`

BSc (Hons) Computer Science

Developing a Research Supported, Cinema Ticket Booking System

Computing Honours Project (COMP10034)

Connor Dillon
B00355490

[01/03/2022]

Supervisor: Henry Hunter

Declaration

This dissertation is submitted in partial fulfilment of the requirements for the degree of BSc (Hons) Computer Science (Honours) in the University of the West of Scotland.

I declare that this dissertation embodies the results of my own work and that it has been composed by myself. Following normal academic conventions, I have made due acknowledgement to the work of others.

Name:

CONNOR DILLON

Signature:

[illegible]

Date:

27/03/2022

Table of Contents

Contents

1.0 Introduction

1.1 Abstract

The development of this application and paper intends to determine the most efficient and appropriate approaches for constructing a self-service cinema ticket booking system. Suitable technologies will be identified, considered, and compared to other applicable solutions using appropriate research and development methodologies. Literature reviews and test results will be used to support technologies that appear or are recommended in the final application.

1.2 Introduction

A cinema ticket booking system will be designed and developed, using the most up to date and efficient technologies. The ticket booking system must provide the most basic functions that a consumer may want to perform, such as browsing movies showing at the cinema and creating a reservation for the showing. When designing a consumer-facing system, the platform on which the application is deployed is a critical consideration. The system must reach the majority of the intended audience without needing extra devices and software to ensure additional complexity is not created.

Concerning a cinema ticket booking system, a self-service system would allow the consumer to feel in control of the process and have business-facing benefits. In a report published by Kingshott, Ueno and Sharma in 2021, two field surveys were conducted: one field survey in Australia consisting of 474 participants and a second in the United Kingdom consisting of 313 participants. Participants were questioned about their experience in using self-service technology within supermarkets. Results of the survey showed that users perceived higher customer satisfaction, a greater quality of service and, in the future, would interact with the self-service technology again. This is just one of many contributing reasons why a cinema would create and deploy a self-service cinema ticket booking system, due to the customer perception of the interactions typically being a positive one.

Two widely recognised research approaches will be used to undertake the necessary research for the prototype: quantitative and qualitative, which relate to analysis using various techniques such as surveys, interviews, analysis of existing systems, and market research.

1.3 Aims and Objectives

During the development, key objectives have been created, illustrating what the application aims to achieve. The aims are as follows:

- Investigate the perceived experience customers have when using self-service systems.
- Create a simple and intuitive user interface.
- Select the most appropriate technologies for the application.
- Create a simple and robust database.
- Create a secure login system.
- Create an application that can interact and modify the database.
- Generate and integrate quick response (QR) codes.
- Integrate a third-party Application Programming Interface (API).
- The prototype is suitable for the use case for which it has been developed.

Objectives that have also been established to ensure that the key aims are met are as follows:

- Investigate the perceived customers' experience when using self-service systems through literature reviews.
- Develop a simple and intuitive user interface supported through iterations using Nielsen's Top Ten Heuristics.
- Identify, contrast, and compare possible technologies suitable for the application.
- Design, create and secure the application's database.
- Investigate and implement a secure login supported with existing literature.
- Create an application that can perform create, read, update, and delete (CRUD) functions on the cinema database.
- Research the most efficient and suitable way to integrate QR codes.
- Explore the possibilities of where the use of a third-party (API) may be suitable and enhance the final application.
- Adopt and integrate an agile development methodology.
- Develop a robust prototype through vigorous testing.

1.4 Justification

During the time of writing this dissertation, Coronavirus is a worldwide pandemic. Many medical advances have been made to control the spread of the virus and reduce severe symptoms through vaccinations and social gathering restrictions to reduce public exposure to the virus. Whilst the world moves into a phase of returning to everyday life, a new variant of the virus, Omicron, is proven to be less deadly than the previous variant. Some individuals are still avoiding social events with a large number of people, whilst others feel that they need to move back to some form of normality and enjoy the activities that they previously indulged in before the lockdown in the United Kingdom.

Cinemas are one of these activities that particular people are returning to whilst attempting to avoid human contact. Whilst visiting the cinema, it is almost impossible to avoid contact with other humans even if it is second-hand interaction, therefore, a self-service system would undoubtedly reduce people's exposure to other individuals. Whilst avoiding the cinema may be the most effective way to ensure you do not contract the virus, some people are willing to visit such establishments whilst still mitigating as many risks as possible. Current consumer habits can support this: 33 per cent of 500 participants used self-service technology weekly or daily, which increased to 43 per cent during the pandemic (Wang, Wong and Yuen, 2021, p.8574). We can hypothesize that consumers feel safer interacting with self-service technology during the pandemic due to their decreased interaction with other people who may unknowingly carry the virus.

Another factor that is a lead contributor to the development of this application is the number of households that have access to the internet. Creating such a system would allow the company to capture a larger audience that may not have been reachable through alternative advertising techniques such as flyers and television adverts. It is estimated that in Great Britain, 95 per cent of households has access to the internet, according to a study performed in 2019, published by the Office for National Statistics.

Lastly, from a business perspective, there are many impacts that self-service systems provide, some of which are more obvious than others. Due to self-service systems reducing the need for service staff, in this case for ticket booking, businesses can save money on the number of employees they must employ. Another less obvious benefit that can be taken advantage of is the greater perceived quality of service that a customer experiences when interacting with self-service systems opposed to when dealing with a store employee, which would be expected to result in repeat visits and satisfied customers.

1.5 Ethical Considerations

During the application development, two ethical considerations must be respected. During the testing or integration of any third-party API, any usage policy that the provider has in place must be respected, which typically advise on the maximum requests per minute that a consumer may make to the API service. Failure to do so may result in the application being banned from the service, being rate limited, or leading to legal issues in severe cases. Lastly, customer data must be secured and protected to a high standard to prevent data breaches and possibly legal penalties.

1.6 Report Structure

A high-level overview of this report can be broken down into seven main sections. Section one is the introduction, which will focus on the reasoning and high-level idea of what the final application will provide. The second portion will focus on performing research into technologies, methodologies, and ideas, all of which will be compared with supporting literature. Thirdly, this report will examine the requirements to ensure that the prototype functions with all the use cases established. This section will also look at the application's design, the database and the security design around the application. Next will be the development process, where the application and all application vectors are created and implemented. The fifth will be the evaluation, where the application is tested and re-developed through an iterative process. Sixth, recommendations and a conclusion will be arrived at and written up. Lastly, a critical appraisal will be created to analyse and list what went well during the development of the application and what would, and could, be improved upon in future iterations of the application.

2.0 Background/Literature Review

2.1 The Effect of Perceived Consumer Experience

As previously discussed, in a study conducted by Sharma, Ueno and Kingshott, published in 2021, it was found that a significant proportion of British and Australian survey participants prefer to use self-service technology. It was also found that the consumer perception of such technology was of an overall improved customer experience. Supporting this, as found by Wang, Wong and Yuen in 2021, many consumers are selecting to interact preferably with the self-service technology during the pandemic. Considering these two findings, we can expect a self-service system to have an increased number of customers who, on average, are receiving a better customer experience.

2.2 Self-Service Effect on Employees

Being burned out is thought to be more common in customer service professions, although it is not unique to this type of work (Maslach and Zimbardo, 2015). Burnout can be experienced through any of the following feelings when felt to extremes: “emotional exhaustion, depersonalization, and reduced personal accomplishment”, typically found when working with people in some manner (Maslach and Zimbardo, 2015).

In a cinema environment, most employees will be expected to deal with customer enquiries and uphold the best customer service possible. By introducing self-service technology, we can assume that this will allow employees to reduce the frequency of their customer interactions, allowing them to focus on the more complicated customer inquiries that are not possible through self-service technology. Due to this reduced amount of consumer interaction, it can also be assumed that employees will display fewer signs of being burnt out, which will positively impact the customers, workers, and the work environment.

2.3 Existing Systems

Major corporations in the cinema industry have already integrated self-service cinema ticket booking technology, which can be seen on their websites and, most of the time, in person. It can be assumed that these corporations have invested large sums of money into investigating the benefits of self-service systems and in the design and development of their own. Analysis of existing systems in the report will compare and identify what the top four leading cinemas self-service systems have in common and attempt to understand the reasoning behind their design decisions to implement aspects into the prototype.

The report will identify the top competitors by how many cinema locations they own worldwide. First is the American Multi-Cinema (AMC), with 1004 locations (Reuters.com, 2022). The second largest is Cineworld, with 787 locations (Reuters.com, 2022). Cinemark comes in third place with 521 locations (Reuters.com., 2022). Lastly, the fourth largest in the world is Cinepolis operating 360 under the name Cinepolis (Cinepolisindia.com, 2022).

2.4 Importance of Security

During the development of almost any application, security is considered one of the most critical aspects, especially when dealing with any customer information. Due to the nature of a database, being the location where any sensitive or personal details are stored, we can assume that this would be one of the many attack targets hackers would attempt to access. In the prototype, we will look at the best practices recommended by International Business Machines Corporation (IBM), a worldwide recognised leader in computer systems. When implementing and evaluating security measures, this report will focus on the eight aspects highlighted by IBM, displayed in the following table, with possible solutions the prototype may implement or recommend carrying out in system deployment.

Security	Solution	Comment
Physical security	Put the database in a secure location.	As this application will be a prototype, a physical database storage unit will be discussed and advised.
Administrative access control	Employees will have the lowest number of permissions possible, whilst still allowing them to do their job.	An access control list should be created to ensure that employees have access to only the controls needed to perform their work.
End user account/device security	All devices should be secure when accessing the database. User logs should be audited and stored for review.	Client device security will be discussed in theory. User logs should be automatically stored and protected.
Encryption	Passwords and sensitive data should be encrypted.	Password hashing algorithms such as bcrypt or SHA256 will be implemented upon analysis, as well as a hashing Salt.
Database software security	Up-to-date DBMS software.	Patches and updates should be applied to a DBMS as soon as possible.
Application/web server security	Security testing. User input sanitization.	The most common area of attack, all user input, should be sanitized to prevent rogue SQL being run on the database. (SQL Injection)
Backup security	Backups are protected and encrypted.	Backups should be run regularly and stored in a secure location. Backup files should also be encrypted.
Auditing	Record all logins and all operations performed.	All operations and logins should be automatically logged and protected. Audits on these logs should be performed regularly.

Table1: Database Security: An Essential Guid - Best Practices

(IBM.com, 2019)

Database security is crucial to any successful company. As noted by IBM, failure to do so may result in “compromised intellectual property, damage to brand reputation, business continuity (or lack thereof), fines or penalties for non-compliance, and lastly listed is the costs of repairing breaches and contacting customers” (IMB.com, 2019).

Multiple standard security procedures are available when protecting sensitive information in a database, such as encryption. Encryption can lead to a reduction in performance. According to a paper created by Ulf T. Mattsson and published by Protegrity Corp, it is stated that “An important step in policy definition is identifying the data that needs to be protected and the data that can remain in cleartext. Many organisations skip this indispensable step and encrypt more data than required. Although it may seem harmless and provide added protection, encrypting more data than needed will come with a price paid in performance”. In order to balance security without hindering performance unnecessarily, sensitive data should be recognised and secured using the appropriate encryption levels.

Selecting the appropriate hashing algorithm for this data is also a consideration. Advanced Encryption Standard (AES) with a key size equal to or greater than 128 bits is used by the US Government for secret, unclassified information. AES keys with a size equal to 256 bits are used for top-secret, classified information (Chen, 2022). Whilst this type of cryptographic encryption has never been broken, it is resource heavy and overkill for individuals’ information the cinema ticket booking will be storing, such as passwords. Narrowing down the appropriate algorithm to MD5 or Bcrypt has its benefits. MD5 is considered faster, while Bcrypt is much slower, exponentially increasing an attacker’s time to brute-force user passwords (Brumen, 2019). In this case, we will opt to use the Bcrypt algorithm due to its security advantages.

To select appropriate security techniques, understanding how an attacker would attempt to decrypt a hashed password must be understood. Hackers will typically take advantage of hash tables and rainbow tables, making decryption of unsalted passwords easier by using already decrypted passwords tables to identify matches. To prevent or discourage these attempts to decrypt passwords if the database is compromised, we will apply a randomly generated salt to each password in the database. The chosen method of salting and hashing users’ passwords before storing them in the database is illustrated in Figure 1.

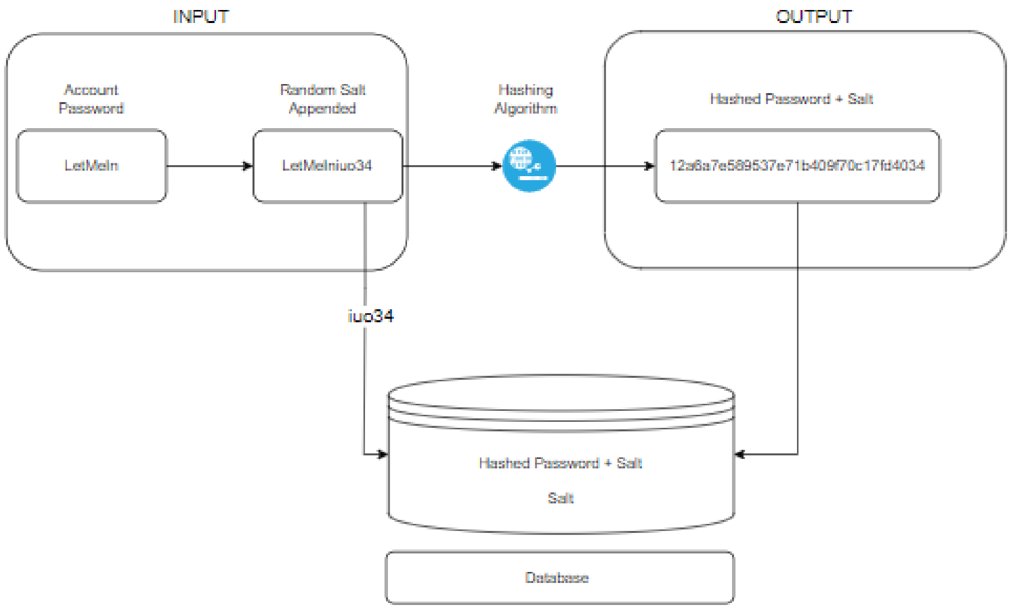


Figure 1: Password Encryption

2.5 Possible Development Frameworks and Languages

A key contributor to the system's efficiency, reliability, and accessibility will be determined by selecting suitable frameworks, software, and development languages. Appropriately selecting the best-suited tools, the application will measure and prioritise tools based on the following: accessibility, reliability, and lastly, efficiency. The reasoning behind accessibility being the top priority is that without customers having the ability to access the application easily, the system will only reach a small proportion of its expected audience whilst creating a complex user experience. Secondly, reliability refers to a tool's ability to perform well constantly; this can typically occur when providers carry out regular updates, which will generally include essential security patches. Lastly, tools will be looked at in terms of efficiency. While efficiency is hugely important in the application, without an audience or an online and active system, efficiency is not needed as no users will interact with the system.

Selecting from many primary development languages can be overwhelming; companies and individuals will usually select a language with which they have experience as this allows them to create a system in a reasonable amount of time and complete it to a high-quality standard as they do not need to learn the language as they develop. It is critical to identify potential development languages and compare them, allowing the selection of a predominant development language throughout the application. Whilst efficiency in the consumer's hardware and internet speeds cannot be controlled from the side of the host of the application. The application will focus on providing the most efficient speeds possible.

In a study conducted by Xiaoli Mao in 2018, a string value was sent from the back end to the system's front end using the same hardware with different language frameworks. Results show that when comparing Express, Symfony, Node.js and ASP.NET, the "performance of ASP.NET was almost 100 times better than Node.js and was over 1,000 times better than Symfony" (Mao, 2018, p.24). Investigating the reliability of Microsofts.NET framework, we can conclude that it has been supported since June the 27th, 2016 and currently has no end date announced, which implies that the framework will receive updates for the foreseeable future (Search Product and Services Lifecycle Information - Microsoft Lifecycle, 2022).

Using a software package such as ASP.net allows the prototype to be deployed on a domain which would allow accessibility from a web browser that can be used on mobile phones, computers and other devices, as opposed to creating an application that is unique to Android or IOS which would drastically reduce user’s access. HTML5 is the newest form of HTML, and due to the nature of this development language, most maintained browsers support the newest versions of HTML, creating an ease of access for consumers. The following table illustrates approximations of what market share browsers have worldwide, all of which support HTML5.

Browser	Market Share (%)
Google Chrome (latest version)	63.8
Safari	19.6
Edge	3.99
Firefox	3.91
Samsung Internet	2.85
Opera	2.35

Table 1:

Browser Market Share Worldwide Dec 2020 - Dec 2021 (gs.statcounter.com, 2022)

Another consideration when deciding upon the predominant development language was PHP, an open-source development language. Comparing ASP.NET and PHP, their efficiency will be measured based upon how fast the framework can load and serve the webpage. In a controlled test performed by Mirzoev, the two frameworks were put head-to-head in a controlled environment, and the results concluded that the “two testing sessions revealed that the reported page load speed advantage of ASP.net over PHP was non-existent. Further, it was shown that neither ASP.net nor PHP showed any advantage in overall consistency of speed during the trials” (Mirzoev, 2013). With this result, it can be assumed that either of the two frameworks would be suitable for efficiency.

SQL injection is one of the most common attack methods on a database in terms of attack vectors. SQL injection allows hackers to run unauthorised SQL statements on the database through un-sanitised user inputs, which can lead to retrieving all the data, deleting the database and much more (Merlo, Letarte and Antoniol, 2007). PHP requires alternative frameworks on top of itself, such as CakePHP, CodeIgniter framework (Lancor and Katha, 2013, pp.518 -519), or the developer creates the code in a defensive manner, which could be highly time-consuming. With PHP being open-source and ASP.NET being constantly maintained by Microsoft, which is also open-source, we can conclude that both frameworks are as secure as possible. In addition to this, benefits drawn from the .NET environment include: “type-safety, multilanguage support, optimised memory management, just-in-time (JIT) compilation model that optimises the availability and performance of the application at runtime” (Hasan and Tu, 2003).

Lastly, when comparing the two, accessibility in terms of the consumer point of view would not be affected by choice of framework, with both technologies being similar in our three key identified aspects, and it is concluded that either of the frameworks will be suitable for this application. In this instance, ASP.net will be the selected framework because it provides automated tools, making development more efficient in terms of speed (Mishra, 2022, p.333).

Through selecting Microsoft's ASP.NET, a decision must be made as to how the data is served to the webpages. Microsoft's Model View Controller structure (MVC) or Razor Pages. According to Microsoft's development team, Razor Pages should be used when developing "page-focused scenarios" (Anderson, Brock and Larkin, 2021). Page-focused scenarios include displaying a list of movies on the user's browser. Razor pages should allow for a more productive and easier development life cycle. Opposing this is the MVC structure used in most other cases, such as creating an API or any other more complex tasks. Razor pages will be selected for this application due to their less complex nature and suitability for the project, contributing to less time needed to develop the prototype.

Lastly, to create the application with the Microsoft ASP.NET framework, one of the following supported languages will have to be used: C-Sharp, F-Sharp, or Visual Basic. Dot Net libraries are primarily written in C-Sharp due to maturity instead of F-Sharp and Visual Basic. According to research conducted by North-eastern-University-Seattle, C-Sharp was the fourth most popular in terms of employers requesting developers with experience in the language, which can be attributed to the language's speed and efficiency. Whilst F-Sharp was not present on their list (Eastwood, 2020). An assumption that can be made is that, generally, a more mature language still being supported will have more significant resources and documentation. Due to the extensive documentation and support for C-Sharp, the prototype will use the C-Sharp development language.

2.6 Database Structures and Management Systems

The backbone of the application will be the database, a substantial contributing factor to the application's functionality, response times and massively influential on the planning required. A range of techniques will be adapted to establish the most suitable model for this use case: relational, semi-structured or non-relational.

Primarily and the most notable advantage that a relational database serves opposed to a non-relational is that: it is more reliable, flexible, robust and scalable (Jatana et al, 2012, p.1). A well-established non-relational database provider, MongoDB states that NoSQL databases should be used if the application meets the following criteria:

1. “Fast-paced Agile development”
1. “Storage of structured and semi-structured data”
2. “Huge volumes of data”
3. “Requirements for scale-out architecture”
4. “Modern application paradigms like microservices and real-time streaming”

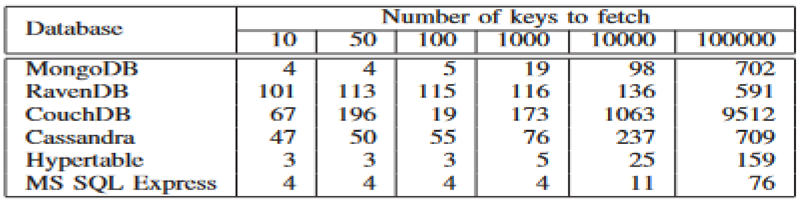
(Schaefer, 2021)

Understanding the advantages and disadvantages of database structures is crucial when selecting a model which will synergise with the prototype. Non-relational databases can be recognised in four main data models: Key-Value database, Column family stores, Document store, and a Graph Database (MongoDB.com, 2021).

According to the corporation and industry leader IBM (International Business Machines), relational databases primarily stand out in terms of flexibility, reduced redundancy, ease of backup and allowing the creator to create meaningful information by joining tables (IBM.com, 2021). A peer-reviewed paper comparing relational and non-relational databases conducted by Jatana, Puri, Ahuja Kathuria, and Gosain demonstrated that non-relational databases provided: Basically Available, Soft state, Eventual consistency (BASE) providing a high availability from the database. Alternatively, SQL databases provide Atomic, Consistent, Isolation, and Durability (ACID) model providing a consistently reliable system. Relational database models have also been recognised for their ability to be queried efficiently and quickly (Jatana et al., 2012, p.1).

Regarding the ticket booking system prototype, all data stored in the database is a string or integer or an expected data type. The only data type that may be a cause for concern and consideration is the ability to store the film's cover image. Understanding and having the ability to expect what sort of data will be stored in the database creates the ability to select the appropriate structure around the type of data the database will be storing. Perceived customer experience can usually be related to the system's consistency, reliability, and efficiency, which can be attributed to the selection of the database model.

Finally, a database that can easily support and represent real-world relations has a considerable advantage in this particular use case; due to a cinema having many relations from bookings to showings. Data from the database will be aggregated and displayed on the self-service system. Due to this, we can expect querying the database to be a regular occurrence, which relational databases excel in. In a study published by Manoharan and Li ini in 2013, Microsoft SQL Express proved the fastest when fetching all keys from the database when comparing alternative database models.



Database	Number of keys to fetch					
	10	50	100	1000	10000	100000
MongoDB	4	4	5	19	98	702
RavenDB	101	113	115	116	136	591
CouchDB	67	196	19	173	1063	9512
Cassandra	47	50	55	76	237	709
Hypertable	3	3	3	5	25	159
MS SQL Express	4	4	4	4	11	76

Table 2:

Time Fetching All Keys (ms)
(Manoharan and Li ini, 2013, p.17)

Relational databases, having the ability to represent real-life relations and efficiently query database information quickly, a relational database is most suited to the use case and will be integrated into the final prototype. To ensure that all the data submitted to the database are expected, a data dictionary will be created, allowing the documentation of data types to be understood before the development stage.

Investigating the most popular database solutions in 2020 voted upon by professional developers, the top five are MySQL, PostgreSQL, Microsoft SQL Server, SQLite and MongoDB, shown in the following Figure 2.

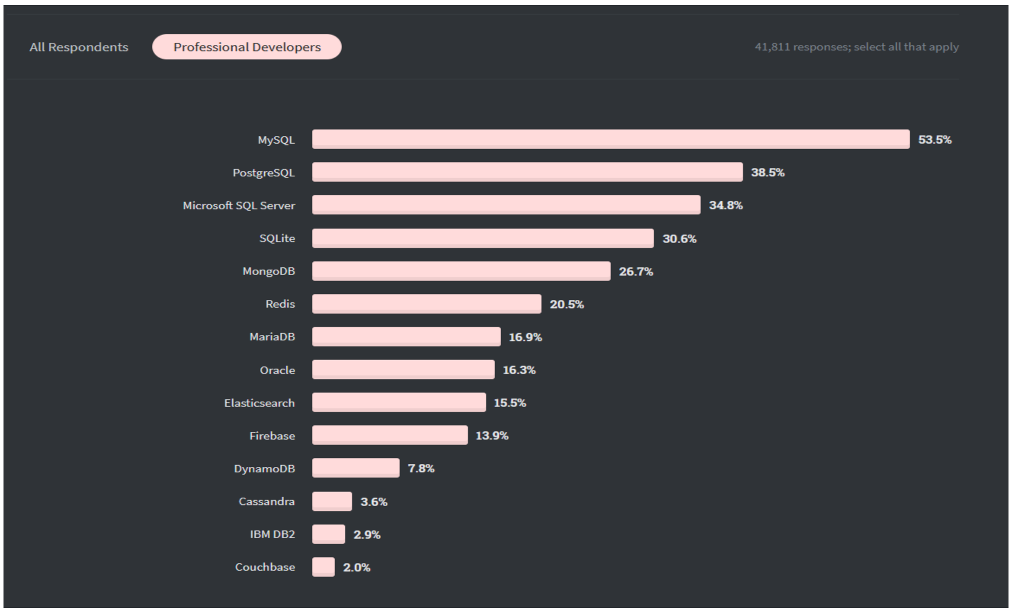


Figure 2:

Database Professional Developers Insight (Stackoverflow.com, 2020)

As previously established, the prototype will make use of a relational database so we can exclude MongoDB as this is a non-relational database type. With such a large proportion of these applications being used by professional developers, it can be assumed that they are selected over other software as they are advantageous in particular ways.

Identified possible database management systems (DBMS) will be compared to select the most appropriate application for the ticket booking system prototype. Areas used to evaluate each DBMS will be the same as previously used, accessibility, reliability, and efficiency.

Whilst comparing DBMS, pricing will play an essential factor in accessibility to any companies that intend to integrate this ticket booking system. Firstly, the MySQL community edition is open source and allows its software to be used at no extra charge in a commercial setting (MySQL.com, 2022). Second is PostgreSQL, which in comparison to MySQL, is the same: it is open source and free to use commercially (PostgreSQL.org, 2022). Microsoft SQL Server 2019, in comparison, is entirely different. Microsoft offers a closed source DBMS, and a fully featured free edition is available but only in a development and test environment. To use this DBMS commercially, the current price for the standard edition is $1,418 per year (Anderson, Brock and Larkin, 2021). Lastly, researching SQLite shows it is similar to MySQL and PostgreSQL, which is open source and free to use commercially. SQLite states, "The code for SQLite is in the public domain and is thus free for use for any purpose, commercial or private" (SQLite.org, 2022). In contrast to the other DBMS, Microsoft's SQL server is slightly less desirable in terms of accessibility due to extra costs and opting to be closed source.

Next will be the investigation of the DBMS selection and what the DBMS has to offer in terms of reliability. MySQL reported that in December 2003, a group named Reasoning "published the results of their Code Quality Inspection Study of MySQL, showing that MySQL code quality was six times better than that of comparable proprietary code. MySQL benefits from the large communities of programmers who "battle test" the code, and MySQL benefits from users who not only report bugs but track down their root cause and fix them" (SQLite.org, 2022). Through these findings, we can expect a high level of reliability from MySQL.

PostgreSQL states that their DBMS "PostgreSQL does everything possible to guarantee reliable operation" (PostgreSQL.org, 2022). This can be seen in their implementation using CRC-32 in WAL files to ensure that record contents are correct (PostgreSQL.org, 2022). Whilst Microsoft SQL server does not boast about its reliability, it can be seen through the constant updates and iterations to the software, leading to fourteen separate updates released for the newest version of Microsoft SQL Server (Ben-Gan I, 2021). SQLite also reports many qualities which can be perceived as reliable such as the following "error reporting paths vigorously tested, SQLite is resilient in the face of corrupt inputs, including maliciously designed database files and SQL strings and SQLite is built using a DO-178B-inspired process. The testing standards for SQLite are among the highest for commercial software" (SQLite.org, 2022). The DO-178B process created by the Radio Technical Commission of Aeronautics Inc. (RTCA) is a standard that is recognised professionally worldwide (Hcltech.com, 2022).

Overall, in terms of reliability all applications would be suitable for this prototype's use case. SQLite is vastly different from the other three contenders in terms of efficiency. SQLite does not require a server to host the database as it is serverless, meaning that the DBMS can write directly from the database onto the disk without the need for an intermediary server process (SQLite.org, 2022). Whilst being serverless comes with its advantages and disadvantages, the main advantage is that there are no additional costs in hosting a server locally or on a cloud service; servers can be costly from configuring, storage, management, setup, and troubleshooting viewpoints.

