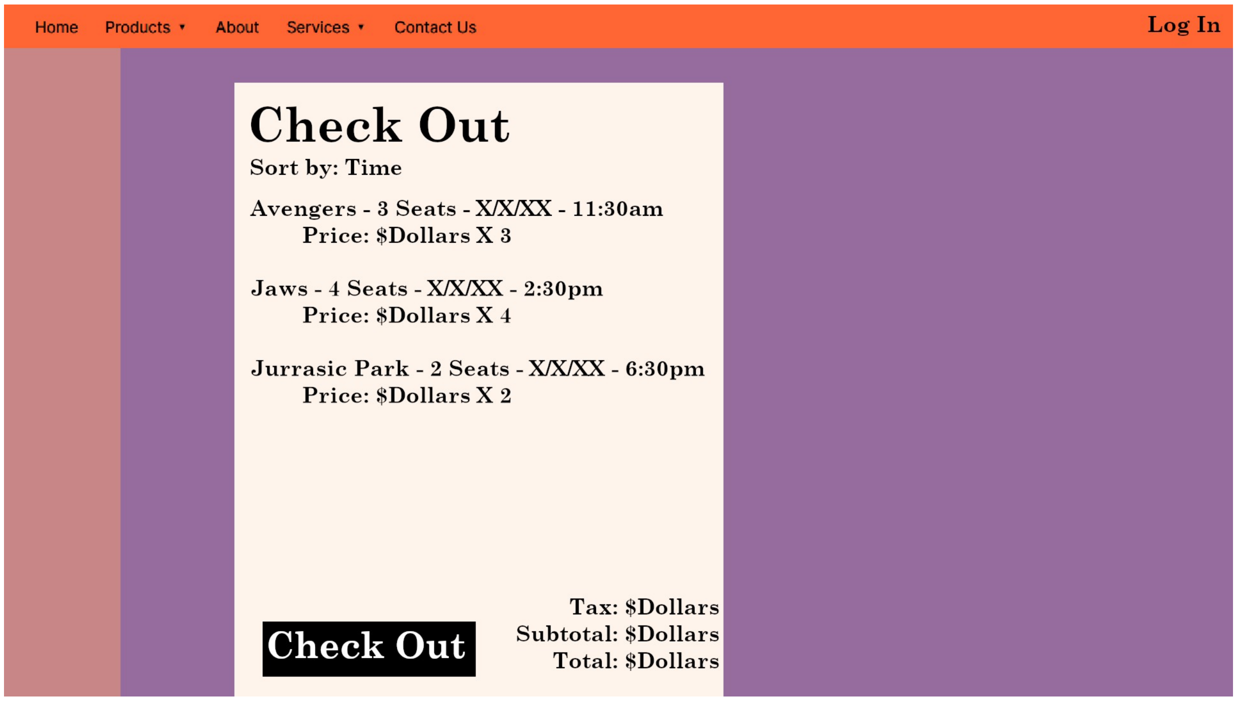
Not the most interesting one, since it's just the admin view minus the edit features, but it is notable for probably being the most symmetric page between the users and admins out of the entire project (since almost nothing is different between them, admins just have the option to edit it).
Even the home page would have more differences (since admins should see very different information than users there).

Pathing: Home → Get Ticket → Choose Your Theater → Choose Your Showing

Total inputs: Depends on if the users already set a theater as a default, if yes, only 2 (ticket, showing). If not, they need to add in searching for a theater, which is an extra click and several keystrokes (to enter the address).

INFO-C451
System Implementations
Connor Smith

Example UI #5, User Check Out:



Check Out
Sort by: Time

Avengers - 3 Seats - X/X/XX - 11:30am
Price: $Dollars X 3

Jaws - 4 Seats - X/X/XX - 2:30pm
Price: $Dollars X 4

Jurrasic Park - 2 Seats - X/X/XX - 6:30pm
Price: $Dollars X 2

Tax: $Dollars
Subtotal: $Dollars
Total: $Dollars

The check out page. While on the surface, pretty typical, there a few things that are intended to be nice for the users here.