Due to SQLite performing well in all three critical aspects identified at the beginning of the comparison, SQLite will be integrated to create and access the database as it is most suitable whilst remaining non-complex in terms of opposed DBMS solutions. SQLite also states that "Any program that is able to access the disk is able to use an SQLite database" (SQLite.org, 2022), and this ensures that it will be compatible with its intended use. Whilst SQLite has been selected to implement this prototype, alternative DBMS solutions can easily be incorporated at later stages.

2.7 Technical Details

HTML 5 is the newest iteration of the worldwide standard for designing and creating web pages. As previously noted, all major market shareholders in terms of internet browsers support the HTML 5 language. In a study conducted by the Office Of National Statistics in 2020, it was estimated that in the months January and February 2020, 96 per cent of Great British households had internet access. The prototype is being developed as a web app to ensure that any companies implementing the system can capture and reach a broad audience.

A technical detail that must be highlighted is that, whilst software packages are updated and improved, any corporations implementing this prototype should keep software up to date to ensure that any potential security flaws are patched, preventing data loss or unauthorised access to company and user data. Updating software is essential for any company storing sensitive data.

As previously established, SQLite will be used to create the prototype, which has no fee for being integrated into a corporate setting. Selecting the appropriate DBMS system relies heavily on what stage and size a business is currently in. Whilst smaller companies looking to start up and save on costs may look to use free open-source projects such as SQLite other significant corporations who could already have deals with Microsoft where they have access to all products might opt to use alternative solutions such as Microsoft SQL Server. Technically, it is a simple process to change to a new DBMS system and does not require advanced technical skills or redesigning databases—a consideration for companies looking to change their DBMS. Using SQLite database browser, a free and open-source software package, the user will navigate to a file and export it as an SQL file, which is importable into all major SQL DBMS systems.

Lastly, to further discuss and enforce the reasoning for the encryption method that has been selected. BCrypt is a derivative designed by Niels Provos and David Mazières, which is based upon the Blowfish cipher, which operated iteratively, BCrypt is also an iterative and adaptive function allowing the count of iterations to be increased to make brute-force attacks take longer, or the conductor needs to acquire more processing power to crack the password (Sriramya and Karthika, 2015, p.5552) A brute force attack is a process where a hacker has a list of encrypted passwords and will use a word list and try all possible combinations to attempt to reverse the encryption of the password, resulting in the password being shown in plain text. What can be concluded from this information is that BCrypt is scalable in terms of being slow to decrypt as computer processing becomes more available and affordable, reducing the possibility of information being of any value to the intruder. Whilst encryption is an essential and universal security technique, the security of the database should not be overlooked, such as poor information access control allowing malicious users to access the encrypted information.

2.8 API Scraping

An Application Programming Interface, also known as an API, will be implemented to demonstrate the power of APIs. Application enhancing features that APIs can provide have been considered. APIs will be used to provide QR code generation and movie images. The Open Movie Database API has been integrated, which allows the application to access a wide range of data associated with movies, such as their critic's ratings, release date, awards received and, in our case, an image to display on the application. It is important to note that the API does require a private API key.

Open Move Database provides two crucial types of access: a premium version that is not limited, and a free version limited to 1000 requests daily. As the application is a prototype, the free key is suitable; a paid API key must be acquired in a production environment. The second use of an API is through QRickIt QR code API, which is limited to 9,999 QR codes per day. A consideration that must be accounted for is that any data passed to the API services will be obtainable by the user hosting the service. In this case, it is recommended that these APIs that handle user data are all hosted locally under the same security recommended for the database in the future and production deployment of the application.

2.9 Nielsen's Top Ten Heuristics

Widely recognised and successful Danish human-computer interface researcher Jakob Nielsen established the top ten heuristics that should be considered when designing a user interface. The stated heuristics are stated to be guidelines rather than strict rules (Nielsen, 2020).

The first heuristic is "visibility of system status" (Nielsen, 2020) to ensure that the user can identify what is going on and provide understandable, relevant and appropriate feedback within a reasonable timeframe. The second heuristic is "match between system and the real world" (Nielsen, 2020). Using this as a guideline for selecting terms, concepts, icons and layout, during the development of the system, the developer should have an idea of what their audience is going to expect and understand, such as having similarities to other ticket booking systems. Thirdly, the guideline of "user control and freedom" (Nielsen, 2020). This guideline intends to ensure that the user can always navigate the application when they have potentially miss-clicked and want to return to the previous page. Users should not be triggered to question the definition of words or actions used in the application. “Consistency and standards” (Nielsen, 2020), another heuristic that is outlined referring to a system that follows industry conventions and standards.

Allowing users to troubleshoot through the use of appropriate and understandable errors are essential, "error prevention" (Nielsen, 2020) is another vital part of the user interface, any part of a system that is causing users errors often should be reconsidered and redeveloped to reduce the number of errors users are presented with, creating a fluid experience. Next, Nielsen has noted "recognition rather than recall". This guideline refers to users being able to understand and navigate a system without any prior knowledge, rather than learning the system and recalling this information every time the system is being interacted with. Nielsen also states the importance of "Flexibility and efficiency of use" (Nielsen, 2020); whilst creating a system, it should be recognisable opposed to users having to use recall, shortcuts and advanced actions should also be added for more advanced users, so that interacting with the system becomes more efficient for the regular users.

The eighth heuristic established by Nielsen is "aesthetic and minimalistic design". Interfaces and designs should only include information needed for the user to perform the task they intend on completing. The ninth heuristic guideline is to "help users recognise, diagnose, and recover from errors" (Nielsen, 2020); this is one of Nielsen's top ten heuristics guidelines. Technical and development error codes should be suppressed and formatted understandably to the target audience, allowing users to troubleshoot and correct errors that the target audience can resolve. Lastly, Nielsen states, "help and documentation". This heuristic should be provided with the users with the ability to self-service any issues and find help for any common errors they may face when interacting with the system.

2.10 User Experience

A core component of any successful application is the user experience a user perceives when interacting with a system. Creating a negative user experience typically leads to angry or disgruntled customers not using the application due to the stress or negative emotions it may be triggering in an individual, which can be assumed would reduce the chances of a customer returning to use the service. To ensure that the prototype attempts to establish a positive user experience, it will adopt a user-centred design established by Stone, Jarrett and Woodroffe in their book “User Interface Design and Evaluation”. Three of these main principles are:

1. “An appropriate allocation of function between user and system”
2. “The iteration of design solutions”
3. “Multidisciplinary design teams”

(Stone, Jarrett and Woodroffe, 2021)

Design activities have also been noted which are critical to the development of a user-centred design:

1. “Understand and specify the context of use”
2. “Specify the user and organizational requirements”
3. “Produce design solutions (prototypes)”
4. “Evaluate designs with users against requirements”

(Stone, Jarrett and Woodroffe, 2021)

Following these core concepts for creating a user-centred design will prove extremely important in creating and enforcing a perceived positive user experience. By following these concepts, other developers will be able to develop the prototype with the confidence that the system will meet the user's standards as these concepts are adhered to. HyperText Markup Language version five (HTML5) and Cascading Style Sheet (CSS) validators will be adopted and utilised to ensure that all the code is valid in terms of the industry standard; jigsaw.w3.org provides the validator.

2.11 Research Methodology

In a rapidly growing sector of consumer-facing self-service technology, efficiency in terms of user interface and technology is quickly evolving. Ensuring that the system uses the most appropriate approaches when selecting technologies and interface design is essential, as it is a major contributing factor to an applications’ success. A range of qualitative and quantitative research methodologies will be incorporated to ensure that the appropriate selections are made, all data collected using these techniques will be analysed and used to improve the prototype.

2.11.1 Qualitative techniques

Qualitative research can typically be identified by analysing non-numerical data and the gathering of data. Qualitative research may provide an insight into which parts of the application may need to be revisited and redesigned. During the documentation of qualitative research, observers should not approach the test with a pre-formed hypothesis, as this will lead to a biased result. Two qualitative pieces of research that will be analysed and incorporated are as follows:

Eye-tracking and qualitative page evaluation. In a study conducted by Hernandez and Resnick using eye-tracking and page evaluation, their hypnosis was proved correct. The Guttenberg pattern was the most-effective layout for the website landing page (Hernandez and Resnick, 2013, pp. 1042-1046). Through the results, we can create a user interface around the locations on a screen where the user's eyes are most focused, creating a more fluid experience.

Nielsen's Heuristic evaluation is a qualitative research technique used for identifying and discovering usability problems in the user interface. Through the use of the evaluation, problems can be discovered and addressed without requiring a large number of testers. Nielsen states that each incrementation of tester from zero to five is drastically effective at identifying problems, but after five was not as effective, which can be viewed in the following image (Nielsen, 1994).

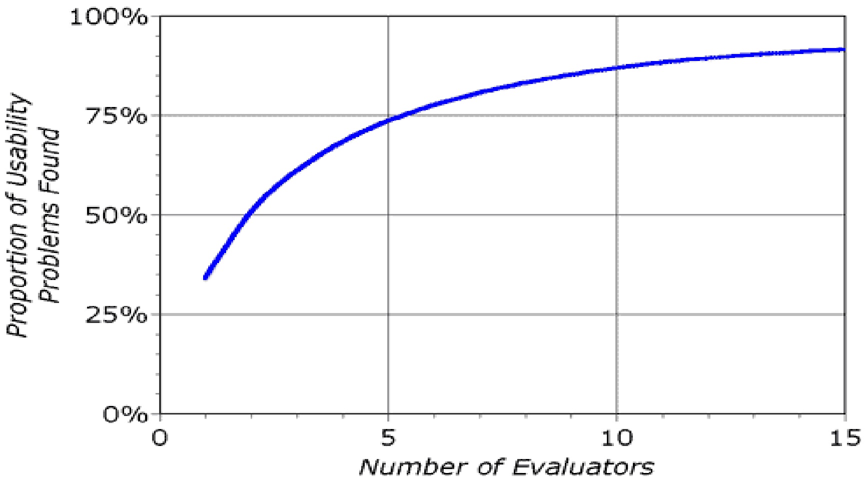


Figure 3:
Problems in an Interface Found by Heuristic Evaluation (Nielsen, 1994)

Nielsen concluded from the research that the following formula could predict the number of usability issues that would be found by adding additional evaluators: ProblemsFound(*i*) = N(1 - (1-l) *i*). Using the information illustrated by Nielsen can be ensured that the testing of the user interface is robust and suitable, whilst not being wasteful in terms of other resources, such as time and money.

2.11.2 Quantitative techniques

Quantitative research techniques will focus upon the amount of analysis conducted and the collection of large amounts of data. Typical characteristics of quantitative data are typical: numeric, percentages, fractions, and statistics. The three most apparent techniques for collecting this sort of data include:

Tracking which can also be described as behavioural analysis. Behavioural analysis is the act of observing a user's actions until particular patterns are identified. Amongst market research, this analysis has proven to be popular. During the development of the prototype, Google Analytics will be integrated to identify patterns in user behaviour, which will help identify any particular areas that the system is causing the customer to abandon the process or where users are having difficulty.

Trends Analysis. Conducting research into what competitors have in common. There is a large array of software packages that can be used to produce large amounts of analysis data that can be of the types: quantitative and qualitative. The tools typically come with a high price of entry. The analysis will be conducted manually for this prototype, allowing the application to use major established competitors' previous research. Resulting in the reduction of planning and preparing research from scratch, it can be assumed that large successful competitors share common entities in their application that are functional and suit the expected user's needs and expectations.

Analytic tools, such as W3.orgs HTML5 and CSS validation free validation service, allows the submission of code and provides the users with statistics on how correct the code is, according to industry standard. Using tools such as these ensure that any other users who wish to develop the project further can be assured that the code they are interacting with and building upon is written as expected. Failing to follow an industry standard could, in future, lead to application failures when software packages are updated and changed. Page Insights is another tool that can be used to analyse the prototype. Page insight allows the analysis of loading times and provides recommendations on how the application could be optimised to load faster, which will positively impact the user experience through fast loading speeds. Whilst a severe consideration, page insights will not be implemented as the application framework, and database structure should provide suitable speeds. In future iterations, page insights will be considered if speeds are not suitable for the application use case.

2.12 Development Approach

During the project development lifecycle of the application, many development methodologies are serious considerations for the application. Methodologies that will be considered are Waterfall, Agile and Spiral models.

Waterfall methodology is a structure that emphasises a structured progression between phases of the development cycle. Each phase can be broken down into activities and deliverables that should be met before moving on to the subsequent key deliverables. The waterfall methodology is sequential (Chandrababu, 2022). Considering this for the development of the prototype has its advantages and disadvantages. Positively, tasks are quickly broken down into subtasks and deliverables, making it easier to ensure that the project is on track to meet the current deadline and an easy-to-understand and straightforward methodology.

An agile development methodology uses an iterative style of development. The Agile development manifesto states that its four fundamental values are individuals and interactions over process and tools, customer collaboration over comprehensive documentation and responding to change over following a plan and customer collaboration over contract negation (Lynn, 2021). An agile development style assumes less risk due to the ability to move back in the process and adjust the application at any given time to rectify potential issues. Scrum meetings are a style of agile development, where small teams meet regularly for short periods of time where a sprint is set, a goal to complete until the next scrum meeting. Whilst assuming less risk, large, complex tasks are broken down into small, manageable tasks.

Lastly, the spiral model will be considered and analysed in terms of how it would help the development of the prototype. Spiral incorporates a top-down and bottom-up methodology. One key benefit of the spiral opposed to the agile and waterfall methods is that the software is produced extremely early in the project development life cycle, as opposed to after the planning stage in the waterfall and agile methods. Negatively, the spiral method is usually taken on by a team of people rather than a sole individual and can be expensive (Kumar Upadhyay, 2020).

Comparing the three methodologies, as the prototype will be designed and created by one developer, the waterfall and agile development methods appear more attractive due to reduce complexity. With a deadline that has also been set, the project will be planned and set out in a GANTT chart (Appendix 10.1.2 - 10.1.3) to ensure that the application is completed in the agreed timeframe, again the waterfall and agile methods complement planning and is less complex to ensure that the project is completed by a specific date. A waterfall method creates a straightforward plan and avoids additional complexity, as no new tasks are assumed until the current one is complete. An agile approach also allows the project to be handled with a waterfall methodology, but also allows developers to return to previous pieces of work and change them if they feel it is more suitable. An agile methodology will be adopted to create the final application to ensure that any issues and queries can be corrected instantly rather than waiting until the testing phase of the application.

3.0 Requirements and Designs

3.1 Introduction

The current chapters will focus on a prototype's requirements to become functional yet easily extendable and customisable. While this section will mainly focus on requirements, the database design and use case diagrams will also be established to ensure that the following development processes meet the prototype's critical necessities. In order to create a prototype that is easily extendable and adaptable by any organisation, the user interface will be designed in a different environment from the prototype to prove how the application may look whilst allowing companies to create their layout and colour schemes.

3.2 Requirements and Features

To create a prototype that is easily extendable and functional, some core requirements have been identified, which are either essential to the system or concepts which may be beneficial to the project in future and final iterations of the complete application; before being deployed in a live setting.

Critical for most systems that involve customer accounts and the ability to either buy or create reservations is the ability for the application to perform Create, Read, Update and Delete functions on a database, also known as supporting CRUD. For the case of the prototype that is currently in development, it will support the full range of CRUD functionality.

3.2.1 Use-case Diagram

Whilst the prototype will support a full range of CRUD functions. It must be established what expected interactions a customer or staff would perform when interacting with the booking system.

A use-case diagram has been created that shows a user's possible interactions with the prototype. Another benefit of the use-case diagram is that a stronger understanding of the expected data types that the database will interact with can be better understood. The use-cases' diagrams and their descriptions are as follows:

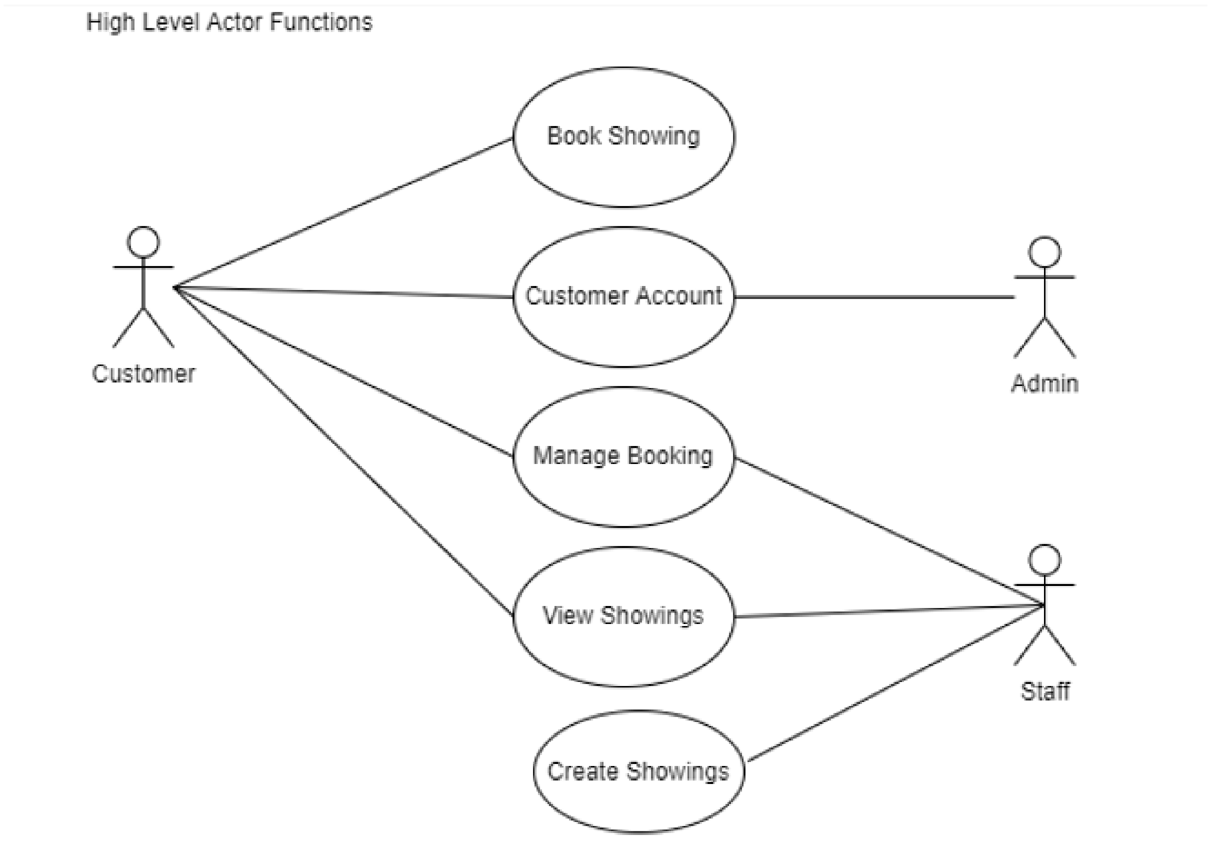


Figure 4: High Level Actions

High-Level Overview - A view shown in Figure 4 that represents all the functions that users are expected to preform when interacting with the system. A customer is expected to book a ticket for a showing, manage their account, manage any bookings they have in their account, and view movies currently showing at the cinema. Secondly, the staff role is the middle-level user in terms of system access. Staff are expected to be able to create showings, view showings and manage bookings that are present in the database. Lastly, is the admin with the highest level of access in the system; the admin should be able to modify and view all accounts that are present in the database.

Customer Booking - The system's core function is built around the customer creating a booking. Figure 5 shows the customer logging in and creating a booking and the total price a customer will incur if they continue with the booking. Staff can cancel all bookings.

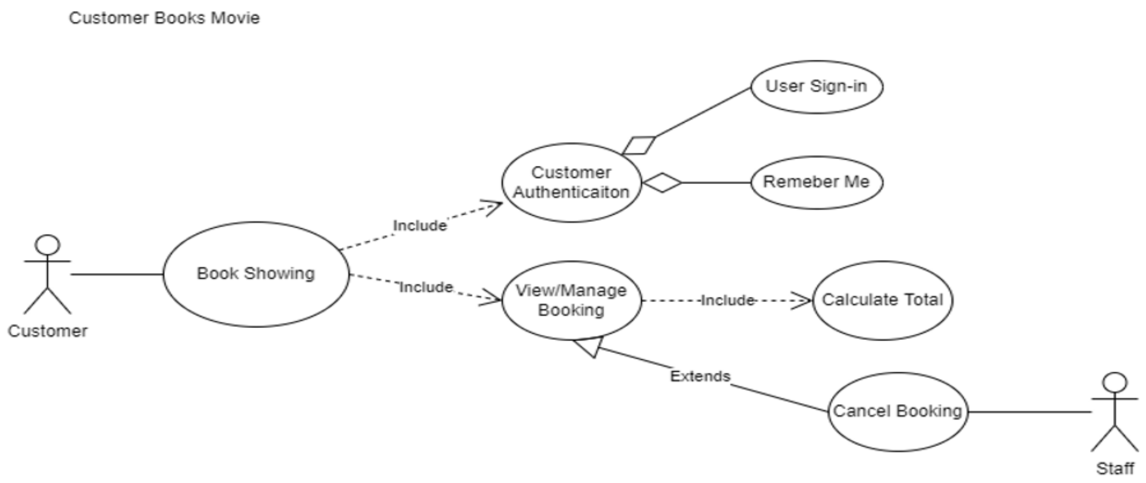


Figure 5: Customer Booking

Creating an Account - Figure 6 represents the process of a customer registering on the website. A customer should be able to create an account and provide their, username, email, password, first name and last name to create their account. Whilst the highest level of access user, the admin, can view all the account details that a user has submitted and be able to reset a user's password.

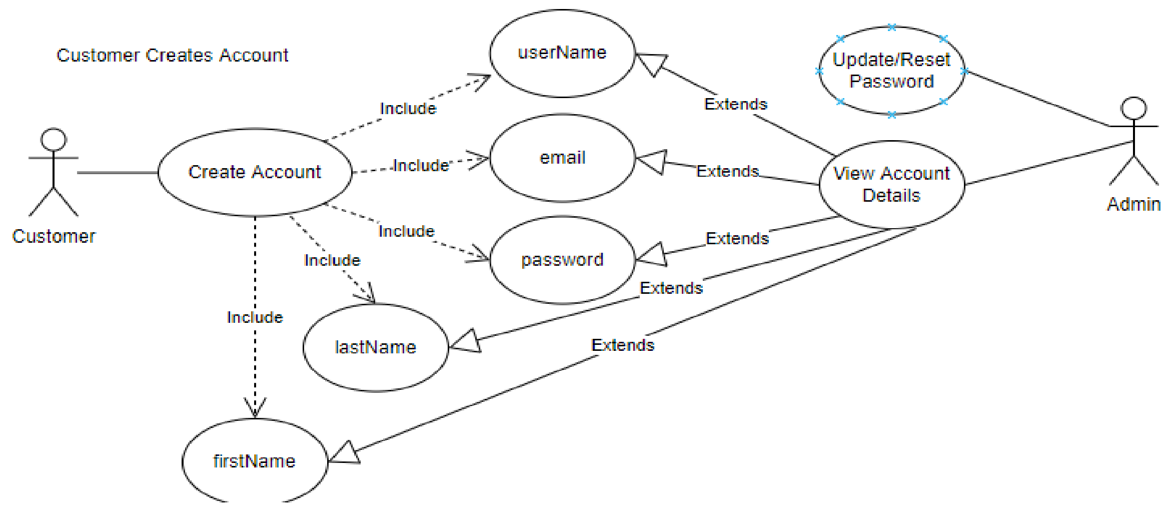


Figure 6: Create Account

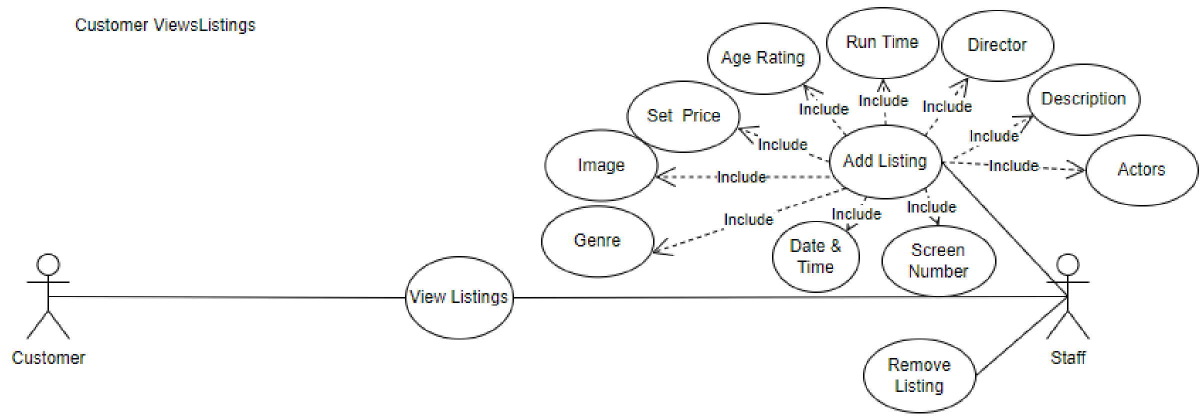


Figure 7: View Listings

View Listings - Customers should be presented with a list of currently showing movies that the staff have added. Staff will be able to create a new movie and add it to the listing of showing movies whilst also adding the attributes that a film should contain, such as the directors and actors, illustrated in Figure 7.

3.2.2 Secure Login

Creating a secure login system is the primary focus of the application. With the prototype storing customer data and personal information, the data should be protected. In order to create a secure login, cookies with appropriate claims and password encryption will be deployed. Using encrypted passwords makes any malicious users who may have obtained viewable database access unable to use the user's stored passwords.

Using cookies and claims also creates the ability for a different level of user access as previously demonstrated in the use-case diagram, there will be three levels of users: SuperUser (Admin), Staff and customers, through the use of cookies and claims page policies will also be deployed in order to prevent unauthorised users accessing services that they should not have access to. Lastly, whilst simple but often overlooked, the input form where a user enters their password will be of the type of password, making the password appear as black dots on the screen rather than readable text, preventing users from shoulder surfing attacks.

3.3 Analysing Existing Systems

Analysing the top four cinema booking web applications will be conducted to take advantage of possible aspects leading to existing competitors' success. Arguably the most crucial application page will be the landing page, which is what a customer is first presented with when accessing the application. In a paper created by Gafni and Dvir, it was found that the application's landing page has a considerable influence on a user's decisions and actions and plays a significant role in forming a buyers' intention (Dvir & Gafni, 2018, pp.19–39).

The number one rated cinema in terms of locations is American Multi-Cinema (AMC). Figure 8 illustrates the landing page that AMC presents their customers when first arriving.

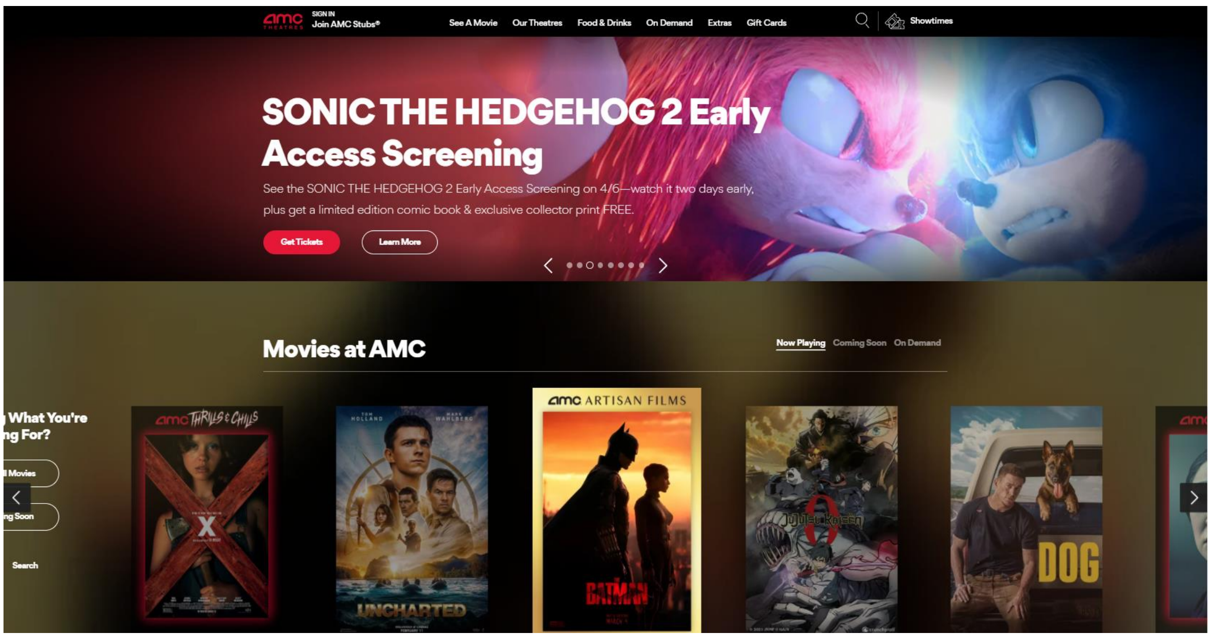


Figure 8: AMC Landing Page

(Amctheatres.com, 2022)

AMC's application required a connection originating from the United States instead of the United Kingdom, which may be a security measure to prevent malicious requests and load from countries that do not have the AMC cinema chain. The user interface layout noticeably has a navigation menu along the top of the application, followed by a promotional carousel which allows the application to run featured adverts. AMC has made a possible significant design decision, placing the current showings movies on the landing page, which may simplify the process for users by reducing the amount of steps/clicks it takes for a user to navigate to book the showing they intend on viewing.

The second largest is Cineworld, with 787 locations (Reuters.com, 2022). The following Figure 9 illustrates the landing page a customer is presented with when accessing the Cineworld application.

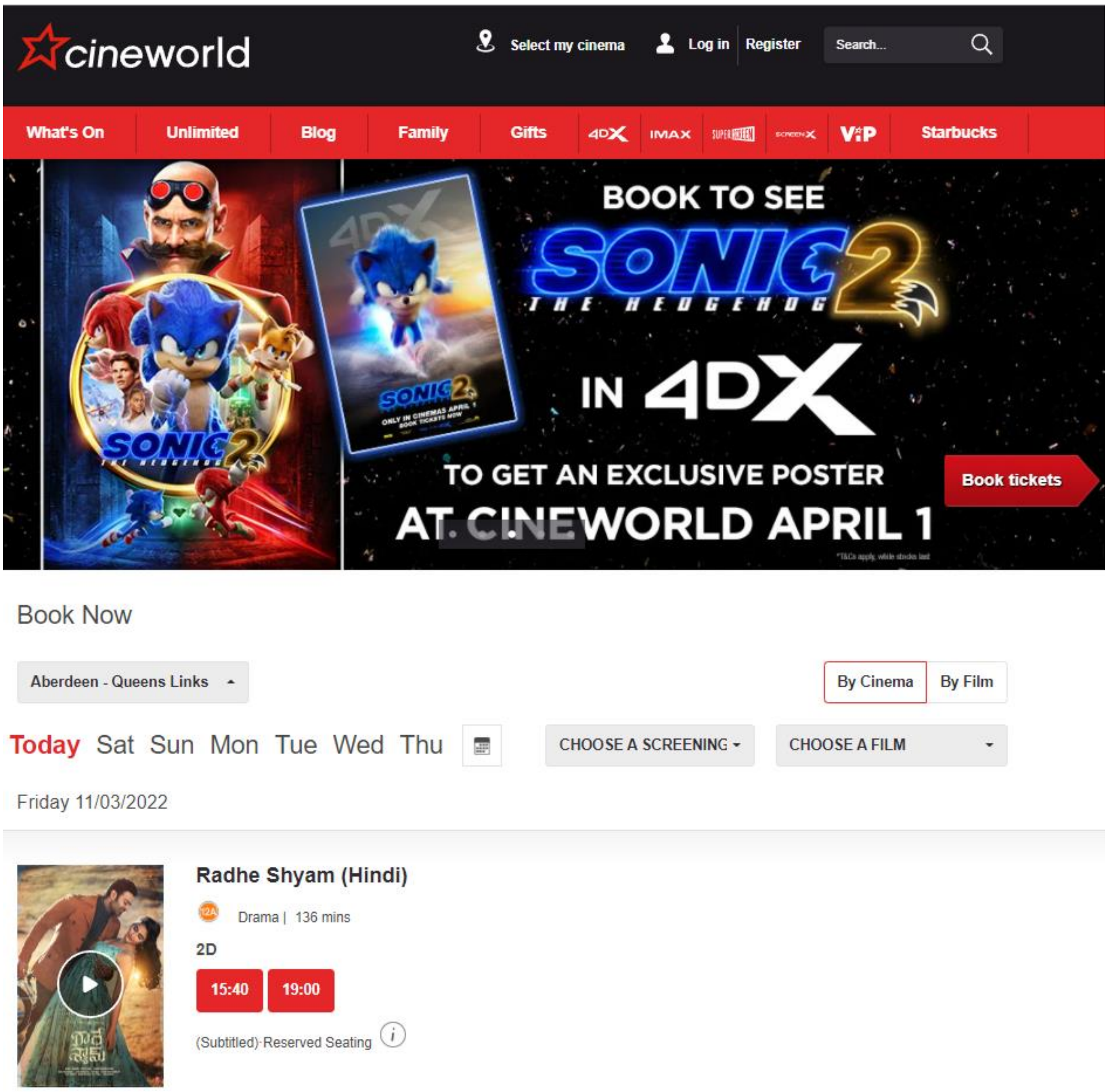


Figure 9: Cineworld Landing Page

(Cineworld.com, 2022)

Cineworld requested the browsers' location when the application was accessed, allowing the most relevant cinema to be automatically selected for the user; this reduced the required amount of steps a customer must perform to book a showing. All application elements are centred down the middle of the webpage in an easy to digest format. It is also shown without the need for navigation all the movies that are showing and the times that they are available, and lastly it is noted that the banner promoting the most popular films

Cinemark comes in third place with a total of 521 locations (Cinemark Holdings, Inc., 2022). Illustrated below Figure 10 is Cinimarks landing page.

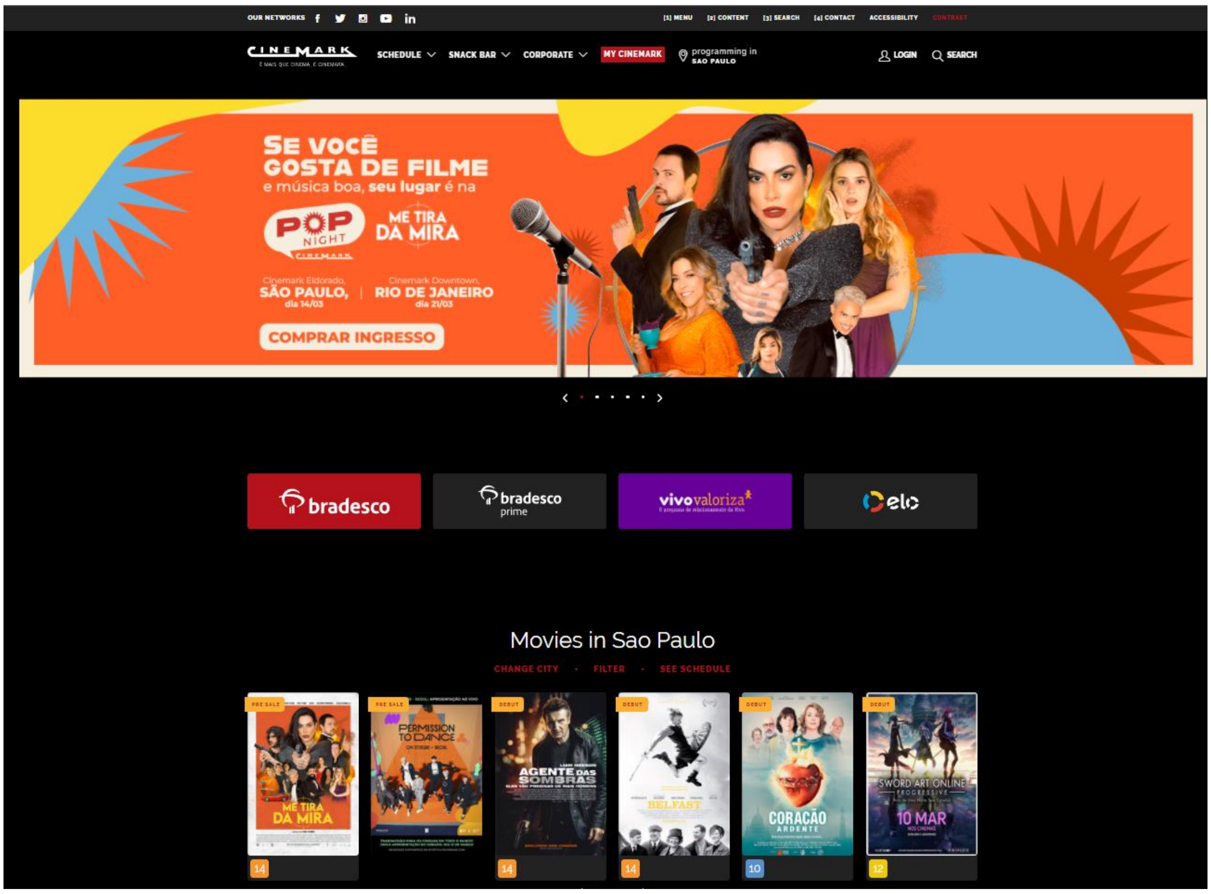


Figure 10: Cinemark Landing Page

(Cinemark.com, 2022)

Cinemark shares similarities in the previously analysed landing page. Notable similarities are that all information is centred on the page, a banner showing cinema promotions is beneath the navigation controls, all film showings can be viewed on the homepage, and lastly, the cinema logo's position is also placed in the upper left of the screen.

Lastly, the fourth largest in the world is Cinepolis operating 360 under the name Cinepolis (cinepolisindia.com, 2022). Illustrated in the following image is the Cinepolis landing page, Figure 11.

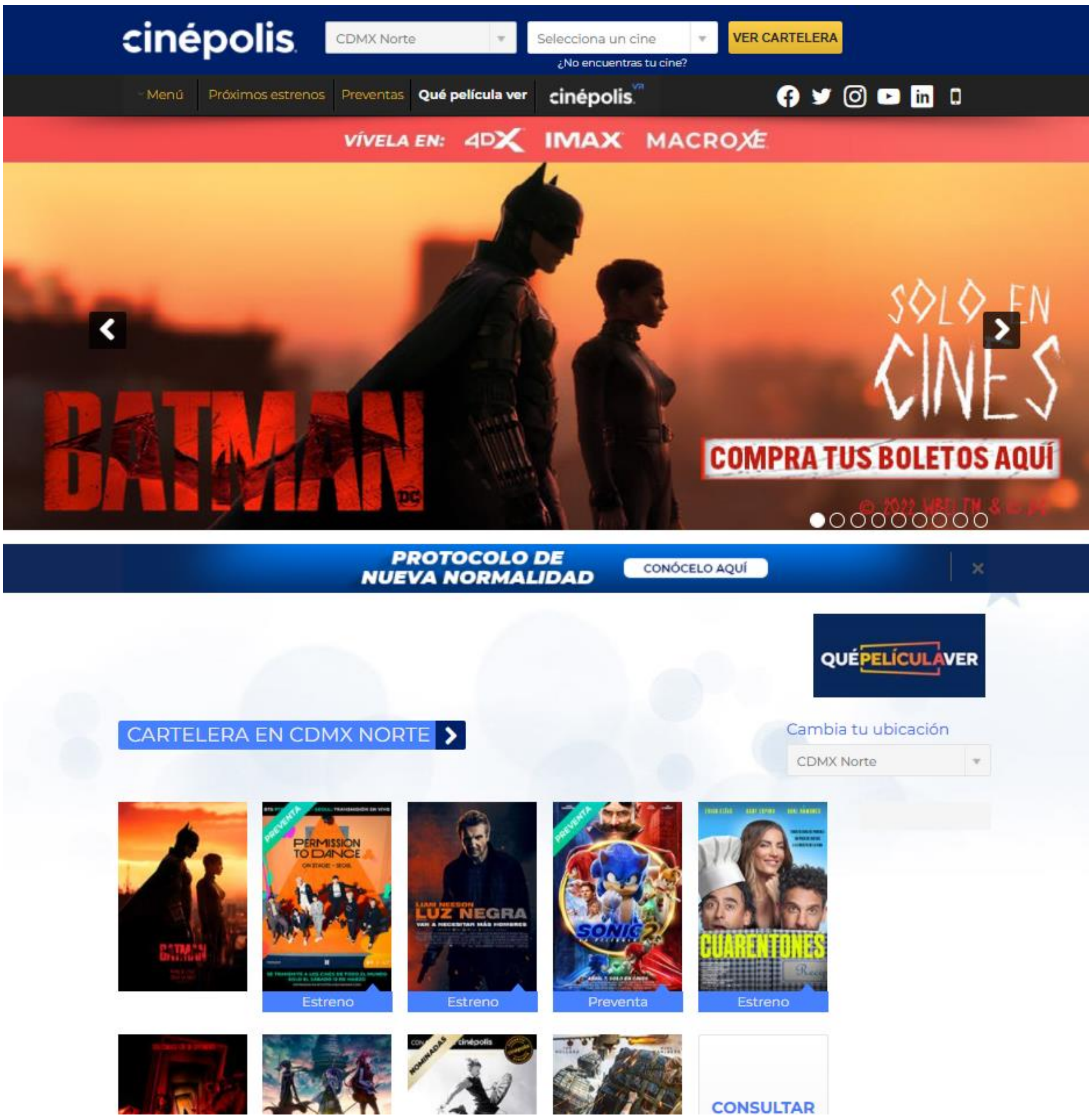


Figure 11: Cinepolis Landing Page

(Cinipolis.com, 2022)

Cinepolis also shares many similarities with the other top cinema competitors. The company's logo is located at the top left of the page. All information is centred on the page. A banner is present at the top to help companies promote films and deals. A navigation bar is found at the top of the page, and lastly, all current showings can be viewed on this page.

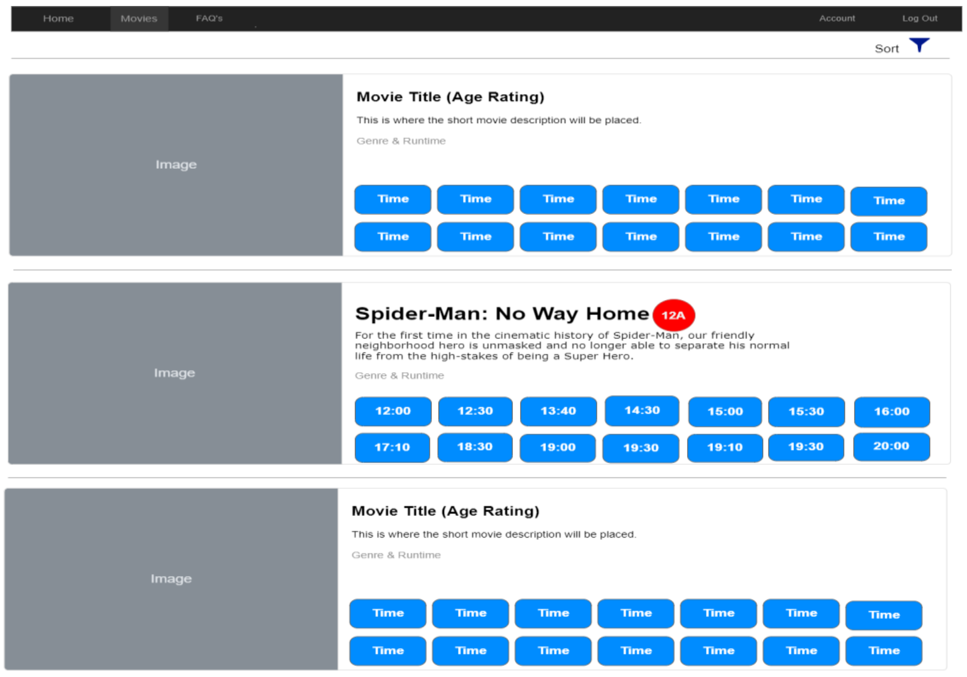
One of Nielsen's top ten heuristics is that "Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform and industry conventions" (Nielsen, 2020). In order to enforce this, the application will follow the similarities identified, such as the following:

1. Logo positioning
2. Navigation bar
3. Information centred
4. Movies showing displayed on landing page

Cinema showing times will also be implemented similarly to Cineworld, as this reduces the steps a customer needs to perform to find out the times a film is showing and creates a more simplistic flow of actions to create a booking lastly, as the application is focusing on a user interface that is simple yet effective. Notable is that a carousel is located at the top of all the major cinema chains applications, which will also be implemented to allow future featured advertisements.

3.4 Application User Interface

Previously, as briefly covered, the Gutenberg pattern will be implemented, as shown in Appendix 1, Figure 4. The Guttenberg pattern suggests that users are most likely to view the upper left of the screen and continue through the centre of the page to the bottom right. In order to take advantage of this, the landing page will be created with this in mind. In a study conducted by Resnick and Hernandez, on average, customers were most likely to add items to their shopping car when integrating the Guttenberg Pattern, and 70% of the content was placed above the fold illustrated in Appendix 1, Figure 5). After careful consideration, the following wireframe illustrated in Figure 12 has been created as a landing page for the prototype.



Movie Title (Age Rating)

This is where the short movie description will be placed.

Genre & Runtime

Spider-Man: No Way Home

For the first time in the cinematic history of Spider-Man, our friendly neighborhood hero is unmasked and no longer able to separate his normal life from the high-stakes of being a Super Hero.

Movie Title (Age Rating)

This is where the short movie description will be placed.

Genre & Runtime

Figure 12: Cinema Ticket System Landing Page

As the prototype will focus on functionality, the design will be created separately for organisations to implement their layout and colour scheme quickly. The previous layout has been designed in a way that is believed and proven to increase customer sales. The Guttenberg pattern suggests that the customer first view the top left, where the navigation controls are located. According to the Gutenberg pattern, the next piece of information that a customer will consume is the carousel, which should be showing new movies and selected adverts.

Next and most importantly, customers can view the movies and showtimes they are interested in viewing. With all this in mind, this landing page is heavily designed around the theory of the Gutenberg pattern, which will lead to a positive and straightforward user experience whilst creating more potential sales for the company.

3.5 Database Design

Critical to the prototype, the backbone of the application, the database must be designed and built in a manner that supports reliability and future expansion whilst supporting all the use-cases established in Figure 4. In order to create an appropriate database, and Enhanced Entity Relationship diagram has been created, shown in Appendix 1 Figure 6. The creation of the model allows the documentation and representation of requirements, relations, and columns in tables. Previously the Entity Relationship diagram was created, which led to the proceeding Enhanced Entity-Relationship model, which is more expansive, allowing a detailed representation of relationships where junction tables are introduced to the structure of the model.

Incorporating a relational database requires the developer to understand and expect the data and types that users submit to the database. A data dictionary has been created to ensure that all the possible data types are accounted for and expected within their defined data types. The data dictionary also allows constraints for columns to be established in advance. The data dictionary can be viewed in Appendix 2. Each entity attribute's data dictionary access settings are all set to public, as sensitive information will be protected using page policies.

3.6 Security

Security is a significant focus in this project. The project will focus upon security in terms of password encryption and hypothetical best practices for any corporations that want to host their iteration of the application in their own controllable environment.

3.6.1 Physical Security

When considering physical security for the database, many users will opt to use cloud services such as Amazon Web Services; in this case, we can expect a corporation of Amazon's size that the protection will be implemented at one of the highest levels. If an organisation wishes to hold their database locally or on their own server, the recommendations are as follows.

The database should be stored in an area where the climate is controllable, where temperatures are monitored and kept cool (IBM.com, 2022). Another physical protection protocol that should be implemented when protecting data, recommended by Carnegie Mellon University, is that all operations where a user intends to obtain physical access to the data centre are formally documented. An already assigned and vetted data steward (Carnegie Mellon, 2022).

3.6.2 Administrative Access Control

Another best practice is establishing access and roles when protecting a database against unauthorised use (IBM.com, 2021). In the case of this prototype, a user access control list will be created, ensuring that only specific users with the appropriate levels of access can interact with particular functions of the database and application. Whilst ensuring access control lists, users should be limited to only the permissions that are needed in order for them to perform their job roles within the company. In this case, there will be three user levels: customers having the lowest level of system access, next will be the staff who will have a more significant level of access allowing them to add movie details and showings to the application and lastly there will be a superuser, who will be able to manage all accounts, the ability to change passwords and lastly, the ability to turn standard customer accounts into staff accounts.

3.6.3 End User Account/Device Security

A recommendation that the prototype will not be implementing is data monitoring solutions; all access should be recorded. Data monitoring tools can recognize irregular patterns in a user's account and what sort of data they are accessing. These tools can be configured to alert others in such an event. Another recommendation that should be implemented if the prototype is deployed in a corporate setting is that any machines connected to the database management system should be secure and configured by the company to ensure that the devices are not compromised (IBM.com, 2021).

3.6.4 Encryption

Encryption is another standard best security practice recommended by IBM (IBM.com, 2021). In terms of security, all user credentials such as passwords will be hashed and salted with the BCrypt hashing algorithm to ensure that passwords and impossible or require massive amounts of computing power to decrypt. Disaster prevention is a standard procedure when dealing with databases. To prevent catastrophic data loss, companies should regularly back up their databases; these database backups should be treated with a high level of security, including the encryption of backups, and storing in a separate location. An alternative to using BCrypt is SHA encryption, SHA encryption may be less suitable for protecting customer passwords as it is known as fast hashing, which means that dependant on the intruders' computer power, millions of guesses can be made per second, whilst BCrypt requires multiple iterations to attempt one password guess (Rietta, 2016).

3.6.5 Database Software Security

Appropriate and up-to-date software has been used to provide the most secure releases; in this case, the latest version of SQLite has been implemented. Whilst this directly affects the database, being the opted solution to use as the DBMS. NET6.0 is also being used as the target framework for the prototype, as it is the most up-to-date version published by Microsoft. It can be assumed that any software that has possible discovered vulnerabilities will be exploited when deployed to a Commercial or non-development setting, which is why software must be regularly updated.

3.6.6 Auditing

All operations performed within the application and database should be logged and protected. In the attempt that any malicious user has access to the database, it can be assumed that they will attempt to reduce the chances of being caught within the system. Logs should be protected at an extremely high level. Auditing these logs should be conducted regularly to ensure that no suspicious activity occurs within the database.

Database auditing can be performed in a multitude of different techniques:

Manual Auditing - Involving SQL queries and manually reading through data logs can become very cumbersome and require large teams as the company and data grow and experience more activity.

SQL Server Triggers - Triggers can all be set up and configured in a way that can easily track a variety of information. Triggers are highly customisable, allowing the database owner to create their own and only log information they believe to be of high value.

Server Transaction Log - The transaction log record everything that occurs in the database, which is extremely useful for auditing. A log of everything can be troublesome to analyse but extremely useful for troubleshooting unexpected and unknown issues.

SQL Profiler and Tracer - An announced depreciated tool that can replay and record events saved to a trace file.

Temporal Tables - Records all history of database changes, which is extremely useful for recovery purposes.

While comparing these different auditing tools, selecting one would not be suitable to cover every possible situation, as each tool has a role designed for. A safe solution would be to incorporate all of the above tools, excluding the Profiler and Tracer, which is becoming a depreciated tool.

3.6.7 Importance of Security

The General Data Protection Regulation (GDPR), which affects all companies that target or collect data from people of Europe, must be considered. The regulation was put into effect on May 15, 2018 (Wolford, 2022). Fines for violating this piece of legislation can be one of two penalties: 20 million euros or four per cent of their global company revenue; which one is higher. A key principle of the legislation is integrity and confidentiality, where it is stated, "Processing must be done in such a way as to ensure appropriate security, integrity, and confidentiality (e.g. by using encryption)" (Wolford, 2022). The failure to follow and enforce the advised steps listed above may lead a company to be guilty of breaching this piece of legislation, whilst heavy fines will be incurred brand damage, and company reputation will be negatively impacted, which could lead to a companies' failure.

4.0 Development

4.1 Introduction

The development section of this document will intend to serve anyone reading how each stage of development was addressed. Development is being conducted using an agile methodology, which allows any tasks to be revisited when appropriate. Agile development can be conducted in multiple flavours; in this case, scrum meetings and sprint targets will be incorporated into the development methodology. Scrums are where small sprint targets are set and worked upon until complete. After the sprint is complete, the process repeats itself until the project is completed. A collection of people typically utilises the scrum framework; in this case, the single developer will adopt it. Tasks are typically broken down into manageable small sprints, which were previously conducted one to two weeks ago but are typically conducted every couple of days (Cohn, 2022).

4.2 Database Development

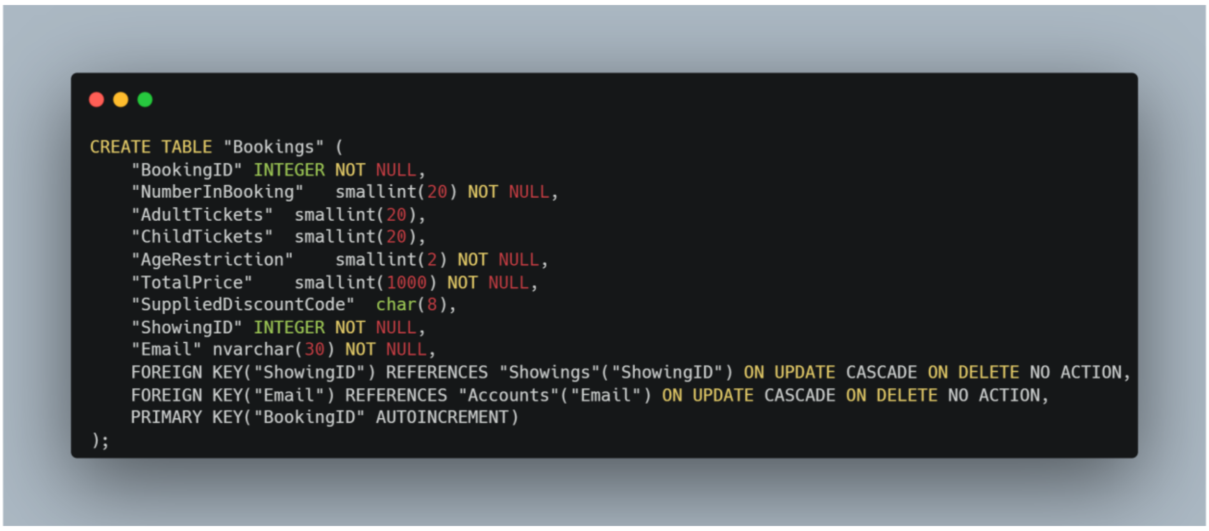
Previously created, the Enhanced-Entity-Relationship (EER) model and data dictionary will allow for developing a coherent and suitable database. The database is a core mechanic of the application, which requires appropriate planning and development. Translation of our model and dictionary will be complete using Structured Query Language (SQL) commands to create the database.

4.2.1 Database SQL

SQL will be used to create the database, which is the industry-accepted standard for interacting with relational databases (Oracle.com, 2022). In order to create the appropriate tables: relations between tables will have to be created using the “FOREIGN KEY” command. Constraints involving these relationships should also be modelled based upon the previously established data dictionary. After defining the table's relations and constraints, the columns will be created by defining a column name, a data type, the acceptance of null values, if the column is the key identifier of the table known as the primary key, and lastly, if the column performs automatic incrementation. The formula for adding a row is as follows:

Row Name Datatype, NULL accepted, AutoIncremement

As the process is repetitive for demonstration, only one of the following table creations will be shared in the documentation, which is shown in Figure 13.