Auto-group seats purchased for the same time and showing. You don't need to see 3 separate tickets in the check out, they're just grouped together.

Order sorting. By default, the system is intended to sort purchased tickets by time (the earlier date and time at the top), so it also doubles as a nice calendar.

Sorting options. If you want to view it differently, just choose how its sorted (EX: time, total cost, number of seats, etc)

Pathing: From any section with an item in the cart → Check Out

Total clicks: 1. To actually check out, several more keystrokes (paying isn't optional of course).

User Effort Estimations:

For all the given use cases above, the expected effort required to reach these points should be fairly minimized, both due to the use of drop down menus (for easy access to any given option) as well as the general simplicity of the concept itself. There aren't that many pages you can expect to need in order to buy a ticket to a movie/show. The admins may have more effort required due to the additional steps required to actually edit rooms, especially one at a time, but to actually reach the point where they can edit a given room, it should be pretty straightforward.

INFO-C451
System Implementations
Connor Smith

Updated Timeline and Future Plans

Updated Estimated Timeline:

Part 1:

~~Week 1: Prepare database and code structure.~~

~~Week 2: Finish tasks begun in week 1.~~

Week 3: Begin setting up the administrator side of the system.

Week 4: Implement registration of locations and their managers, as well as the ability to group locations under one 'central' administrator.

~~Week 5: Implement the ability to edit rooms (number of seats, special rules, etc) and set screenings by room.~~

Week 6: Implement time slots in rooms, allow managers to set the times directly or auto-generate based on rules (play-time, clean up time, breaks, etc).

~~Week 7: Polish currently implemented features, conduct preliminary tests.~~

Week 8: Record demo for mid-term.

Part 2:

Week 9: Use feedback to improve current features and identify if additional new features should be added. → **Reattempt to finish weeks 3 and 4 tasks that weren't completed**

Week 10: Finish tasks in week 9

Week 11: Begin work on the customer end of the system (purchasing tickets, payment systems, checking for specific seating arrangements, etc).

Week 12: Continue week 11 tasks and finish user interface.

Week 13: Begin final tests of implemented features, iron out any errors.

Week 14: Final touch-ups and testing.

Week 15: Record demo for final.

Future Plans:

In order to limit the scope of the project, for now rooms are limited to being rectangular in seating arrangements, that is, any given row or column has the name number of seats as any other. As the project progresses, if possible, this may be improved upon to allow unusually arranged rooms to be entered into the system as well.

The tasks laid out for weeks 3 and 4 turned out to be unfeasible to complete at the time they were attempted, as later tasks (primarily week 5) had yet to be fully realized and, upon completion, rendered prior work unsatisfactory for this project. Rather than attempting to "fix" it at this time, it was deemed best to use time pre-allocated in the schedule for such a situation and move the completion of these features to weeks 9 and 10, where it could be implemented after the features it would use have been implemented.

Additionally, the original timeline didn't really specify when the user interface would be completed, and as such, this has been rectified and placed in weeks 12 and 13, as it would be best to make sure all functional requirements are met before working on displaying it in a user-friendly manner. For now, a modified HTML template from W3Schools has been placed in the project as both a placeholder and a more concrete example of how the final result should look. This template is not integrated with the rest of the system and is only to show a proposed visual style for the user interface that is generally cleaner than the mockups.

Non-special room types (Standard and Priority for the non-special seats) are functionally done, as they only had to keep track of how many tickets have been sold and not sell tickets they can't; these rooms do not interact with users beyond that. The special room types (the special Priority seats, Assigned, and Restricted) are not completely functional yet, but should be after the tasks in week 11.

The most major task that needs to be completed at this point is to finalize the integration between the Java end and the MySQL end of the program. At the moment, the two feel very 'separate' for lack of a better word. Due to the rewriting of the Java portion of the code, many classes currently don't use the database, and instead function more like traditional Java-only implementations of the same thing. It shouldn't take too long to fix.

The other task that needs fixing is the implementation of time slots back into the program. The old implementation of them left a lot to be desired, as it was clunky and relied on using the Java Calendar class and making each showing keep track of the time until it aired, and the rewrite broke them completely, so they have been removed from this demo.