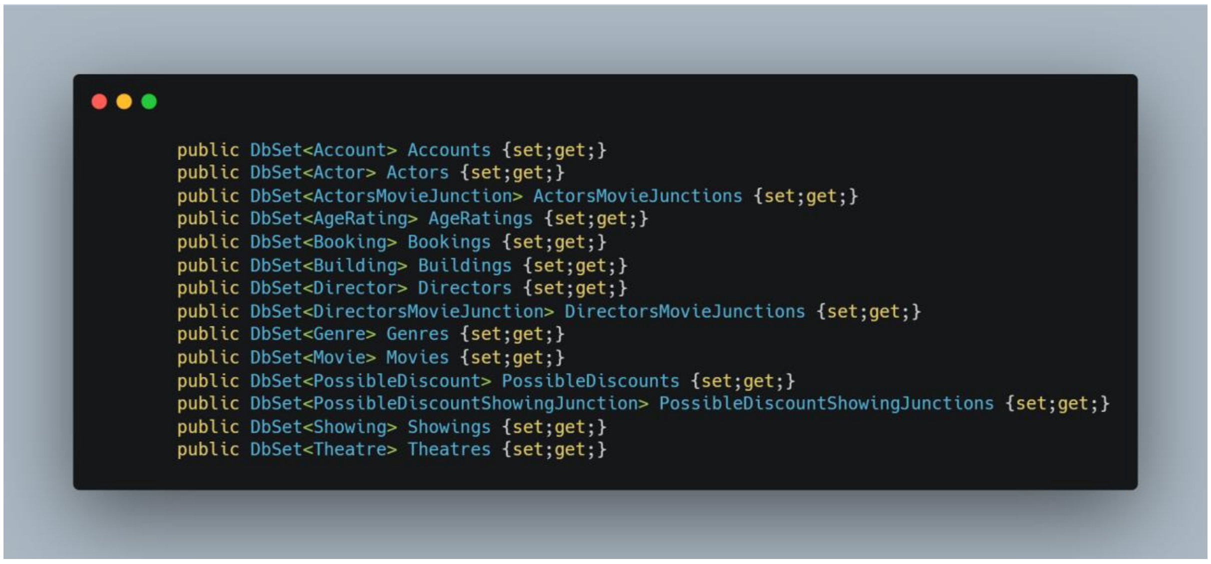
```sql
CREATE TABLE "Bookings" (
    "BookingID" INTEGER NOT NULL,
    "NumberInBooking"   smallint(20) NOT NULL,
    "AdultTickets"  smallint(20),
    "ChildTickets"  smallint(20),
    "AgeRestriction"    smallint(2) NOT NULL,
    "TotalPrice"    smallint(1000) NOT NULL,
    "SuppliedDiscountCode"  char(8),
    "ShowingID" INTEGER NOT NULL,
    "Email" nvarchar(30) NOT NULL,
    FOREIGN KEY("ShowingID") REFERENCES "Showings"("ShowingID") ON UPDATE CASCADE ON DELETE NO ACTION,
    FOREIGN KEY("Email") REFERENCES "Accounts"("Email") ON UPDATE CASCADE ON DELETE NO ACTION,
    PRIMARY KEY("BookingID" AUTOINCREMENT)
);
```

Figure 13: SQL Sample of Creating Booking Table

From the previous Figure 13, we can view that a table named "Bookings" has been created in the database. After the create table command has been issued, the table columns and their data types are defined. Lastly, the relationships are created following the constraints already established in the EER model. In Figure 13, it is shown that two foreign keys have been added to the database, referencing other tables' primary keys with a particular constraint that defines what will happen if the key is updated or deleted within the table. Lastly, we can conclude that a primary key for the table, typically a unique identifier, has been added with the column name "BookingID", which is set by the database and auto-incremented each time an entity is set is added to the database.

4.2.2 Context File

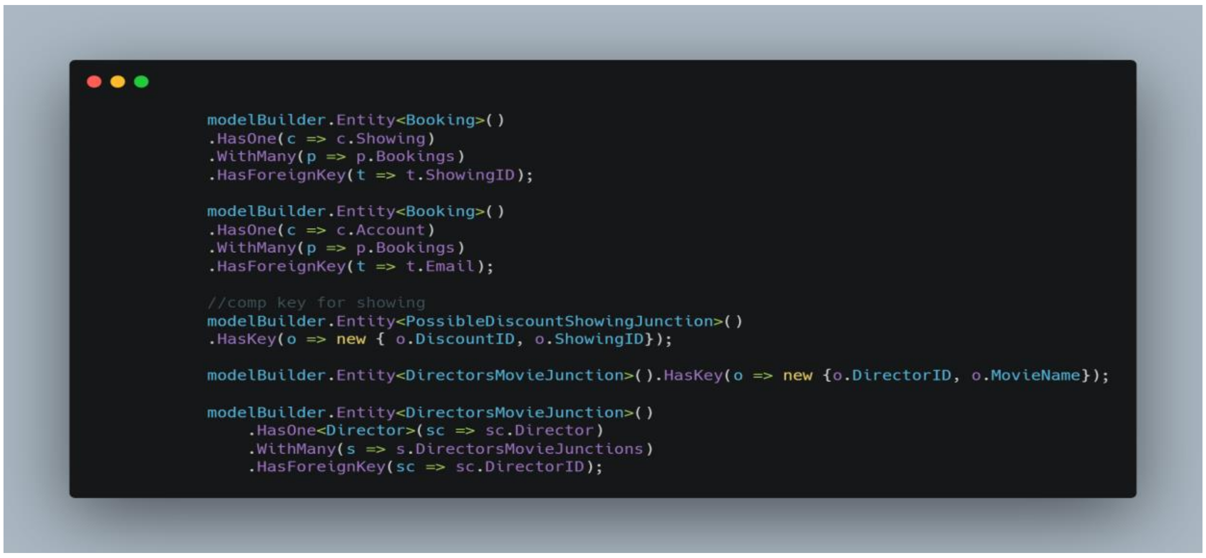
For the application to be able to represent the database, a context file has been created. The Cinema.cs file inherits from Microsofts pre-defined DbContext class, which is used to represent a combination of the unit of work and patterns which are needed in order to query from a database and group together collections of data that will be written in the database (Microsoft.com, 2022). Inheriting the DbContext class will allow the DbSet class to define sets of data, as can be seen in the following Figure 14.



```
public DbSet<Account> Accounts {set;get;}
public DbSet<Actor> Actors {set;get;}
public DbSet<ActorsMovieJunction> ActorsMovieJunctions {set;get;}
public DbSet<AgeRating> AgeRatings {set;get;}
public DbSet<Booking> Bookings {set;get;}
public DbSet<Building> Buildings {set;get;}
public DbSet<Director> Directors {set;get;}
public DbSet<DirectorsMovieJunction> DirectorsMovieJunctions {set;get;}
public DbSet<Genre> Genres {set;get;}
public DbSet<Movie> Movies {set;get;}
public DbSet<PossibleDiscount> PossibleDiscounts {set;get;}
public DbSet<PossibleDiscountShowingJunction> PossibleDiscountShowingJunctions {set;get;}
public DbSet<Showing> Showings {set;get;}
public DbSet<Theatre> Theatres {set;get;}
```

Figure 14: DbSets

After creating the data sets relating to the tables constrained within the database, the next stage is using the modelBuilder to create purposeful relationships within the application. Relationships are also defined in the Cinema.cs class, aswell as the creation of composite keys and defining foreign keys, which a small snippet of the code can be viewed in the following Figure 15.

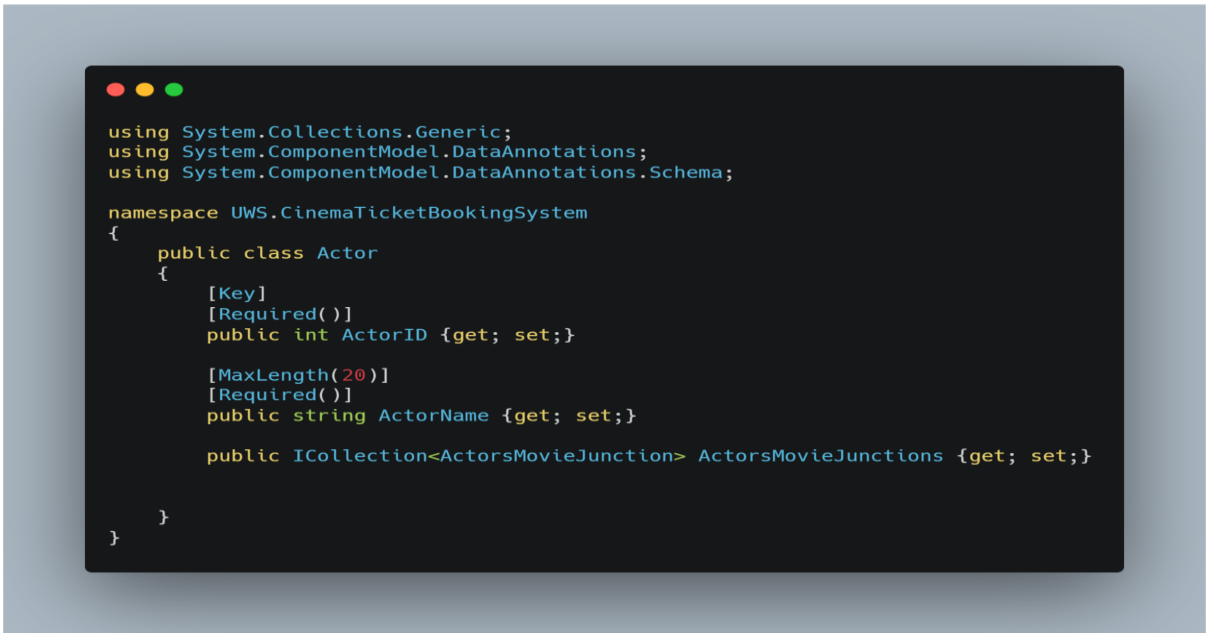


```
modelBuilder.Entity<Booking>()
.HasOne(c => c.Showing)
.WithMany(p => p.Bookings)
.HasForeignKey(t => t.ShowingID);

modelBuilder.Entity<Booking>()
.HasOne(c => c.Account)
.WithMany(p => p.Bookings)
.HasForeignKey(t => t.Email);

//comp key for showing
modelBuilder.Entity<PossibleDiscountShowingJunction>()
.HasKey(o => new { o.DiscountID, o.ShowingID});

modelBuilder.Entity<DirectorsMovieJunction>().HasKey(o => new {o.DirectorID, o.MovieName});

modelBuilder.Entity<DirectorsMovieJunction>()
    .HasOne<Director>(sc => sc.Director)
    .WithMany(s => s.DirectorsMovieJunctions)
    .HasForeignKey(sc => sc.DirectorID);
```

Figure 15: Example of relations in Cinema.cs

4.2.3 Entities

Lastly, before connecting the database to the application, it requires that the application can understand what is contained within the tables shown in the following Figure 16.



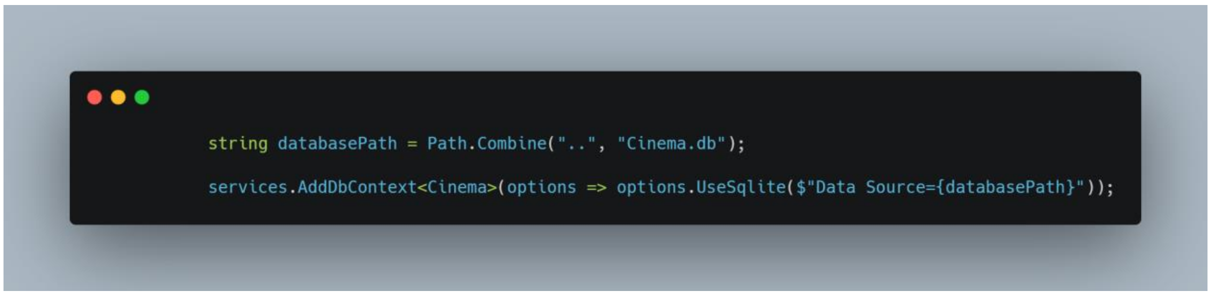
```
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace UWS.CinemaTicketBookingSystem
{
    public class Actor
    {
        [Key]
        [Required()]
        public int ActorID {get; set;}

        [MaxLength(20)]
        [Required()]
        public string ActorName {get; set;}

        public ICollection<ActorsMovieJunction> ActorsMovieJunctions {get; set;}


    }
}
```

Figure 16: Example of Entity Class for Actor

Using the Microsoft DataAnnotations library, which is seen to be imported on the second line of the previous image, DataAnnotations allows the developer to define many constraints, in this case, the “[Key]” which represents the primary key and the required tag which ensures that data cannot be submitted when the ActorID column is not set to a value. Lastly to note is the collection of “ActorsMovieJunctions” refers to relationships that the Actor table share with the “ActorsMovieJunctions” table; which in this particular case is a one to many from the Actor to the junction table hence the use of a collection and not a data type which stores a singular instance of data.

4.2.4 Database Connection

As previously mentioned, the database will be managed using SQLite. For the application to connect to the database, the DB file must be identified and targeted in the startup.cs file, where the data source should also be set.



```
string databasePath = Path.Combine("..", "Cinema.db");

services.AddDbContext<Cinema>(options => options.UseSqlite($"Data Source={databasePath}"));
```

Figure 17: Database Target and Driver Interface

Figure 17 illustrates the connection and interface used when interacting with the database. The code shown is from the start-up file, which is the root location that the application is executed from. Database location is set using the “Path.Combine” function followed with the parameters, which is the location to where the database is stored in this case; the application steps out of the folder directory that is holding the startup file, which is represented by the “..” notation. Once the application has stepped out of the folder, the “Cinema.db” file is located and set in the string variable. Figure 17 is the selection of DBMS used as the driver to connect to the database, which is set by using “options.UseSqlite”.

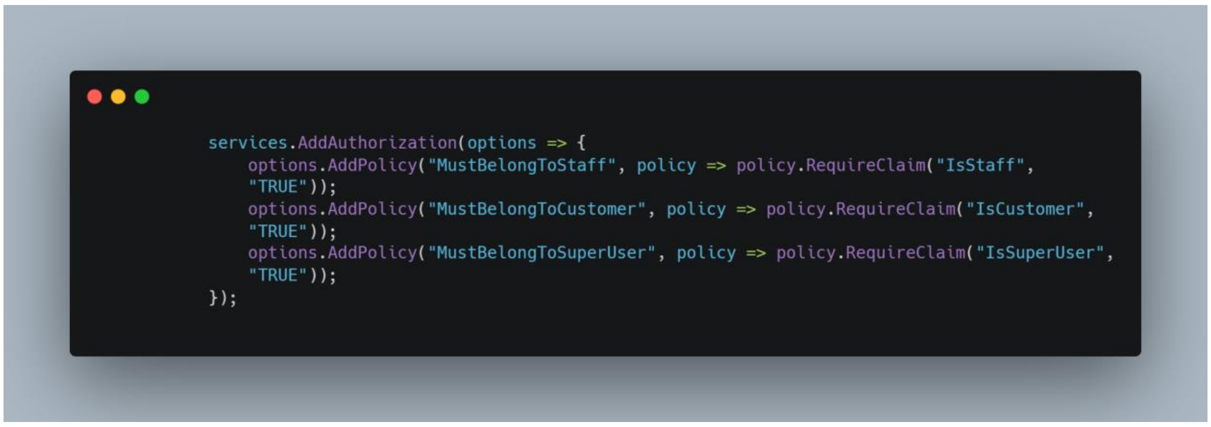
4.2.5 Pages

Pages directory within the prototype contains all the razor pages and their corresponding model classes. Razor pages are used to present the user with a front end to interact with whilst the model handles the POST and GET requests. GET requests are “used to request data from a specified resource” POST requests are the processes of when “data sent to the server with POST is stored in the request body of the HTTP request” (W3schools.com, 2022). Most commonly in the application, GET requests will be called when a user requests a particular razor page, which will trigger the OnGet function stored in the model, whilst post requests are sent calling the OnPost method when a forum is submitted to the application; example code extract is shown in Appendix 1, Figure 7.

To maintain a similar layout throughout the application whilst reducing the amount of repetitive code, the “Pages/Shared/_Layout.cshtml” will be implemented to ensure that the prototype has a high-level template for views within the application (Smith and Brock, 2022).

4.2.6 Cookies Claims and Page Policy

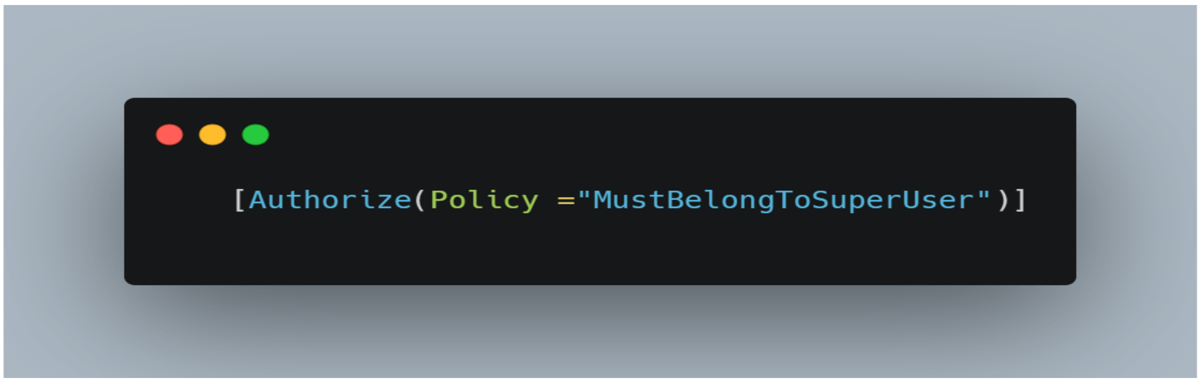
As a primary focus of the application, user accounts and database security have been a predominant focus in enforcing the access control list in terms of access to the application. Cookie claims have been implemented to identify levels of users and restrict application access based upon their encrypted delegated cookie. In Figure 18, it is shown the three levels of access a user account may be assigned.



```
services.AddAuthorization(options => {
    options.AddPolicy("MustBelongToStaff", policy => policy.RequireClaim("IsStaff",
    "TRUE"));
    options.AddPolicy("MustBelongToCustomer", policy => policy.RequireClaim("IsCustomer",
    "TRUE"));
    options.AddPolicy("MustBelongToSuperUser", policy => policy.RequireClaim("IsSuperUser",
    "TRUE"));
});
```

Figure 18: Add Policies to application

Figure 18 illustrates the three possible levels of access to the application. “IsSuperUser” is the highest level of user, and it is recommended that only one user should ever have this level of access, as they can alter all accounts and upgrade standard customer accounts to staff accounts. The second level of access is the “IsStaff” staff are assigned permissions that are needed to perform their job roles, such as creating film listings and showings. Lastly is the customer cookie claim, “IsCustomer”, allowing users to create bookings. Using these cookies and policies, the application can implement page policies to restrict user access based on their assigned cookie illustrated in Figure 19.



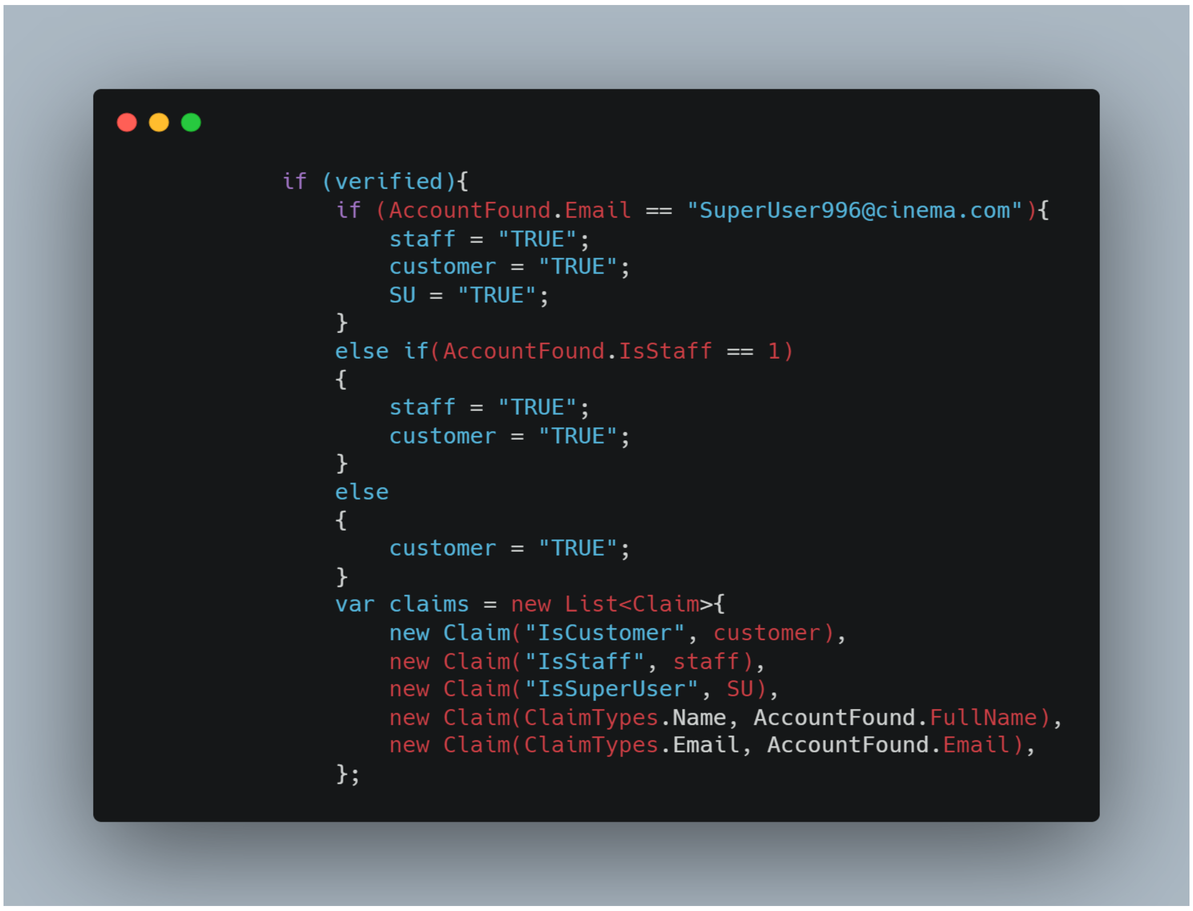
```
[Authorize(Policy ="MustBelongToSuperUser")]
```

Figure 19: Page Policy

During the user logging into their account is where cookie claims are assigned to the user. Claims are assigned depending on a few factors:

1. That a user's account in the database isStaff is set to 1
2. User Account email is “SuperUser996@cinema.com”
3. Customer is logged in

While these factors will decide what claims are assigned to a user's cookie, password verification must be completed before any claims can be assigned to the user. All claims are set to False and updated according to the account after logging in, shown in Figure 20.



```
if (verified){
    if (AccountFound.Email == "SuperUser996@cinema.com"){
        staff = "TRUE";
        customer = "TRUE";
        SU = "TRUE";
    }
    else if(AccountFound.IsStaff == 1)
    {
        staff = "TRUE";
        customer = "TRUE";
    }
    else
    {
        customer = "TRUE";
    }
    var claims = new List<Claim>{
        new Claim("IsCustomer", customer),
        new Claim("IsStaff", staff),
        new Claim("IsSuperUser", SU),
        new Claim(ClaimTypes.Name, AccountFound.FullName),
        new Claim(ClaimTypes.Email, AccountFound.Email),
    };
```

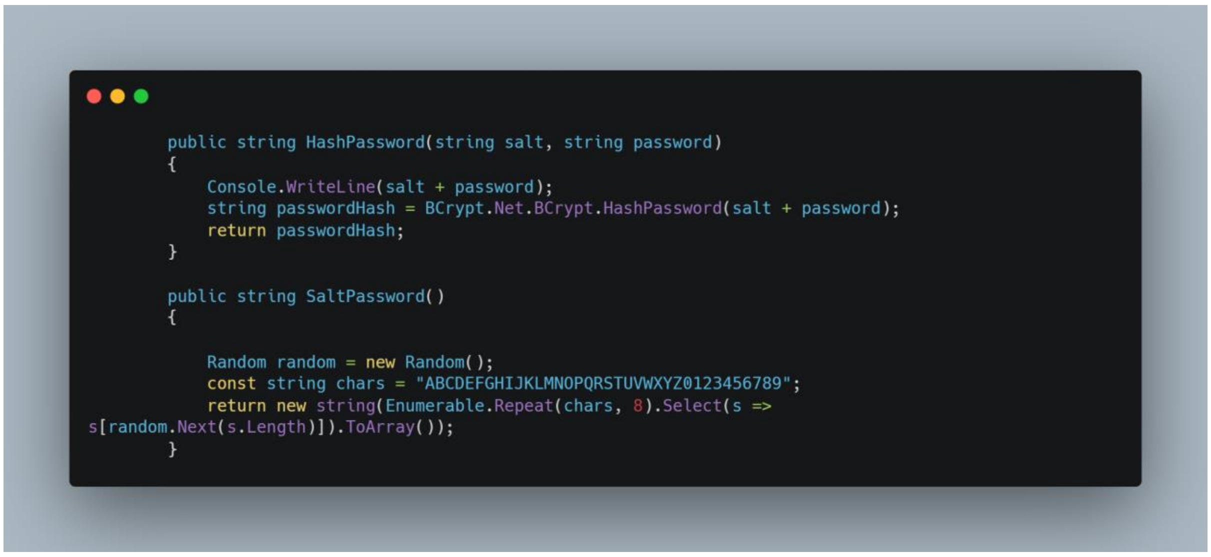
Figure 20: cookie claims at login

4.2.7 Password Protection Implementation

Previously discussed, Bcrypt and password salting have been the selected method of account protection. Password protection and validation occurs mainly in two stages of the application:

1. Creating and storing an account
2. Verifying a users' password when logging in

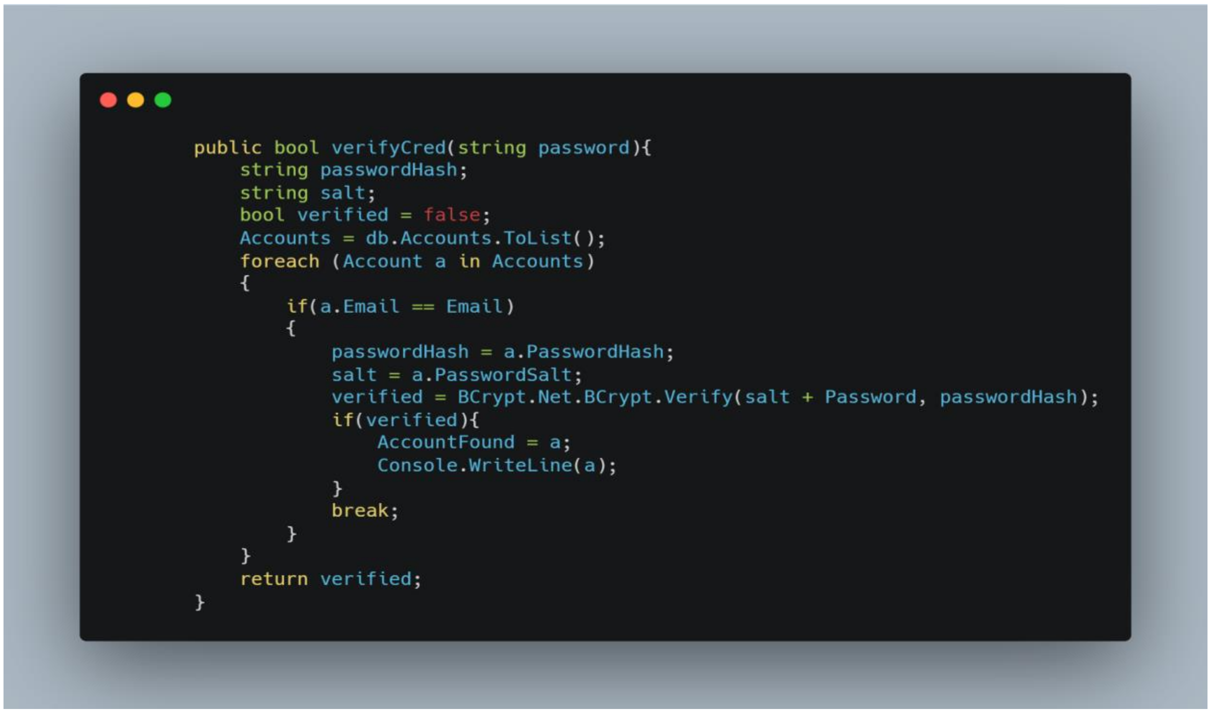
When creating a user account, the application receives a plain text password from the user sent in the body of a POST request. The application then generates a random salt, appending it to the start of the password. The password is then passed through the BCrypt function, resulting in a non-human readable safe to store password, as shown in Figure 21.



```
        public string HashPassword(string salt, string password)
        {
            Console.WriteLine(salt + password);
            string passwordHash = BCrypt.Net.BCrypt.HashPassword(salt + password);
            return passwordHash;
        }

        public string SaltPassword()
        {

            Random random = new Random();
            const string chars = "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789";
            return new string(Enumerable.Repeat(chars, 8).Select(s =>
s[random.Next(s.Length)]).ToArray());
        }
```

Figure 21: salt and hash functions

When a user is logging in, the database will need to be presented with the hash value from the users' forum submission, as there is no way for the server to decrypt a users' password securely. The user logging in is visible in Figure 22, shown below.



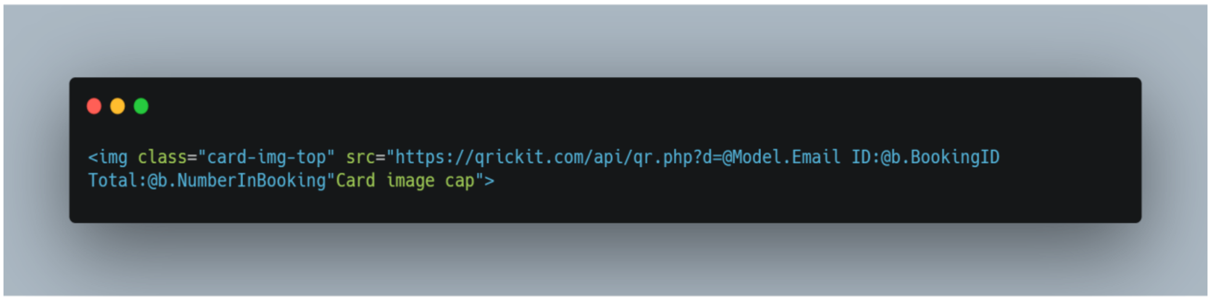
```
public bool verifyCred(string password){
    string passwordHash;
    string salt;
    bool verified = false;
    Accounts = db.Accounts.ToList();
    foreach (Account a in Accounts)
    {
        if(a.Email == Email)
        {
            passwordHash = a.PasswordHash;
            salt = a.PasswordSalt;
            verified = BCrypt.Net.BCrypt.Verify(salt + Password, passwordHash);
            if(verified){
                AccountFound = a;
                Console.WriteLine(a);
            }
            break;
        }
    }
    return verified;
}
```

Figure 22: Verify Password

The function in Figure 22 verifies a user's email and password. Firstly, the application will ensure that the email address submitted is within the database. After confirming the email is present, the password hash value is compared against the submitted password with the appended salt from the database, which is passed through the same BCrypt function; if the password is the same, the function will result in the comparison proving true, after the appropriate cookies are assigned.

4.2.8 API QRickit

Previously, described that the API provided for free by QRickit.com, is extremely simple to use and does not require any private keys. Parameters for the QR code are sent after the ".php?", as shown in the following Figure 23.



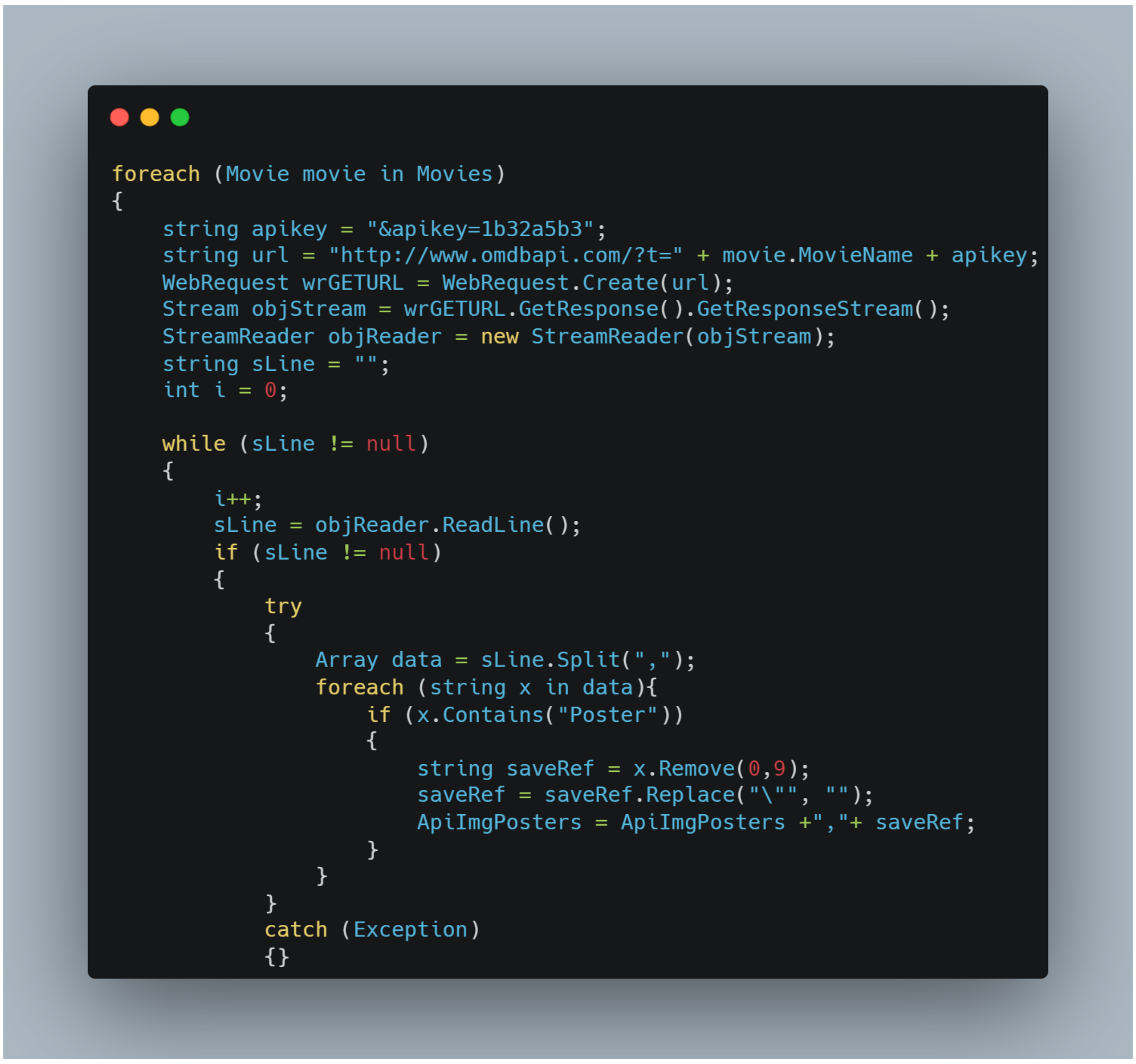
```
<img class="card-img-top" src="https://qrickit.com/api/qr.php?d=@Model.Email ID:@b.BookingID
Total:@b.NumberInBooking"Card image cap">
```

Figure 23: QRickit API

Figure 23 shows the user's email address, booking ID, and the total number of people in the party sent to the API. The API transforms this data and responds with a generated QR code that is scannable using the small program developed in Python to confirm the booking.

4.2.9 API OMDb

OMDb will be providing the prototype with its images for the application's index page. A private key has been requested for this project and obtained, which is stored in the "apikey" variable, shown in the following Figure 24. Next, the application requests information about the movie using the APIs source point with the movie name and API key; a JSON object is returned containing relative information about the movie that was requested. In the application's case, the "Poster" (image source) is extracted and appended to a string used in the razor page image tag to display the image to the customer.

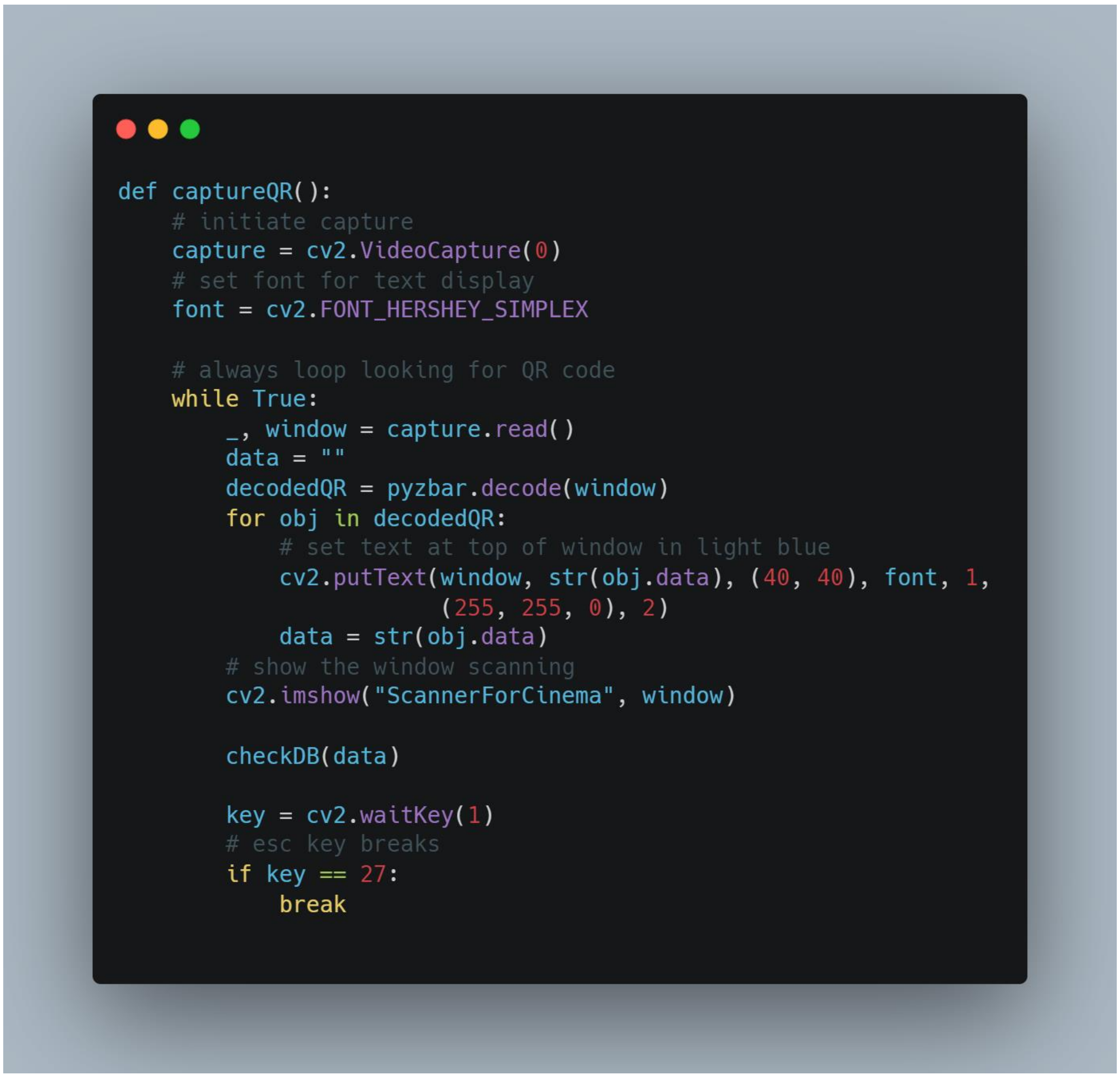


```
foreach (Movie movie in Movies)
{
    string apikey = "&apikey=1b32a5b3";
    string url = "http://www.omdbapi.com/?t=" + movie.MovieName + apikey;
    WebRequest wrGETURL = WebRequest.Create(url);
    Stream objStream = wrGETURL.GetResponse().GetResponseStream();
    StreamReader objReader = new StreamReader(objStream);
    string sLine = "";
    int i = 0;

    while (sLine != null)
    {
        i++;
        sLine = objReader.ReadLine();
        if (sLine != null)
        {
            try
            {
                Array data = sLine.Split(",");
                foreach (string x in data){
                    if (x.Contains("Poster"))
                    {
                        string saveRef = x.Remove(0,9);
                        saveRef = saveRef.Replace("\"", "");
                        ApiImgPosters = ApiImgPosters +","+ saveRef;
                    }
                }
            }
            catch (Exception)
            {}
```

Figure 24: Movie Information API

4.2.10 Camera Scanner Python

A system designed separately for self-service admission has been created using Python. In order to operate this application, the OpenCV Python library must be imported. Figure 25, shown below, illustrates the application first attempting to locate a video capture device in the prototype. The application is making use of the webcams video capture interface. Using the library, the application will attempt to detect any QR codes and displays the decoded QR text at the top of the video capture feed.

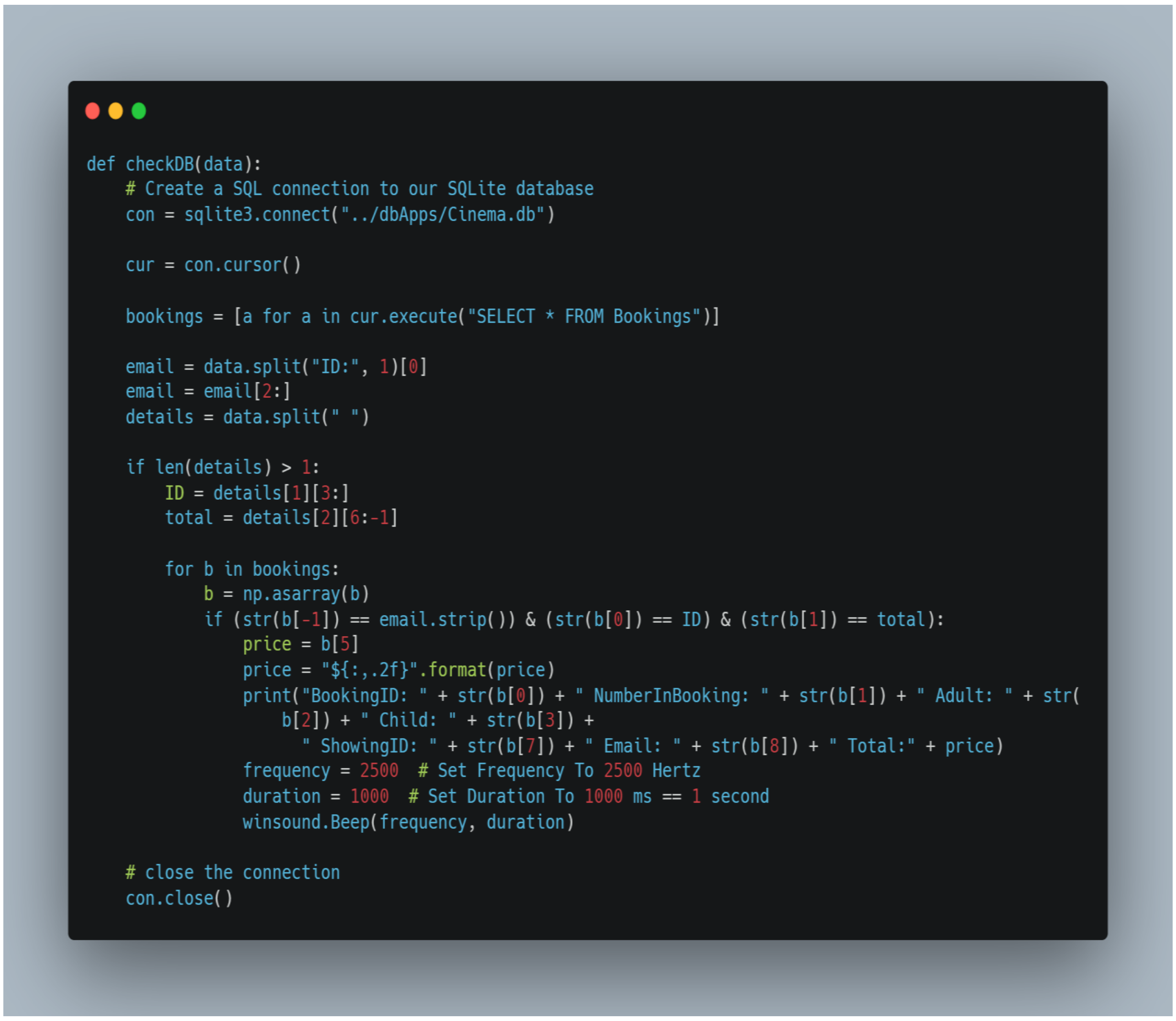


```python
def captureQR():
    # initiate capture
    capture = cv2.VideoCapture(0)
    # set font for text display
    font = cv2.FONT_HERSHEY_SIMPLEX

    # always loop looking for QR code
    while True:
        _, window = capture.read()
        data = ""
        decodedQR = pyzbar.decode(window)
        for obj in decodedQR:
            # set text at top of window in light blue
            cv2.putText(window, str(obj.data), (40, 40), font, 1,
                        (255, 255, 0), 2)
            data = str(obj.data)
        # show the window scanning
        cv2.imshow("ScannerForCinema", window)

        checkDB(data)

        key = cv2.waitKey(1)
        # esc key breaks
        if key == 27:
            break
```

Figure 25: Camera Connection and QR Detection

After decoding the QR code, the data must be checked against the bookings in the cinema database. Using SQLite, as previously stated and shown in the following

Figure 26, the Python application connects to the database and requests all bookings in the database. The QR code contains the relevant email, booking ID, and the total number of people in the booking compared against the database bookings. If the data is present, the application will display the booking details in the Python terminal whilst creating a loud beep noise to indicate that the QR code has been accepted. The connection to the database is then closed to enforce security, that the connection is not left open unnecessary.

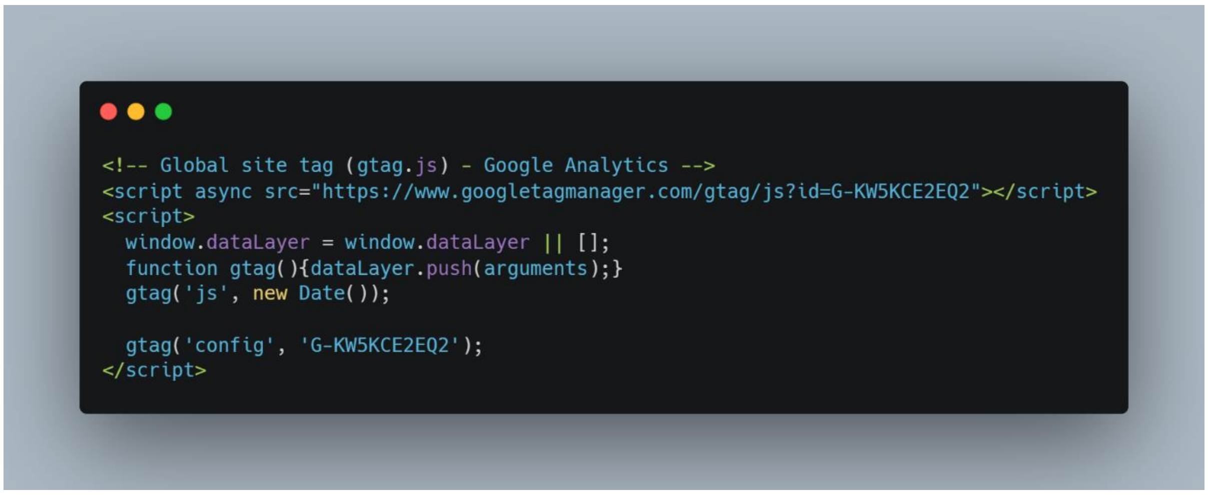


```python
def checkDB(data):
    # Create a SQL connection to our SQLite database
    con = sqlite3.connect("../dbApps/Cinema.db")

    cur = con.cursor()

    bookings = [a for a in cur.execute("SELECT * FROM Bookings")]

    email = data.split("ID:", 1)[0]
    email = email[2:]
    details = data.split(" ")

    if len(details) > 1:
        ID = details[1][3:]
        total = details[2][6:-1]

        for b in bookings:
            b = np.asarray(b)
            if (str(b[-1]) == email.strip()) & (str(b[0]) == ID) & (str(b[1]) == total):
                price = b[5]
                price = "${:,.2f}".format(price)
                print("BookingID: " + str(b[0]) + " NumberInBooking: " + str(b[1]) + " Adult: " + str(
                    b[2]) + " Child: " + str(b[3]) +
                      " ShowingID: " + str(b[7]) + " Email: " + str(b[8]) + " Total:" + price)
                frequency = 2500  # Set Frequency To 2500 Hertz
                duration = 1000  # Set Duration To 1000 ms == 1 second
                winsound.Beep(frequency, duration)

    # close the connection
    con.close()
```

Figure 26: Check Database For QR Code Value

4.2.11 Google Analytics

Google Analytics has also been added which can be used to monitor a multitude of activities such as, number of users, bounce rates, average session durations and page views. Google Analytics has been added by applying the code provide by Google to the shared layout which can be viewed in Figure 27.



```
<!-- Global site tag (gtag.js) - Google Analytics -->
<script async src="https://www.googletagmanager.com/gtag/js?id=G-KW5KCE2EQ2"></script>
<script>
  window.dataLayer = window.dataLayer || [];
  function gtag(){dataLayer.push(arguments);}
  gtag('js', new Date());

  gtag('config', 'G-KW5KCE2EQ2');
</script>
```

Figure 27: Google Analytics

As the prototype is not currently hosted on a live domain, Google analytics will not be able to monitor the traffic accessing the application. When deploying the system in a corporate setting, users should assign their own “gtag” to connect the analytics to a private monitoring dashboard. Analytics can be used to discover where customers are abandoning their bookings, allowing owners to identify issues and rectify them to create a better experience for the customer.

4.3 Development of Website

Wireframe development has been completed to ensure an appropriate user interface following the Guttenberg pattern, shown in Appendix 1 Figure 4. Wireframes will also attempt to convey a sense of simplicity; this should be illustrated using minimum elements per page. Lastly, another way to increase simplicity in terms of the user interface is by creating a minimum number of steps that are needed to perform an action that has been previously established in the UML diagram. The landing page is potentially the most important page, which is what a customer is first presented with after logging in.

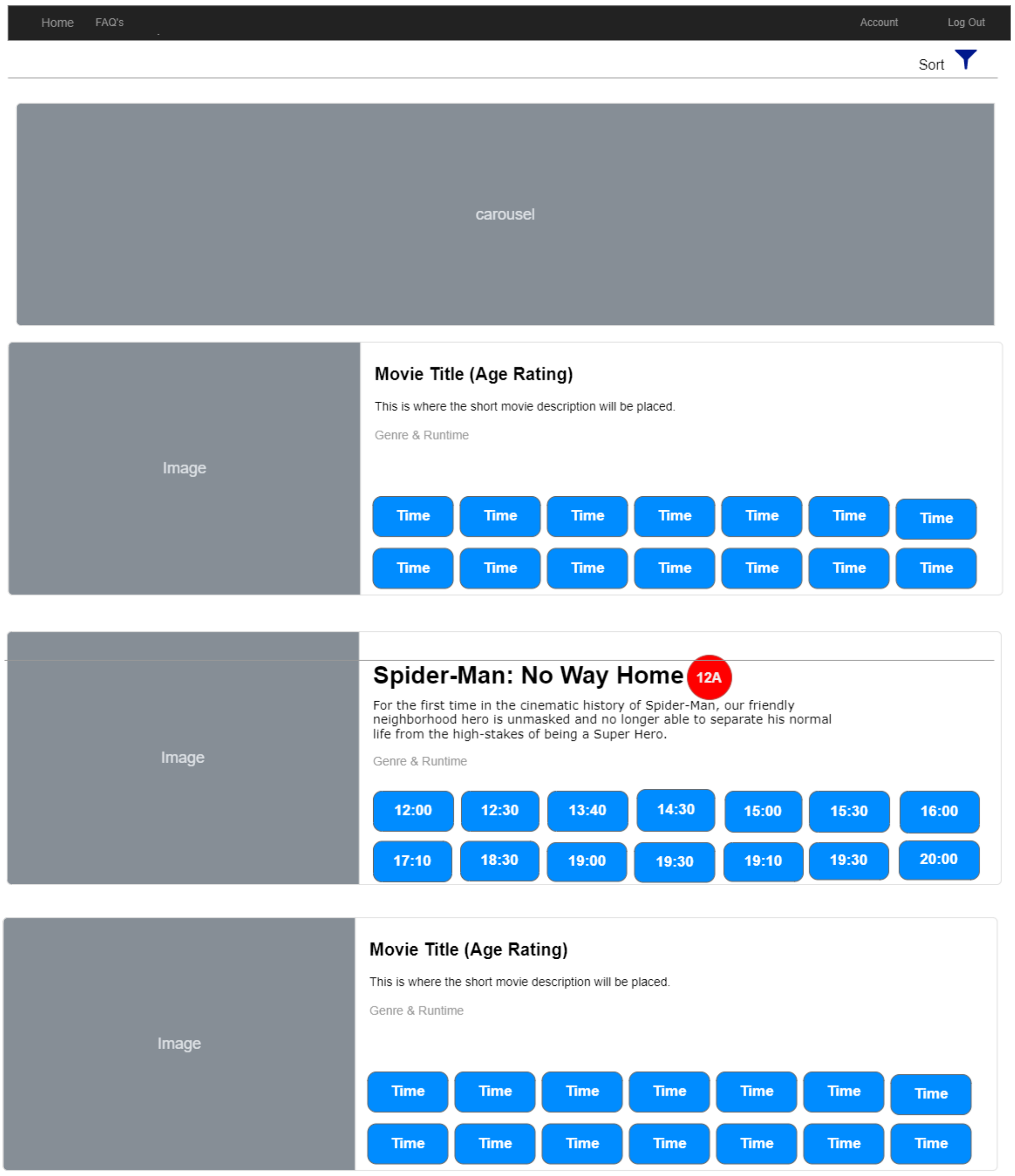
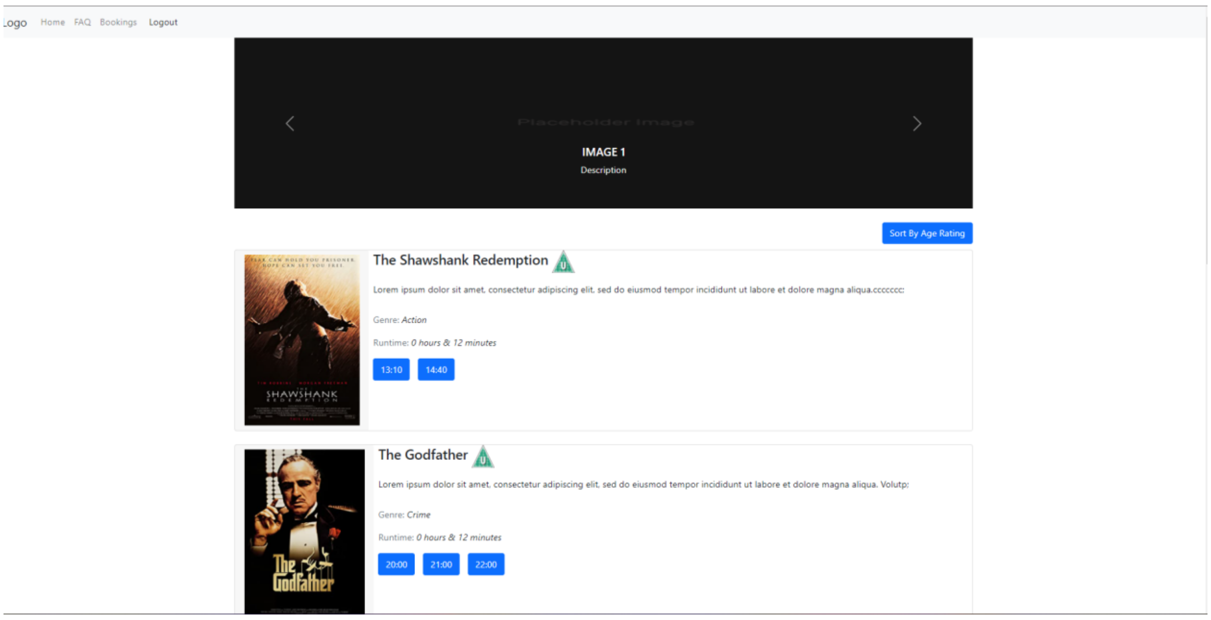

Home
FAQ's
Account
Log Out
Sort
carousel
Image
Movie Title (Age Rating)
This is where the short movie description will be placed.
Genre & Runtime
Time
Image
Spider-Man: No Way Home
12A
For the first time in the cinematic history of Spider-Man, our friendly neighborhood hero is unmasked and no longer able to separate his normal life from the high-stakes of being a Super Hero.
Genre & Runtime
12:00
12:30
13:40
14:30
15:00
15:30
16:00
17:10
18:30
19:00
19:30
19:10
19:30
20:00
Image
Movie Title (Age Rating)
This is where the short movie description will be placed.
Genre & Runtime
Time

Figure 28: Landing Page Wireframe

Figure 28 shows the application's homepage, where users can scroll through film listings and select a suitable time. Bookings are created by clicking the time that a user intends to book.

The following Figure 29 represents the translation from planning to the development of the booking page. The relocation of the logout button and bookings have been moved in order relating to the Guttenberg pattern, ensuring users notice all options available to them on logging in, as the top left is where a user's eyes are most likely to start digesting a webpage.



The Shawshank Redemption

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.ccccccc:

Genre: *Action*

Runtime: *0 hours & 12 minutes*

The Godfather

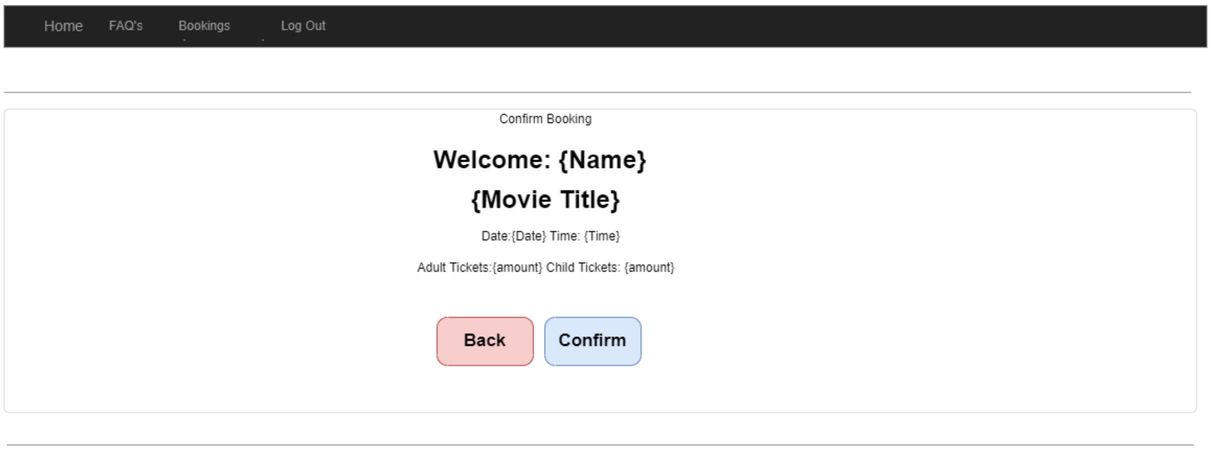
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Volutp:

Genre: *Crime*

Runtime: *0 hours & 12 minutes*

Figure 29: Landing Page Implementation

Following the flow of the customer selecting a time they wish to book a film, they will then be presented with a confirmation screen to ensure that no bookings are created by accident and to provide user feedback. Figure 30 demonstrates what a confirmation for a selected film and time would present.



Confirm Booking

Welcome: {Name}

{Movie Title}

Date:{Date} Time: {Time}

Adult Tickets:{amount} Child Tickets: {amount}

Figure 30: Booking Confirmation Wireframe

Figure 30 has been translated into the following Figure 31. The illustrated step encourages a customer to ensure that the correct reservation is being created, resulting in a better customer experience.

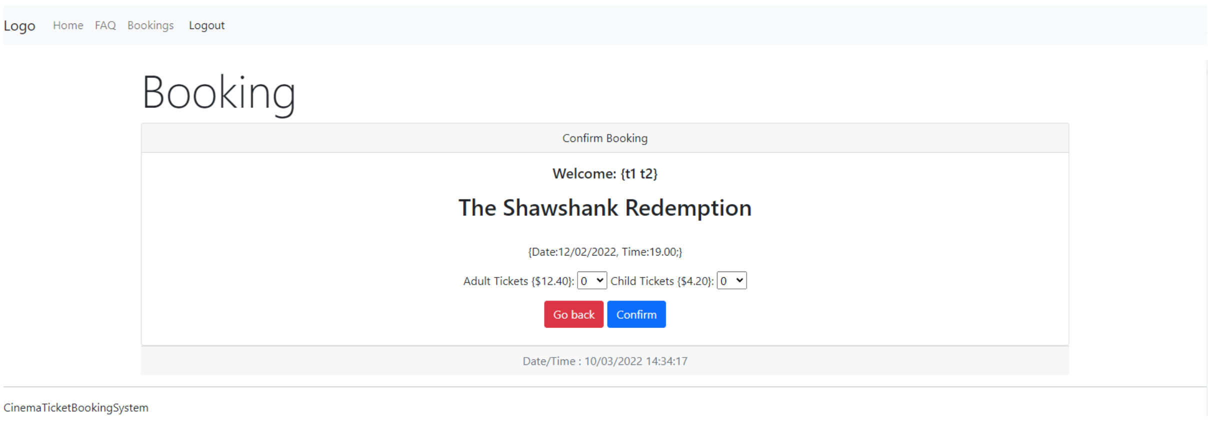


Figure 31: Confirm Booking Implementation

Confirming the booking will then take the user to their account bookings. Customers can view all the bookings that they currently store in their accounts. The wireframe shown below in Figure 32 has been created with ease of access and simplicity.

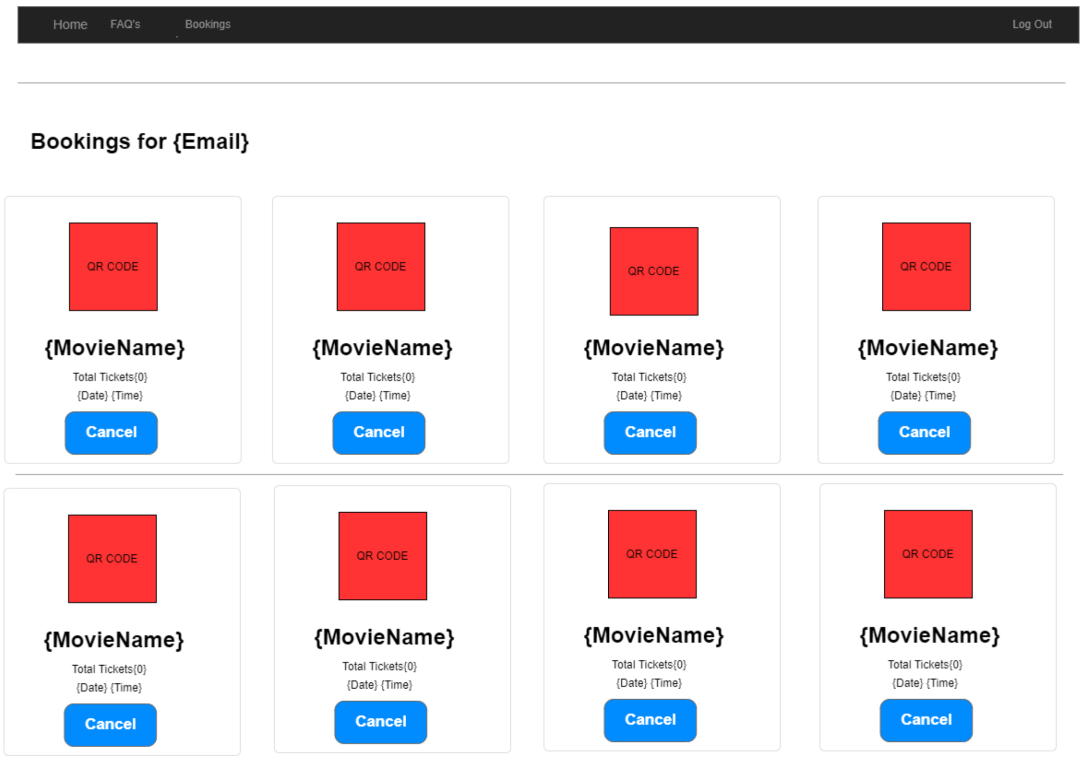
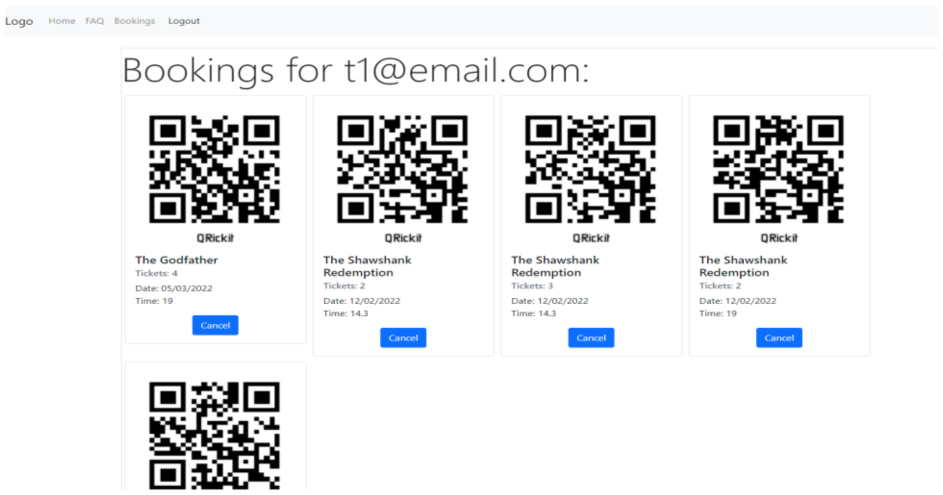


Figure 32: Bookings View Wireframe

As shown above, the booking page presents customers with all of their bookings in card formats, displaying QR codes that can be scanned to validate the booking. The wireframe has been translated into the following Figure 33.



Bookings for t1@email.com:

QRickit
The Godfather
Tickets: 4
Date: 05/03/2022
Time: 19
Cancel

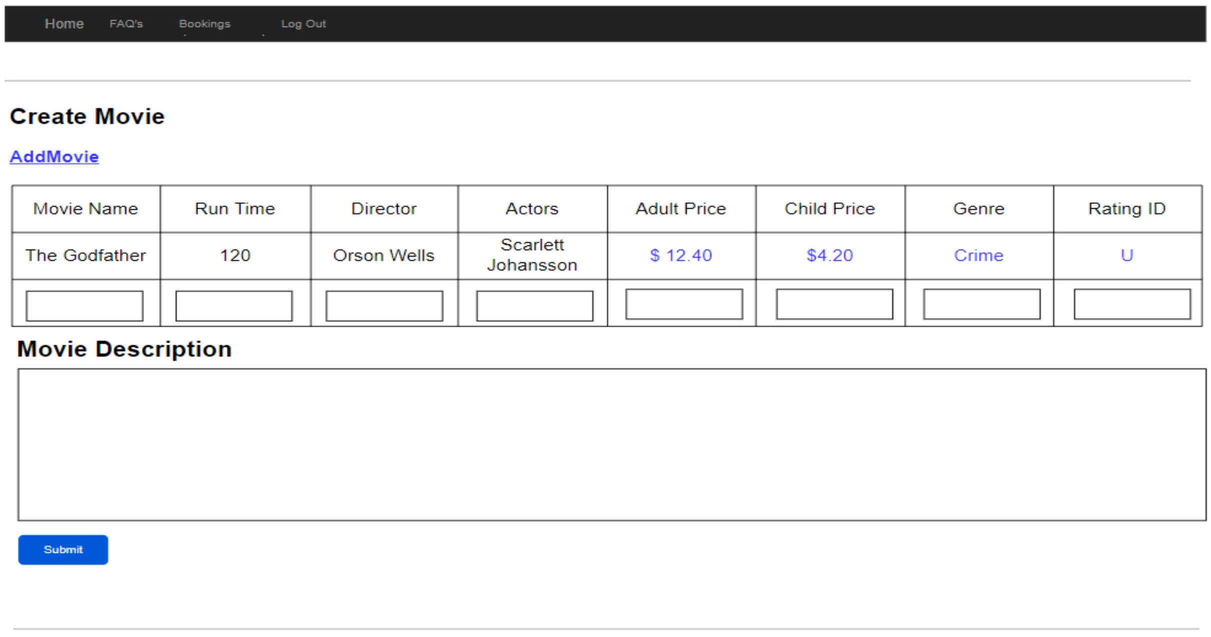
QRickit
The Shawshank Redemption
Tickets: 2
Date: 12/02/2022
Time: 14.3
Cancel

QRickit
The Shawshank Redemption
Tickets: 3
Date: 12/02/2022
Time: 14.3
Cancel

QRickit
The Shawshank Redemption
Tickets: 2
Date: 12/02/2022
Time: 19
Cancel

Figure 33: Bookings Implementation

Creating a movie, which is an action provided for staff, has been designed through the following wireframe shown in Figure 34.



Create Movie

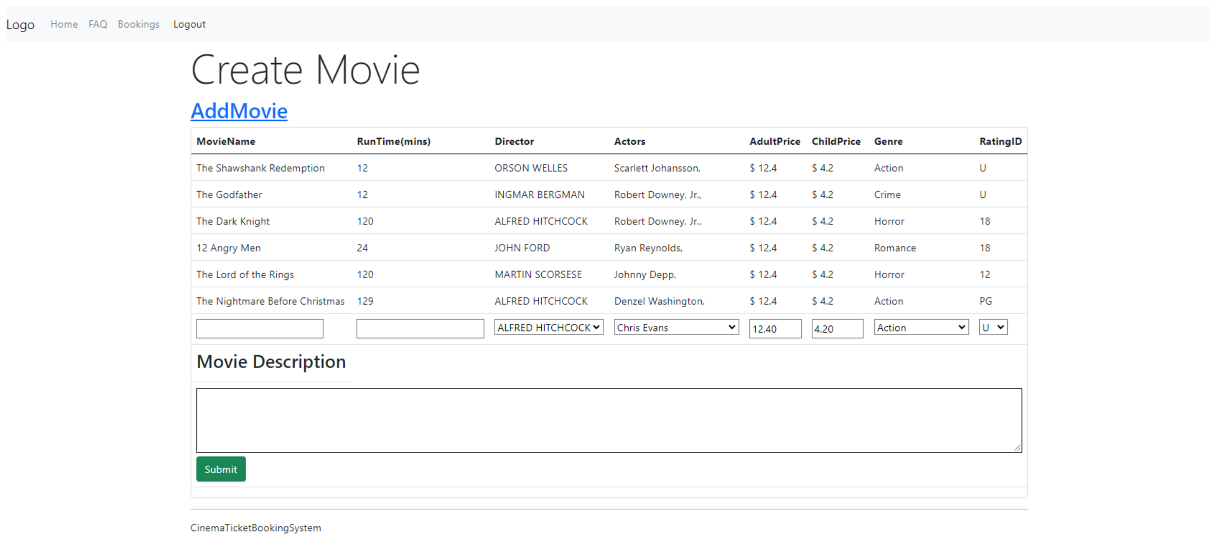
AddMovie

Movie Name	Run Time	Director	Actors	Adult Price	Child Price	Genre	Rating ID
The Godfather	120	Orson Wells	Scarlett Johansson	$ 12.40	$4.20	Crime	U

Movie Description

Figure 34: Create Movie Wireframe

Figure 34 illustrates the simple yet effective design wireframe which has been translated to the following, Figure 35.



Create Movie

AddMovie

MovieName	RunTime(mins)	Director	Actors	AdultPrice	ChildPrice	Genre	RatingID
The Shawshank Redemption	12	ORSON WELLES	Scarlett Johansson,	$ 12.4	$ 4.2	Action	U
The Godfather	12	INGMAR BERGMAN	Robert Downey, Jr.,	$ 12.4	$ 4.2	Crime	U
The Dark Knight	120	ALFRED HITCHCOCK	Robert Downey, Jr.,	$ 12.4	$ 4.2	Horror	18
12 Angry Men	24	JOHN FORD	Ryan Reynolds,	$ 12.4	$ 4.2	Romance	18
The Lord of the Rings	120	MARTIN SCORSESE	Johnny Depp,	$ 12.4	$ 4.2	Horror	12
The Nightmare Before Christmas	129	ALFRED HITCHCOCK	Denzel Washington,	$ 12.4	$ 4.2	Action	PG
		ALFRED HITCHCOCK	Chris Evans	12.40	4.20	Action	U

Movie Description

Figure 35: Create Movie Implementation

Similar to the previous wireframe, all staff action pages will be formatted to the same accord to create familiarity and simplicity throughout the staff management section of the prototype.

The previous interface designs intend to convey simplicity through the minimal amount of information and steps needed to create a booking. Whilst ensuring that no bookings are made in error, which is attempted through confirmation prompts. Additional wireframes can be shown in Appendix 1, Figures 8 to 14.

5.0 Evaluate

5.1 Introduction

The following section intends to test and refine the whole system. Overall, the main focus of this section, but not limited to, is the user-facing portions of the application, as customer experience is a pivotal contributor to any successful business.

5.2 Nielsen's Heuristic Evaluation

Nielsen's Heuristic Evaluation will be conducted with four anonymous participants. That evaluation will be conducted on the user registration page and a customer's processes creating a booking for a movie showing. Four random anonymous individuals have tested the processes that are expected to be carried out regularly in the application, shown in Figures: four, five, six and seven. When interacting with the staff section of the application, evaluators started at the /Staff/Staff page. Due to complications with the COVID-19 virus, evaluation has been conducted over Microsoft Teams, where users have interacted remotely with the prototype software running from my home network, opposed to the previously planned in-person software evaluation.

Evaluator 1 –

Observing the user creating an account, logging in and performing a booking showing proved incredibly straightforward. The main issue observed was that the customer was unsure that the booking was completed when the QR code was shown, which may suggest that customers may need more feedback on the status of their booking. Cancelling the booking also was a function that the evaluator performed without any problems. Performing the adding and new showing for a movie also proved extremely simple for the evaluator.

Evaluator 2 –

Observing the user creating an account logging in again was performed without issues. The evaluator opted to use the go back button while creating the booking, which did not result in the expected action. After some time, the customer managed to resolve the issue using the home button. The customer was asked to cancel the booking, which was performed without any issues. Evaluator two was required to change an account's email address, finding the appropriate page was located as expected, observations suggested that the users may have been confused when attempting to search an account email address which did not provide the expected results, on the second attempt the evaluator performed the required action suitably.

Evaluator 3 -

Evaluator three from observation results was able to create an account, log in, and create a booking with no issues. Cancelling the booking was performed as expected. Using the staff action to add a movie proved no issues for the evaluator.

Evaluator 4 -

Through observation, the user creating an account and logging in proved to work as the customer assumed, which led to the functions being carried out quickly. Evaluator 4 had no issues with being able to cancel their booking. Evaluator four was required to manage showings and remove the appropriate one, which again the evaluator did so with no issues.

5.2.1 Findings

The results provided from the evaluation it has shown that when a user completes a booking, visual feedback should be displayed to assure the user that the action has been completed successfully. Secondly, the next issue discovered was that the go back button did not work when the wrong showing had been selected for a booking, which will be resolved. Lastly, when a user was evaluating the admin action of updating an accounts email, a mistyped email was entered, which provided no feedback which proved to cause some confusion for the evaluator. More feedback will be added to this section and across the application as a whole as this aspect of the application has proven to cause the most issues during the conduction of Nielsen's Heuristic Evaluation.

5.3 Heuristics Followed

Nielsen’s top ten heuristics have been considered an implemented where appropriate which can be displayed in the following Table 3. The following shows where industry leaders have considered and abided to the Nielsen’s ten heuristics and where the prototype has attempted to do the same.

Heuristic (Nielsen, 2020)	Cineworld, Cinemark & AMC	Prototype
“Visibility of system status”	Feedback Present as soon as possible.	Prevents user with success or error feedback as soon as possible.
“Match between system and the real world”	Simple terms used in all cinemas such as “bookings”.	Similar terminology is used such as “bookings” and “movies”.
“User control and freedom”	Supports the ability to go back and cancel.	Support the ability to move back in the process or cancel the process completely.
“Error prevention”	Unable to create errors.	All non-user created errors should not be visible or effect the user.
"Recognition rather than recall"	Use of cinema showing images and times.	Times and images used in the same format.
“Flexibility and efficiency of use”	The ability to submit selections using the enter key.	POST requests can be made using the enter key.
"Aesthetic and minimalistic design”	Centred on page and minimal steps to create a booking.	Minimal amount of deign as it has been designed allowing corporations to implement a theme that they feel suits.
“Help users recognise, diagnose, and recover from errors”	Red text on errors, such as invalid login.	Green text for success messages. Red text for invalid messages.
“Consistency and standards”	All three sites share similar landing pages.	Landing page follows the convention set by the three analysed leading applications.
“Help and documentation”	Customer service contact email located in footer of page.	Video demonstration created with thorough documentation.

Table 3: Heuristics Discovered and Followed

5.4 Testing

Before testing commences, ensuring that the application has full functionality previously established in the UML diagram Figures: four, five, six and seven. Ensuring full functionality has been met ensures that the testing process is vigorous and does not fail to test any core functions of the application. The following tables: four, five and six illustrate that the functions in the UML diagram have been met.

Customer Functions

Requirements	Status
Login	✓
Create Account	✓
Create a Booking	✓
View Cinema Listings	✓

Table 4: Customer Functions

Staff Functions

Requirements	Status
Manage Bookings	✓
Add Movie	✓
Add Showing	✓
Create building	✓
Create Director	✓
Create Actor	✓

Table 5: Staff Functions

Admin Functions

Requirements	Status
Update/Reset Password	✓
View Account Details	✓
Manage All Account Details	✓
Create Building	✓

Table 6: Admin Functions

The testing plan and results can be viewed in Appendix 1, Figure 15. Failure to test can cause damage to brands, which can also lead to frustration and a loss in customer engagement (IBM, 2022). In some severe incidents, failure to test can lead to legal trouble and, therefore, finical problems.

5.4.1 HTML5 and CSS Standards Validation

All validation of HTML5 code will be translated using W3Cs HTML validator, which can be found at https://validator.w3.org. Validation is used to adhere to the standard and recognised practices accepted and familiar in the website development industry. The use of a validator is to ensure that code is easily expandable by other developers in the field. W3C has created a list of all the benefits of using a validator, which is as follows:

- "Validation as a debugging tool"
- "Validation as a future-proof quality check"
- "Validation eases maintenance"
- "Validation is a sign of professionalism"
- "Validation helps teach good practices"

(W3.org, *2022*)

As this project aims to develop a prototype that is easily extendable by any entity that wishes to develop it further, validation will support this through professional practices, future-proofing work and ease of maintenance. The following table seven, shows the validation results for each web page whilst validation is essential. It can be expected that some expected errors will remain and do not hinder the application and noted to ensure future developers are aware of the unrecognised conventions implemented.

WebPage (cshtml)	Errors	Resolved	Comment
Login	5	4	asp-for error left, as this connects to the bind property set in the CS file.
Index	6	5	All image tags appropriate alt value added.
FAQ	0	0	
Booking/Create	7	7	misplaced tag
RemoveBookingCustomers	78	78	due to the iterative loop, the number of errors

			was dramatically increased.
Staff/Staff	0	0	
Staff/Staff/Create	0	0	
Staff/Staff/Create/ Movie	11	11	Stray tags, placed correctly.
Staff/Staff/Create/Showing	0	0	
Staff/Staff/Create/Actor	0	0	
Staff/Staff/Create/Director	0	0	
Staff/Staff/Create/Building	0	0	
Remove/RemoveBookingStaff	103	86	Iterative error, asp-for binding values recognised as non-unique ideas in runtime translation.
Remove/RemoveShowings	382	378	Iterative error, using asp-for, binding values recognised as repeated binding ID’s. In this case, accepted and expected.
Staff/Admin	1	1	Stray Tag.
AccountManagement/AccountManagement	3	3	

Table 7: HTML Validation Results

5.4.2 Testing Table

As all proper functionality derived from the UML diagram has been implemented, testing will be conducted to ensure that functionality is appropriate for the application's use case established in Figures four to seven. The testing procedure will aim to reduce the likeliness of a non-functional application due to errors. A range of tests will be carried out with data ranging from expected, extreme, and exception triggering data, which can be viewed in Appendix 1 Figure 15, the testing table.

As mentioned beforehand, the application will be tested using Postman, which will allow the submission of custom POST and GET requests to the database. Whilst the testing will not cover all possible attack vectors, it demonstrates the security that the prototype possesses. The following table eight demonstrates potential attack vectors using custom POST and GET requests.

Type	Target	Query	Result	Reasoning
POST	https://localhost:7257/Login/Login	Email: t1@gmail.com Password: password	400 Bad Request	Request verification token not assigned by application
POST	https://localhost:7257/AccountManagement/AccountManagement	Email: t1@gmail.com	200 OK Redirected to login page	Not logged in
GET	https://localhost:7257/Remove/RemoveShowing		200 OK redirected to login page	Not logged in

Table 8: Postman Custom Requests

5.4.3 Findings and Improvements

Due to the agile development approach adopted, any unexpected actions discovered during the testing were revisited and corrected at the time of testing. A potential problem that occurred more than once was that those forum submissions were able to be submitted to the database with blank values, which is not how the application is intended to operate. To prevent users from having the ability to submit empty input fields, JQuery validation has been implemented, allowing developers to block forum submissions unless the data is suitable whilst providing appropriate user messages if the data is not suitable. During testing, other issues found in functional testing appeared to be isolated occurrences due to logical errors in the codebase, which were revisited and iterated upon until testing proved a functional and robust codebase.

6. Documentation

Substantial documentation can be found throughout the essay. The current section will focus on using the application and essential details that are needed to get started. Whilst customers can create accounts; a superuser account is pre-established for the deployment of the application. The email: "SuperUser996@cinema.com" and the password: "password", which has the maximum access to the application, can upgrade normal user accounts into staff accounts. Whilst the password convention, which does not force the user to include a mixture of unique, numerical, and alphabetical characters, is not recommended for final deployment as it is not as secure as alternatives, it has been left the way it is for simplicity in testing and development. In order to access staff controls, the user must navigate to the domain followed by "/Staff/Staff" this has been implemented in such a way to ensure that standard users do not have to receive any element that is not suited to their task. A shortlist has been created to ensure that any future developers can easily develop upon this prototype:

1. Superuser account is hard-coded must remain as the email is already set as.
2. The database location path hard coded in both applications must remain where it is.
3. SQLite is required to access the database.
4. Python3 must be installed.
5. Microsoft .Net framework is required.
6. Age rating images are locally stored, paths to these files are hard-coded.
7. Multiple libraries and dependencies in the following section.

6.1 Dependencies

The prototype has multiple dependencies in order for the application to function as expected, the following table will identify which dependencies are used and what function they play in the application.

Dependencies	Purpose
.NET 6.0 Framework	Core framework for the application to be built upon
BCrypt.Net-Next	Supplies the BCrypt password encryption function
jQuery.Validation	Used to validate forums client sided and provide appropriate error messages
Microsoft.AspNetCore.Razor	Allows the creation of dynamic webpages
Microsoft.EntityFrameworkCore.Sqlite	Accepts SQLite to interact with the entity framework core
Microsoft.AspNetCore.Mvc.Core	Allows the use of the Model View Controller model whilst allowing the inheritance of the “PageModel”
Newtonsoft.Json	Popular framework that makes the interaction with Json objects simpler
Microsoft.AspNet.Web.Optimization	Bundles CSS and JavaScript files efficiently
Bootstrap	Front end framework to create mobile and responsive front end designs
JQuery	JQuery provides additional simplicity to incorporating JavaScript functions

Table 9: Prototype Booking System Dependencies

The following table refers to the dependencies that the QR code scanner, which has been developed in Python, requires.

Dependencies	Purpose
Python3	The selected programming language for the application
Cv2 (opencv-python)	Real-time computer vision, interacting with the camera
Pyzbar	Decodes one dimensional QR codes
Sqlite3	A driver to allow the connection to an SQLite database
Numpy	A library that has been used to easily convert a list of database values to an array
Winsound	Allows the program to access basic built-in Windows sounds

Table 10: Python QR Code Scanner Dependencies

Powerful APIs have been integrated into the system to demonstrate the capabilities of integrating APIs. Integrated APIs are required in order for the application to work as expected, which can be viewed in the following table:

API Endpoint	Function
`https://qrickit.com/api/qr.php?d=yourdata`	Generates API images with data passed to the endpoint
`http://www.omdbapi.com/?apikey=[yourkey]`	Provides a wide range of additional movie details. The API provides the prototype with images for the movies.

Table 11: API Endpoints

Images that are respective of a movie's age rating are stored locally, which must be present in order for them to display in the application. Lastly, the database is stored locally, accessed through a hard-coded file path that must be present for the application, or a new empty database will be created.

During installation and setting up the software it is recommended that users navigate to the read me presented on the GitHub public repository where additional technical instructions are present:

https://github.com/ConnorDillonDev/CinemaTicketBookingSystemPublic#installation-guide

7.0 Conclusion

7.1 Introduction

The project aimed to create a prototype that allowed users to create bookings and scan their booking's QR codes without the need for any staff intervention. All implementations and suggestions for full deployment which have been selected have been supported with proper literature and expert reference to support the reasoning for the implementation.

7.2 The Applications Success

I believe that the prototype will succeed due to its ability to be used as it is whilst also allowing the users to extend the application and database in any use case, they deem fit. All choices have been researched and analysed using qualitative and quantitative analysis techniques to ensure that the most appropriate solutions and tools have been selected to create the prototype. Nielsen’s evaluation is also why I believe the application will succeed, as the prototype complies with all the top ten heuristic evaluations whilst also being designed around the Guttenberg pattern. Lastly, due to vigorous testing and security recommendation, I believe the application is secure and either impossible or extremely difficult to break the application; whilst an agile methodology has been adopted, allowing future issues to be corrected as they arise.

7.3 Conclusion

Ensuring that the prototype has met the use case established in Figures 4 to 7, that it has been designed for as can be seen in chapter 5.4, Testing. All the expected customer, superuser and staff functions are present and working as intended. Although the user interface may not be as refined as was planned in the planning stage, it is essential to note that the application is a prototype, and anyone intending to deploy this in a professional setting would be recommended to apply a theme that is similar to the company’s logos and colour schemes.

Overall, the application has concluded in a manner that I believe is to a high standard. Throughout this paper are recommendations that should be followed in a non-development setting. During the planning stage of the project, many keystone goals were set to ensure that the final prototype and paper met specific requirements, which are as follows:

A comprehensive literature and technical review -

The dissertation describes and investigates literature chapter 2.0. A comprehensive technical review was performed to better understand and establish the most appropriate technology for the tasks that had been previously established in the planning stage.

Fully functional CRUD ticket booking system -

Thoroughly tested, which can be seen in the testing table found in Appendix 1, Figure 15. The testing table demonstrates that the prototype provides multiple functions in terms of CRUD, for example:
Create creating new movies. Read, reading movies from the database. Update, updating customer account details. Delete, removing showings from the database.

Research, design, and implement a database -

Comprehensive research has been performed studying relational and non-relational database models through qualitative and quantitative research techniques performed through analysing peer-reviewed papers and other data sources. Chapter 2.6 refers to database research, whilst chapter 3.5, Database Design, refers to the design of the database. Lastly, implementation can be found in the prototype.

Support customer and administrative authentication -

Three-tier user login system has been implemented. The three-security clearance levels a user can receive from the highest clearance to lowest is superuser, staff, and customer, delegated through cookie assignment, explained in 3.2.2 Secure Login. Page restrictions have also been tested based on the users' cookie clearance, which is assigned at login, shown in Appendix 1, Figure 15, the Testing Table.

The system will be controlled through an intuitive, easy to use, intuitive user interface. Which will be iterated upon following Nielsen's Heuristic evaluation -

Using Nielsen's top ten heuristics described in chapter 2.9 Nielsen's Top Ten Heuristics, designing the user interface has been a primary consideration during the creation of the wireframes and implementation to ensure that a simple and easy to use interface has been created. Following the creation of the wireframes and implementation, the final refinement steps were complete using Nielsen's Heuristic Evaluation, shown in chapter 5.2 Nielsen's Heuristic Evaluation, which provided the appropriate redesign and iterations to ensure that the interface is easy to use, which should convey a positive user experience.

Additional deliverables that were established during the planning stage, which would be described as advanced features of this paper and prototype, are as follows:

Robust system due to testing -

As shown through testing, which can be seen in Appendix 1, Figure 15; the testing table, extensive testing has taken place, including edge, expected, and exceptional data inputs, to ensure the application can handle a range of data inputs. SQL injection attacks have also been attempted on the prototype, which is also viewable in the testing table. Lastly, unexpected custom POST and GET requests were attempted, which can be viewed in chapter 5.4.2, which proves the robustness of the prototype.

Enhanced comprehensive literature and technical review -

As shown through the analysis of multiple peer-reviewed research papers and information gained from Industry giants such as IBM public domains. I believe an enhanced review has been shown in chapter 2.0 Background/Literature Review and chapter 2.7 Technical Reviews.

Public API scraping OR self-created private API implemented on the webpage -

In the prototypes instance, both API endpoints were provided by third parties. Chapter 4.2.8 demonstrates the use of the API "QRickit", which provides the prototype with the ability to generate QR codes. Secondly, QMDbs API provides a wide range of information about movies, used to fetch Movie artwork displayed within the application.

Ticket Creation (Generate and print QR code) QR code scanner -

Using "QRickit" powerful API, QR codes are generated in relation to the cinema booking. A small Python program has been developed to scan the QR codes, which decodes the data and checks for the record in the database, which can be noted in chapter 4.2.10 Camera Scanner Python.

Research, compare and implement the most appropriate database protection measures with justification -

Security has been the focus of the prototype. Whilst the prototype has not been officially deployed and executed in a development environment; many security procedures have been identified, recommended, and implemented. Security importance, recommendations, and implementation can be found in chapter 2.4 Importance of security, chapter 3.6 recommends security solutions for deployed systems and lastly, in terms of implementation, it can be located in chapter 4.2.7 Password Protection Implementation and 4.2.6 Cookies Claims and Page Policy.

The prototype meets all the pre-established criteria noted above; I believe the system is suitable for its intended use case and a base for companies or users to build upon and deploy in a setting of their choice.

8. Critical Appraisal

The project and dissertation proved to be more challenging than expected, whilst also proving to be one of the most valuable assignments I have had to complete during my time at university. During the project's planning, where key deliverables were established, I may have underestimated the amount of time required to complete these tasks in terms of the amount of time I was assigning myself to complete the tasks whilst other modules required my time and attention. However, through dedication and applying soft skills, I was able to deliver a project that meets the criteria stated in the project specification. I have recorded a small preview of the application in use on YouTube, which can be found at: https://www.youtube.com/watch?v=KKyfqA224yQ.

Whilst at university, I have developed using a range of frameworks and tools. Whilst I have worked with the same framework in the past, it was not as advanced or as up to date as in terms of software packages as the project that has been developed. Using these development frameworks in the project required me to expand my understanding of the frameworks whilst also learning new techniques to solve the problems that arose during the project development.

An agile methodology proved extremely helpful during the prototype's development life cycle. Whilst work started to build upon and become worryingly large towards the end of the project. The agile methodology allowed me to revisit and iterate over problems as they were discovered, helping me reduce the amount of corrective work appended to the end of the project development. I feel as though I delegated my time appropriately and stayed on track throughout the project.

During the time of writing the dissertation, COVID-19 has been represented and classed as a worldwide pandemic. COVID-19 proved to create difficulties in time management due to the required isolation times and being unwell. As I was not affected hugely by the virus, I am grateful, whilst it did lead to unusual circumstances such as presenting my PowerPoint from home at my computer as opposed to in person. Whilst the presentation went better than I had expected it would go due to my nerves, technical issues occurred, such as a pre-recorded demonstration was not able to be heard by the audience, which led to me talking over the demonstration, which was improvised. Overall, my grade for the presentation was much better than I expected, and I believe it will reduce my nerves in the future as I know I can present a well-constructed presentation. Another skill I have grown more confident in is my ability to produce academic reports. As I am an individual who has dyslexia, essay writing on a large scale can prove to be difficult. Through the creation of this dissertation, I am more confident that I am just as capable as any other individual of producing a suitable piece of writing.

Lastly, the feedback received during the project's development was invaluable, as it helped me stay on track and query any worries that I had. This feedback was received mainly in terms of weekly meetings, where myself and personal tutor were able to discuss the progress and future plans of the project, meeting notes can be found in Appendix 3, chapter 10.4.3. I feel as though the project has allowed me to push the boundaries, I had set myself in terms of my programming and self-management skills, which has led to a dramatic increase in self-confidence. Whilst I would have enjoyed and preferred to have created a fully deployed and polished application, the time constraints set by the university did not provide enough time to complete this, that is why during the planning, it was decided that the application would be a prototype. The prototype has taught me many things in terms of database development and programming, which I believe will significantly benefit my future endeavours.

9. Bibliography

Amctheatres.com (2022) *AMC Landing Page* [Image]. Available: https://www.amctheatres.com/. (Accessed: 10 February 2022)

Ben-Gan, I. (2022) *Latest Builds of SQL Server 2019* [Online]. SQLPerformance.com. Available: https://sqlperformance.com/latest-builds/sql-server-2019. (Accessed: 29 January 2022)

Browser Market Share Worldwide. (2022) [Online]. StatCounter Global Stats. Available: https://gs.statcounter.com/browser-market-share. (Accessed: 8 November 2022)

Brumen, B. (2019) Security analysis of Game Changer Password System. *International Journal of Human-Computer Studies*. Vol.126, pp.44–52. (Accessed: 20 January 2022)

Chandrababu, A. (2005) A Comparison between Agile and Traditional Soware Development Methodologies. Academia Accelerating the world's research. *School of Computer Science and software Engineering, The University of Western Australia*. [Online]. Available: https://www.academia.edu/38895564/A_Comparison_between_Agile_and_Traditional_Software_Development_Methodologies. (Accessed: 15 February 2022)

Cinemark.com (2022) *Cinemark Landing Page* [Image]. Available: https://www.cinemark.com/movies. (Accessed: 1 December 2021)

Cinepolisindia.com (2022) [Online]. *About - Cinépolis*. Available: https://www.cinepolisindia.com/about-us. (Accessed: 15 January 2022)

Cineworld.com (2022) *Cineworld Landing Page* [Image]. Available: https://www.cineworld.co.uk/#/. (Accessed: 1 February 2022)

Cinipolis.com (2022) *Cinepolis Landing Page* [Image]. Available: https://cinepolis.com/. (Accessed: 5 December 2021)

Cohn, M. (2019) *Scrum Methodology and Project Management* [Online]. Mountain Goat Software. Available: https://www.mountaingoatsoftware.com/agile/scrum. (Accessed: 1 March 2022)

Chen, L. (2016) *AES Development - Cryptographic Standards and Guidelines | CSRC* [Online]. Nist.gov. Available: https://csrc.nist.gov/projects/cryptographic-standards-and-guidelines/archived-crypto-projects/aes-development. (Accessed: 15 January 2022)

Database Security: An Essential Guide. (2022) [Online]. IBM. Available: https://www.ibm.com/uk-en/cloud/learn/database-security#toc-best-pract-v7bPIzDi (Accessed: 28 February 2022)

Dvir, N. and Gafni, R. (2018) When Less Is More: Empirical Study of the Relation Between Consumer Behaviour and Information Provision on Commercial Landing Pages. *Informing Science: The International Journal of an Emerging Transdiscipline*. [Online] Vol.21, pp.019–039. Available: https://www.informingscience.org/Publications/4015. (Accessed: 2 February 2022)

Eastwood, B. (2020) *The 10 Most Popular Programming Languages to Learn in 2020* [Online]. Northeastern University Graduate Programs. Available: https://www.northeastern.edu/graduate/blog/most-popular-programming-languages/. (Accessed: 2 November 2021)

Hasan, J. and Tu, K. (2003) *Performance Tuning and Optimizing ASP.NET Applications* [Online]. Berkeley, CA: Apress. pp. 23. Available: https://doi.org/10.1007/978-1-4302-0758-0. (Accessed: 12 December 2021)

Hcltech.com (2022) *What is do-178b?* [Online]. HCL Technologies
Available: https://www.hcltech.com/technology-qa/what-is-do-178b (Accessed: 18 January 2022)

Hernandez, A. and Resnick, M. L. (2013) Placement of Call to Action Buttons for Higher Website Conversion and Acquisition. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. [Online] Vol.57 (1), pp.1042–1046. Available: https://journals.sagepub.com/doi/pdf/10.1177/1541931213571232. (Accessed: 4 November 2021)

IBM (2019) *What is software testing?* [Online]. IBM. Available: https://www.ibm.com/topics/software-testing. (Accessed: 3 March 2022)

IBM.com (2019) *Database Security: An Essential Guide* [Online]. www.ibm.com. Available: https://www.ibm.com/cloud/learn/database-security. (Accessed: 2 January 2022)

IBM.com (2019) *relational-databases* [Online]. Ibm.com. Available: https://www.ibm.com/cloud/learn/relational-databases. (Accessed: 26 October 2021)

Jatana, N., Puri, S., Ahuja, M., Kathuria, I. and Gosain, D. (2012) *A Survey and Comparison of Relational and Non-Relational Database* [Online]. citeseerx.ist.psu.edu p.1 Available: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.678.9352&rep=rep1&type=pdf. (Accessed: 22 October 2021)

Kumar Upadhyay, R. (2020) *Advantages and Disadvantages of using Spiral Model* [Online]. GeeksforGeeks. Available: https://www.geeksforgeeks.org/advantages-and-disadvantages-of-using-spiral-model (Accessed: 15 February 2021)

Lancor, L. and Katha, S. (2013) Analyzing PHP frameworks for use in a project-based software engineering course. *Proceeding of the 44th ACM technical symposium on Computer science education - SIGCSE '13*, pp.518-519 (Accessed: 29 February 2022)

Li, Y. and Manoharan, S. (2013) A performance comparison of SQL and NoSQL databases. *2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM).* [Online]. Available: https://www.researchgate.net/publication/261079289_A_performance_comparison_of_SQL_and_NoSQL_databases p.17 (Accessed: 24 November 2021)

Lynn, R. (2022) *History of Agile | Planview LeanKit* [Online]. Planview. Available: https://www.planview.com/resources/guide/agile-methodologies-a-beginners-guide/history-of-agile. (Accessed: 1 November 2021)

Mao, X. (2018) *COMPARISON BETWEEN SYMFONY, ASP.NET MVC, AND NODE.JS EXPRESS FOR WEB DEVELOPMENT* [Online]. library.ndsu.edu. North Dakota State University of Agriculture and Applied Science. Available: https://library.ndsu.edu/ir/bitstream/handle/10365/28191/Comparison%20between%20Symfony%2C%20ASP.NET%20MVC%2C%20And%20Node.js%20Express%20for%20Web%20Development.pdf?sequence=1&isAllowed=y/%3E. (Accessed: 1 November 2021)

Maslach, C. (2015) *Burnout : the cost of caring Preview* [Online]. 1st ed. Los Altos, Ca: Malor Books. Available: https://malorbooks.com/pdf/Burnout_preview.pdf. (Accessed: 5 January 2022)

Mattsson, U. T. (2005) *Database Encryption - How to Balance Security with Performance* [Online]. papers.ssrn.com. Available: https://ssrn.com/abstract=670561. (Accessed: 15 January 2022)

Merlo, E., Letarte, D. and Antoniol, G. (2007) SQL-Injection Security Evolution Analysis in PHP. *2007 9th IEEE International Workshop on Web Site Evolution*. (Accessed: 10 January 2022)

Microsoft .NET and .NET Core - Microsoft Lifecycle (2022). [Online]. docs.microsoft. Available: https://docs.microsoft.com/en-us/lifecycle/products/microsoft-net-and-net-core (Accessed: 29 March 2022)

Microsoft, Anderson, R., Brock, D. and Larkin, K. (2022) *Introduction to Razor Pages in ASP.NET Core* [Online]. docs.microsoft.com. Available: https://docs.microsoft.com/en-us/aspnet/core/razor-pages/?view=aspnetcore-6.0&tabs=visual-studio. (Accessed: 19 March 2022)

Microsoft.com (2022) *DbContext Class System.Data.*Entity [Online]. docs.microsoft. Available: https://docs.microsoft.com/en-us/dotnet/api/system.data.entity.dbcontext. (Accessed: 1 March 2022)

Mirzoev, T. and Sack, L. (2013) Webpage Load Speed: ASP.net vs. PHP. i-manager's *Journal on Information Technology*. Vol.2 (2), pp.4–5. (Accessed: 17 January 2022)

Mishra, A. (2014) Critical Comparison Of PHP And ASP.NET For Web Development. *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH.* [Online] Vol.3 (7).p.333 Available: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.590.6590&rep=rep1&type=pdf. (Accessed: 17 January 2022)

MongoDB.com (2019) *NoSQL Databases Explained* [Online]. MongoDB. Available: https://www.mongodb.com/nosql-explained. (Accessed: 25 October 2021)

MySQL.com (2022) *MySQL :: MySQL Server FAQ* [Online]. Mysql. Available: https://www.mysql.com/industry/faq (Accessed: 18 January 2022)

National Center for Education Statistics (2019) *The NCES Fast Facts Tool provides quick answers to many education questions (National Center for Education Statistics)* [Online]. Ed.gov. Available: https://nces.ed.gov/fastfacts/display.asp?id=46. (Accessed: 12 January 2022).

Nielsen, J. (1994) *Heuristic Evaluation: How-To: Article by Jakob Nielsen* [Online]. Nielsen Norman Group. Available: https://www.nngroup.com/articles/how-to-conduct-a-heuristic-evaluation (Accessed: 2 October 2021)

Nielsen, J. (2020) *10 Heuristics for User Interface Design* [Online]. Nielsen Norman Group. Available: https://www.nngroup.com/articles/ten-usability-heuristics/. (Accessed: 29 January 2022)

ONS (2020) *Internet access – households and individuals, Great Britain - Office for National Statistics* [Online]. www.ons.gov.uk. Available: https://www.ons.gov.uk/peoplepopulationandcommunity/householdcharacteristics/homeinternetandsocialmediausage/bulletins/internetaccesshouseholdsandindividuals/2020. (Accessed: 13 February 2022)

Oracle.com (2020) *Database SQL Reference* [Online]. docs.oracle Available: https://docs.oracle.com/cd/B19306_01/server.102/b14200/index.html (Accessed: 1 March 2022)

PostgreSQL.org (2019) *PostgreSQL: About* [Online]. Postgresql.org. Available: https://www.postgresql.org/about/. (Accessed: 19 January 2022)

Reuters.com (2022) [Online]. *AMC - AMC Entertainment Holdings Inc Profile | Reuters.* Available: https://www.reuters.com/companies/AMC. (Accessed: 5 January 2022)

Reuters.com (2022) *CNK.N - Cinemark Holdings, Inc. Profile | Reuters* [Online]. Reuters. Available: https://www.reuters.com/companies/CNK.N (Accessed: 15 January 2022)

Reuters.com. (2022) *CINE.L - Cineworld Group plc Profile | Reuters* [Online]. Reuters. Available: https://www.reuters.com/companies/CINE.L. (Accessed: 12 January 2022)

Rietta, F. (2016) *What is the difference between bcrypt and SHA256?* [Online]. Rietta. Available: https://rietta.com/blog/bcrypt-not-sha-for-passwords/. (Accessed: March 11 2022)

Sharma, P., Ueno, A. and Kingshott, R. (2020) Self-service technology in supermarkets – Do frontline staff still matter? *Journal of Retailing and Consumer Services*. Vol.59, p.102356. (Accessed: 14 January 2022)

Smith, S. and Brock, D. (2022) *Layout in ASP.NET Core* [Online]. docs.microsoft.com. Available: https://docs.microsoft.com/en-us/aspnet/core/mvc/views/layout?view=aspnetcore-6.0. (Accessed: 2 March 2022)

SQL Server 2019—Pricing | Microsoft. (2022) [Online]. Microsoft. Available: https://www.microsoft.com/en-us/sql-server/sql-server-2019-pricing. (Accessed: 18 January 2022)

SQLite.org (2022) [Online]. *SQLite Is Serverless*. Available: https://www.sqlite.org/serverless.html (Accessed: 17 January 2022)

SQLite.org (2022) [Online]. *SQLite: High Reliability*. Available: https://www.sqlite.org/hirely.html (Accessed: 30 March 2022)

Sriramya, P. and Karthika, R. A. (2015) *Providing Password Security By Salted Password Hashing Using Bcrypt Algorithm* [Online]. citeseerx.ist.psu.edu. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1072.20&rep=rep1&type=pdf. p.5552 (Accessed: 14 February 2022)

Stack Overflow Developer Survey 2020. (2020) [Image]. Stack Overflow. Available: https://insights.stackoverflow.com/survey/2020#technology-databases-professional-developers4. (Accessed: 17 January 2022)

Stone, D. L. (2009) *User interface design and evaluation*. 1st ed. Amsterdam: Elsevier, pp 5-7 (Accessed: 10 January 2022)

Ciip.group.cam.ac.uk (2021) [Image]. *The UK Innovation Report 2021*. Available: https://www.ciip.group.cam.ac.uk/uk-innovation-report-2021/. (Accessed: 12 January 2022).

Types of NoSQL Databases. (2019). [Online]. MongoDB. Available: https://www.mongodb.com/scale/types-of-nosql-databases. (Accessed: 25 October 2021)

University, C. M. (2022) *Guidelines for Data Protection - Physical Security - Information Security Office - Computing Services - Carnegie Mellon University* [Online]. cmu.edu. Available: https://www.cmu.edu/iso/governance/guidelines/data-protection/physical-security.html (Accessed: 28 February 2022)

W3.org (2022) *Why Validate?* [Online]. validator.w3.org. Available: https://validator.w3.org/docs/why.html. (Accessed: 13 March 2022)

W3Schools.com (2019) *HTTP Methods GET vs POST* [Online]. W3schools.com. Available: https://www.w3schools.com/tags/ref_httpmethods.asp. (Accessed: 2 March 2022)

Wang, X., Wong, Y. D. and Yuen, K. F. (2021) Does COVID-19 Promote Self-Service Usage among Modern Shoppers? An Exploration of Pandemic-Driven Behavioural Changes in Self-Collection Users. *International Journal of Environmental Research and Public Health*. Vol.18 (16), p.8574. (Accessed: 17 October 2021)

Wolford, B. (2018) *What is GDPR, the EU’s new data protection law?* [Online]. GDPR.eu. Available: https://gdpr.eu/what-is-gdpr. (Accessed: 28 February 2022)

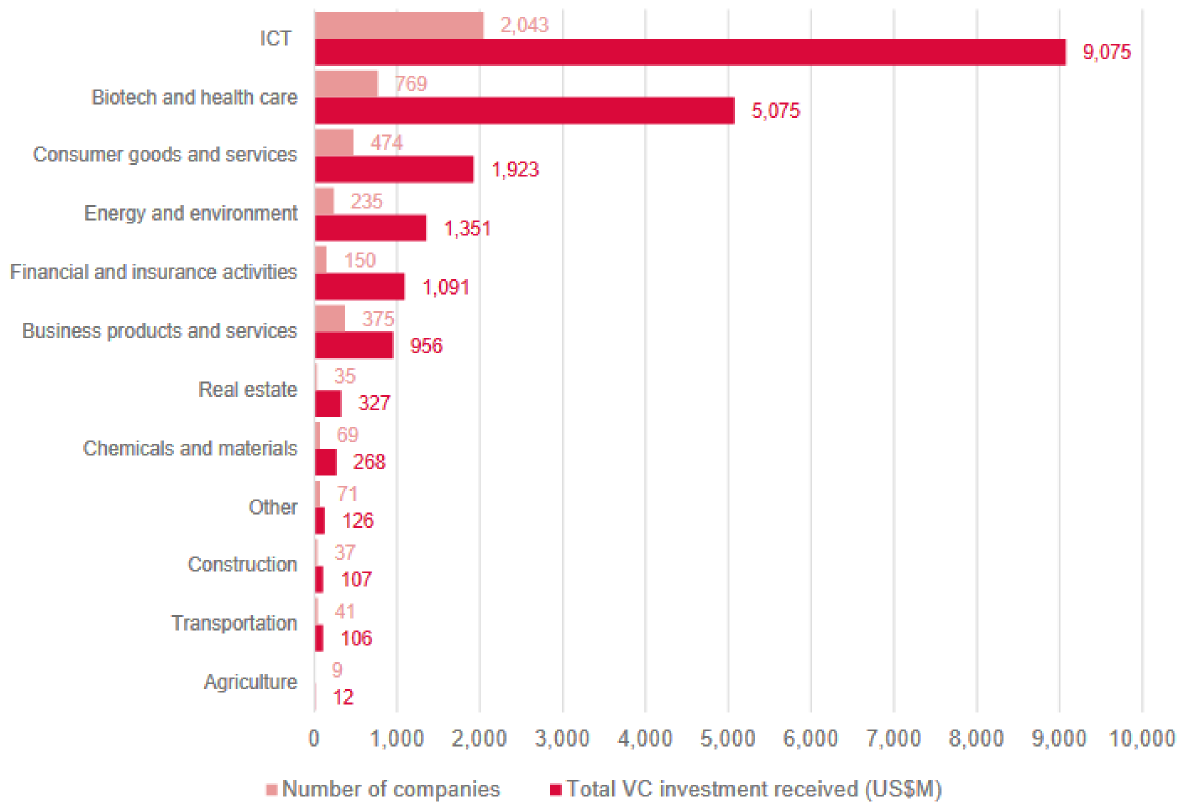
10. Appendices

10.1 Appendix 1

10.1.1 Figure 1: UK innovation Report

Chart 2.14. Venture capital investments in the UK (2)

Sectors by venture capital (VC) funding received, 2007–2019

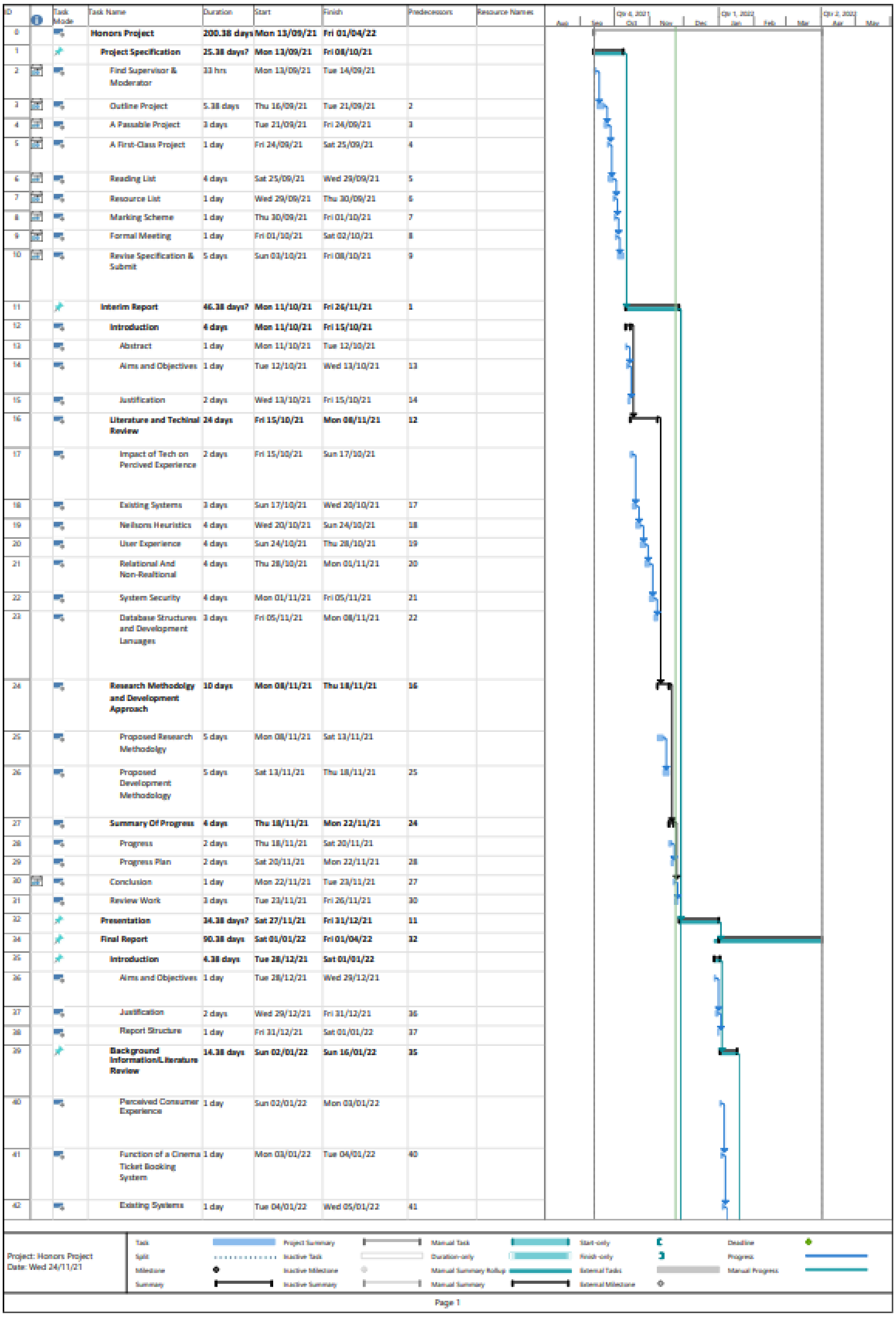


Note: Original values in euros, converted at the annual average nominal exchange rate.
Source: Invest Europe (2020). Activity Report 2007–2019 – European Private Equity.

UK INNOVATION REPORT

(Ciip.group.cam.ac.uk, *2021 p.37)*

10.1.2 Figure 2: GANTT Chart



ID		Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names
0			**Honors Project**	**200.38 days**	**Mon 13/09/21**	**Fri 01/04/22**		
1			**Project Specification**	**25.38 days?**	**Mon 13/09/21**	**Fri 08/10/21**		
2			Find Supervisor & Moderator	33 hrs	Mon 13/09/21	Tue 14/09/21		
3			Outline Project	5.38 days	Thu 16/09/21	Tue 21/09/21	2	
4			A Passable Project	3 days	Tue 21/09/21	Fri 24/09/21	3	
5			A First-Class Project	1 day	Fri 24/09/21	Sat 25/09/21	4	
6			Reading List	4 days	Sat 25/09/21	Wed 29/09/21	5	
7			Resource List	1 day	Wed 29/09/21	Thu 30/09/21	6	
8			Marking Scheme	1 day	Thu 30/09/21	Fri 01/10/21	7	
9			Formal Meeting	1 day	Fri 01/10/21	Sat 02/10/21	8	
10			Revise Specification & Submit	5 days	Sun 03/10/21	Fri 08/10/21	9	
11			**Interim Report**	**46.38 days?**	**Mon 11/10/21**	**Fri 26/11/21**	1	
12			**Introduction**	**4 days**	**Mon 11/10/21**	**Fri 15/10/21**		
13			Abstract	1 day	Mon 11/10/21	Tue 12/10/21		
14			Aims and Objectives	1 day	Tue 12/10/21	Wed 13/10/21	13	
15			Justification	2 days	Wed 13/10/21	Fri 15/10/21	14	
16			**Literature and Techinal Review**	**24 days**	**Fri 15/10/21**	**Mon 08/11/21**	12	
17			Impact of Tech on Percived Experience	2 days	Fri 15/10/21	Sun 17/10/21		
18			Existing Systems	3 days	Sun 17/10/21	Wed 20/10/21	17	
19			Neilsons Heuristics	4 days	Wed 20/10/21	Sun 24/10/21	18	
20			User Experience	4 days	Sun 24/10/21	Thu 28/10/21	19	
21			Relational And Non-Realtional	4 days	Thu 28/10/21	Mon 01/11/21	20	
22			System Security	4 days	Mon 01/11/21	Fri 05/11/21	21	
23			Database Structures and Development Lanuages	3 days	Fri 05/11/21	Mon 08/11/21	22	
24			**Research Methodolgy and Development Approach**	**10 days**	**Mon 08/11/21**	**Thu 18/11/21**	16	
25			Proposed Research Methodolgy	5 days	Mon 08/11/21	Sat 13/11/21		
26			Proposed Development Methodology	5 days	Sat 13/11/21	Thu 18/11/21	25	
27			**Summary Of Progress**	**4 days**	**Thu 18/11/21**	**Mon 22/11/21**	24	
28			Progress	2 days	Thu 18/11/21	Sat 20/11/21		
29			Progress Plan	2 days	Sat 20/11/21	Mon 22/11/21	28	
30			Conclusion	1 day	Mon 22/11/21	Tue 23/11/21	27	
31			Review Work	3 days	Tue 23/11/21	Fri 26/11/21	30	
32			**Presentation**	**34.38 days?**	**Sat 27/11/21**	**Fri 31/12/21**	11	
34			**Final Report**	**90.38 days**	**Sat 01/01/22**	**Fri 01/04/22**	32	
35			**Introduction**	**4.38 days**	**Tue 28/12/21**	**Sat 01/01/22**		
36			Aims and Objectives	1 day	Tue 28/12/21	Wed 29/12/21		
37			Justification	2 days	Wed 29/12/21	Fri 31/12/21	36	
38			Report Structure	1 day	Fri 31/12/21	Sat 01/01/22	37	
39			**Background Information/Literature Review**	**14.38 days**	**Sun 02/01/22**	**Sun 16/01/22**	35	
40			Perceived Consumer Experience	1 day	Sun 02/01/22	Mon 03/01/22		
41			Function of a Cinema Ticket Booking System	1 day	Mon 03/01/22	Tue 04/01/22	40	
42			Existing Systems	1 day	Tue 04/01/22	Wed 05/01/22	41	

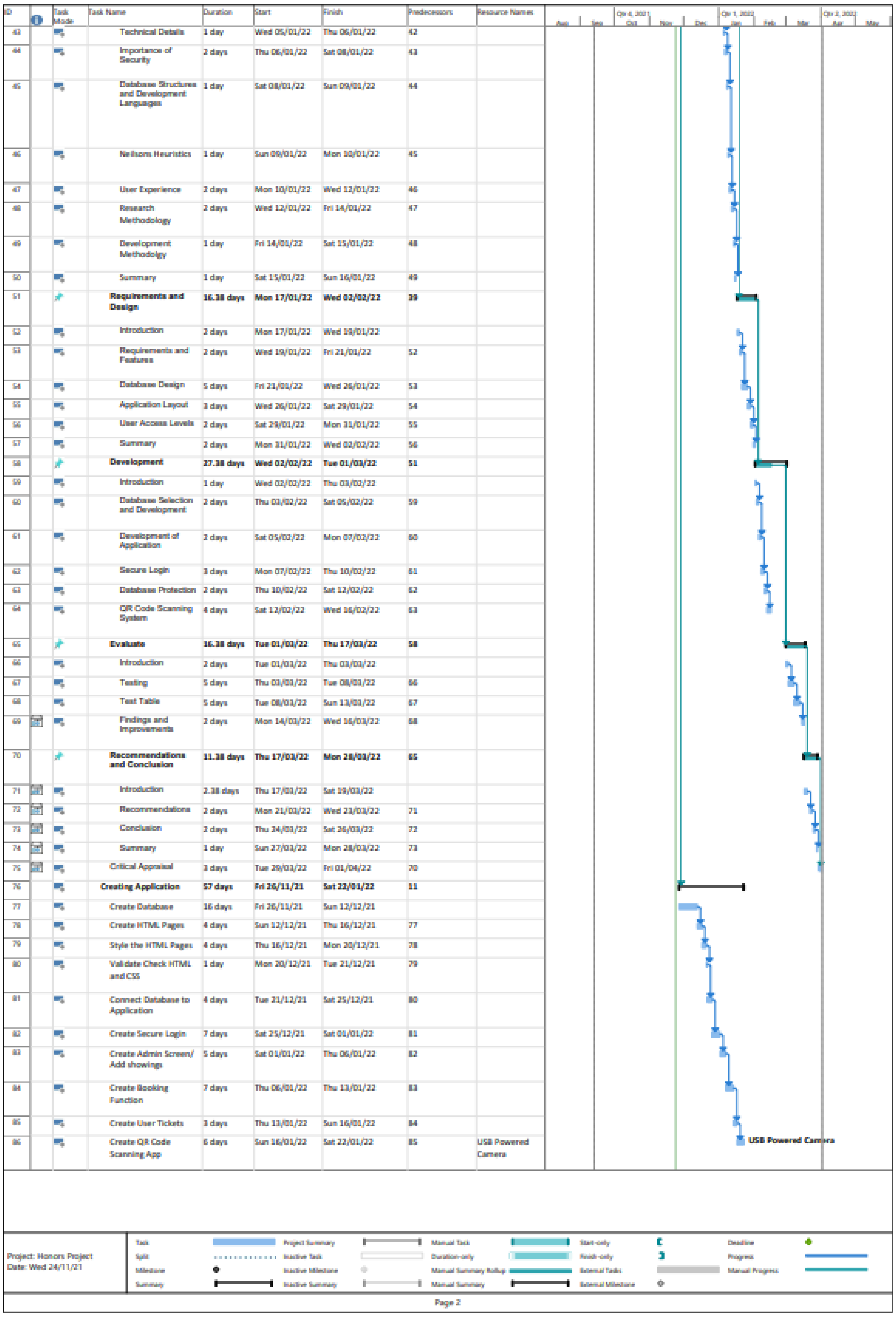
Project: Honors Project
Date: Wed 24/11/21

Task		Project Summary		Manual Task		Start-only		Deadline	
Split		Inactive Task		Duration-only		Finish-only		Progress	
Milestone	●	Inactive Milestone		Manual Summary Rollup		External Tasks		Manual Progress	
Summary		Inactive Summary		Manual Summary		External Milestone			

Page 1

79

10.1.3 Figure 3: GANTT chart Part 2



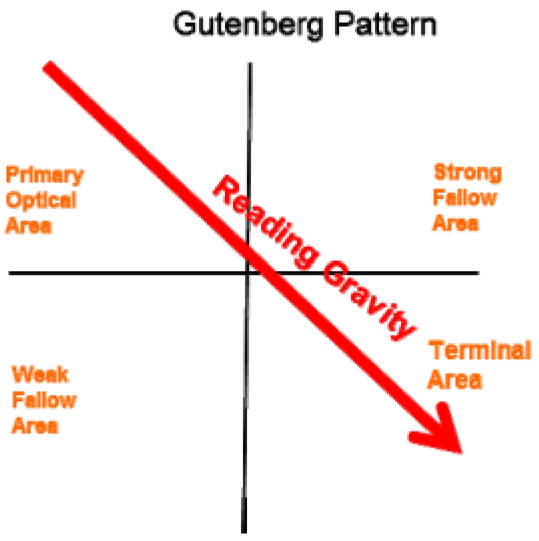
ID	Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names
43		Technical Details	1 day	Wed 05/01/22	Thu 06/01/22	42	
44		Importance of Security	2 days	Thu 06/01/22	Sat 08/01/22	43	
45		Database Structures and Development Languages	1 day	Sat 08/01/22	Sun 09/01/22	44	
46		Neilsons Heuristics	1 day	Sun 09/01/22	Mon 10/01/22	45	
47		User Experience	2 days	Mon 10/01/22	Wed 12/01/22	46	
48		Research Methodology	2 days	Wed 12/01/22	Fri 14/01/22	47	
49		Development Methodolgy	1 day	Fri 14/01/22	Sat 15/01/22	48	
50		Summary	1 day	Sat 15/01/22	Sun 16/01/22	49	
51		**Requirements and Design**	**16.38 days**	**Mon 17/01/22**	**Wed 02/02/22**	**39**	
52		Introduction	2 days	Mon 17/01/22	Wed 19/01/22		
53		Requirements and Features	2 days	Wed 19/01/22	Fri 21/01/22	52	
54		Database Design	5 days	Fri 21/01/22	Wed 26/01/22	53	
55		Application Layout	3 days	Wed 26/01/22	Sat 29/01/22	54	
56		User Access Levels	2 days	Sat 29/01/22	Mon 31/01/22	55	
57		Summary	2 days	Mon 31/01/22	Wed 02/02/22	56	
58		**Development**	**27.38 days**	**Wed 02/02/22**	**Tue 01/03/22**	**51**	
59		Introduction	1 day	Wed 02/02/22	Thu 03/02/22		
60		Database Selection and Development	2 days	Thu 03/02/22	Sat 05/02/22	59	
61		Development of Application	2 days	Sat 05/02/22	Mon 07/02/22	60	
62		Secure Login	3 days	Mon 07/02/22	Thu 10/02/22	61	
63		Database Protection	2 days	Thu 10/02/22	Sat 12/02/22	62	
64		QR Code Scanning System	4 days	Sat 12/02/22	Wed 16/02/22	63	
65		**Evaluate**	**16.38 days**	**Tue 01/03/22**	**Thu 17/03/22**	**58**	
66		Introduction	2 days	Tue 01/03/22	Thu 03/03/22		
67		Testing	5 days	Thu 03/03/22	Tue 08/03/22	66	
68		Test Table	5 days	Tue 08/03/22	Sun 13/03/22	67	
69		Findings and Improvements	2 days	Mon 14/03/22	Wed 16/03/22	68	
70		**Recommendations and Conclusion**	**11.38 days**	**Thu 17/03/22**	**Mon 28/03/22**	**65**	
71		Introduction	2.38 days	Thu 17/03/22	Sat 19/03/22		
72		Recommendations	2 days	Mon 21/03/22	Wed 23/03/22	71	
73		Conclusion	2 days	Thu 24/03/22	Sat 26/03/22	72	
74		Summary	1 day	Sun 27/03/22	Mon 28/03/22	73	
75		Critical Appraisal	3 days	Tue 29/03/22	Fri 01/04/22	70	
76		**Creating Application**	**57 days**	**Fri 26/11/21**	**Sat 22/01/22**	**11**	
77		Create Database	16 days	Fri 26/11/21	Sun 12/12/21		
78		Create HTML Pages	4 days	Sun 12/12/21	Thu 16/12/21	77	
79		Style the HTML Pages	4 days	Thu 16/12/21	Mon 20/12/21	78	
80		Validate Check HTML and CSS	1 day	Mon 20/12/21	Tue 21/12/21	79	
81		Connect Database to Application	4 days	Tue 21/12/21	Sat 25/12/21	80	
82		Create Secure Login	7 days	Sat 25/12/21	Sat 01/01/22	81	
83		Create Admin Screen/ Add showings	5 days	Sat 01/01/22	Thu 06/01/22	82	
84		Create Booking Function	7 days	Thu 06/01/22	Thu 13/01/22	83	
85		Create User Tickets	3 days	Thu 13/01/22	Sun 16/01/22	84	
86		Create QR Code Scanning App	6 days	Sun 16/01/22	Sat 22/01/22	85	USB Powered Camera

Project: Honors Project
Date: Wed 24/11/21

Task		Project Summary		Manual Task		Start-only		Deadline	
Split		Inactive Task		Duration-only		Finish-only		Progress	
Milestone	◆	Inactive Milestone	◇	Manual Summary Rollup		External Tasks		Manual Progress	
Summary		Inactive Summary		Manual Summary		External Milestone	◆		

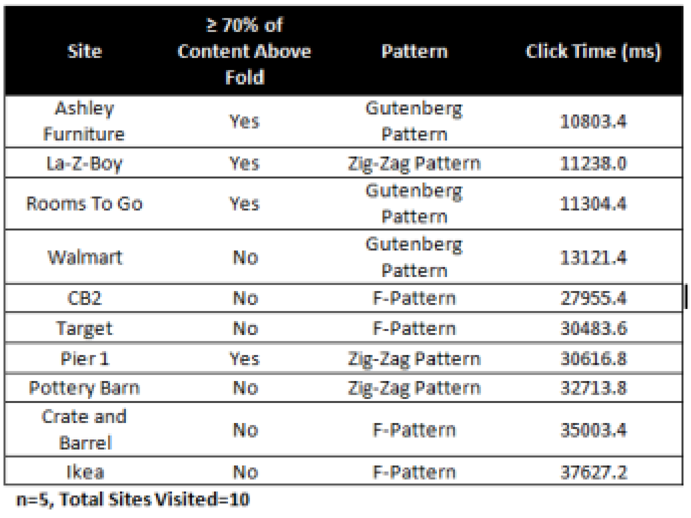
Page 2

10.1.4 Figure 4: Gutenberg Pattern



(Hernandez and L Resnick, 2022)

10.1.5 Figure 5: Results of Study



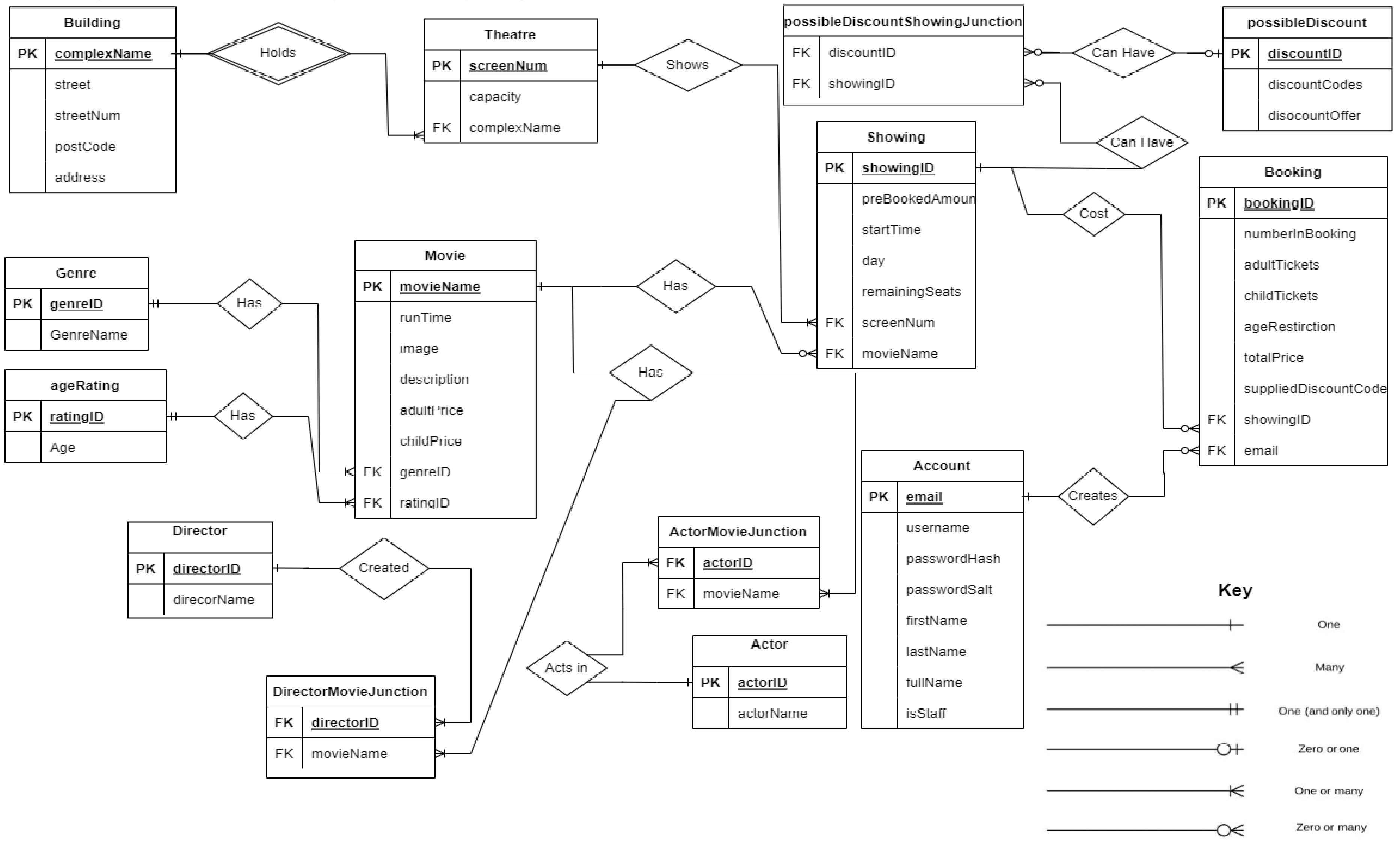
Site	≥ 70% of Content Above Fold	Pattern	Click Time (ms)
Ashley Furniture	Yes	Gutenberg Pattern	10803.4
La-Z-Boy	Yes	Zig-Zag Pattern	11238.0
Rooms To Go	Yes	Gutenberg Pattern	11304.4
Walmart	No	Gutenberg Pattern	13121.4
CB2	No	F-Pattern	27955.4
Target	No	F-Pattern	30483.6
Pier 1	Yes	Zig-Zag Pattern	30616.8
Pottery Barn	No	Zig-Zag Pattern	32713.8
Crate and Barrel	No	F-Pattern	35003.4
Ikea	No	F-Pattern	37627.2

n=5, Total Sites Visited=10

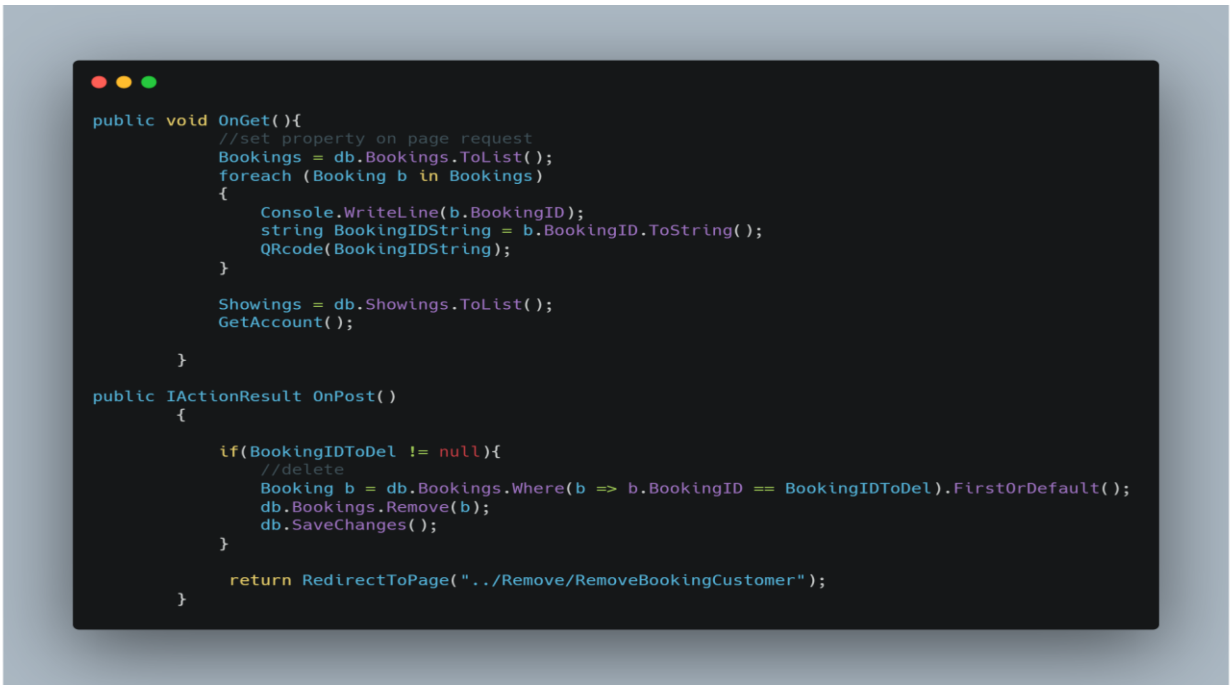
Performance of the 10 Furniture Websites Used in the Study

(Hernandez and L Resnick, 2022)

10.1.6 Figure 6: Enhanced Entity Relationship Diagram

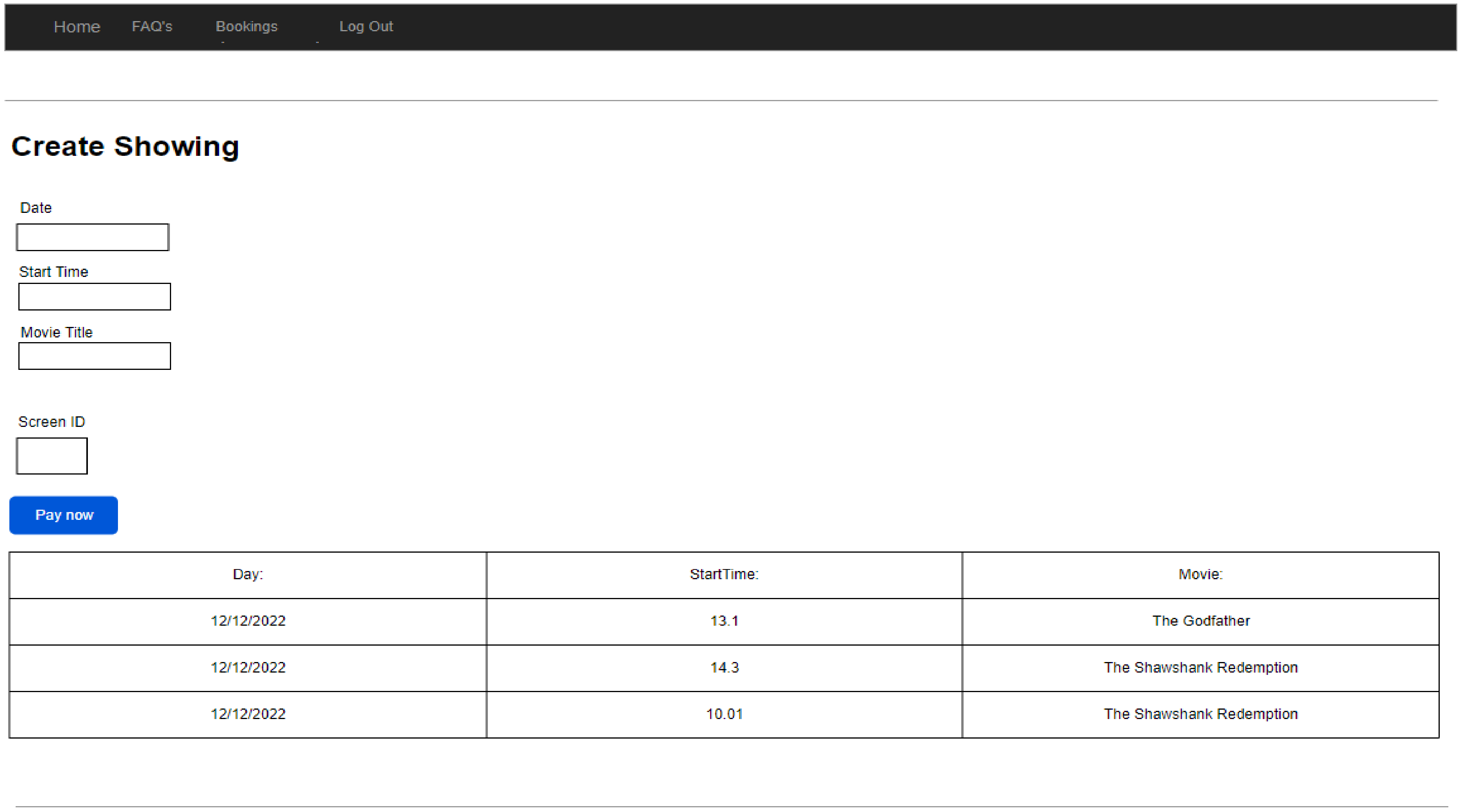


10.1.7 Figure 7: Booking Post and Get Requests



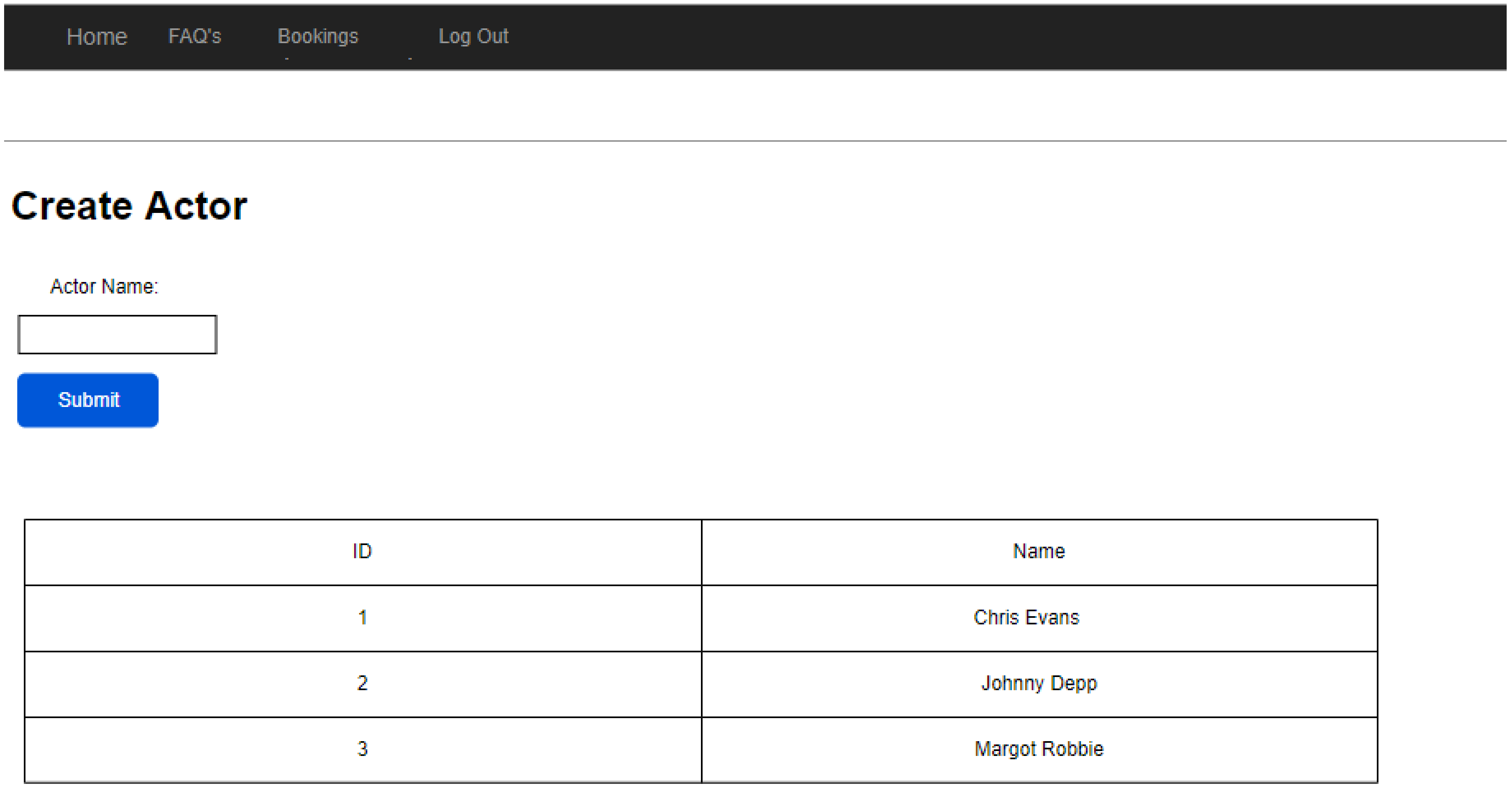
```
public void OnGet(){
            //set property on page request
            Bookings = db.Bookings.ToList();
            foreach (Booking b in Bookings)
            {
                Console.WriteLine(b.BookingID);
                string BookingIDString = b.BookingID.ToString();
                QRcode(BookingIDString);
            }

            Showings = db.Showings.ToList();
            GetAccount();

        }

public IActionResult OnPost()
        {

            if(BookingIDToDel != null){
                //delete
                Booking b = db.Bookings.Where(b => b.BookingID == BookingIDToDel).FirstOrDefault();
                db.Bookings.Remove(b);
                db.SaveChanges();
            }

             return RedirectToPage("../Remove/RemoveBookingCustomer");
        }
```

10.1.8 Figure 8: Create Showing Wireframe



Day:	StartTime:	Movie:
12/12/2022	13.1	The Godfather
12/12/2022	14.3	The Shawshank Redemption
12/12/2022	10.01	The Shawshank Redemption

10.1.9 Figure 9: Create Actor Wireframe

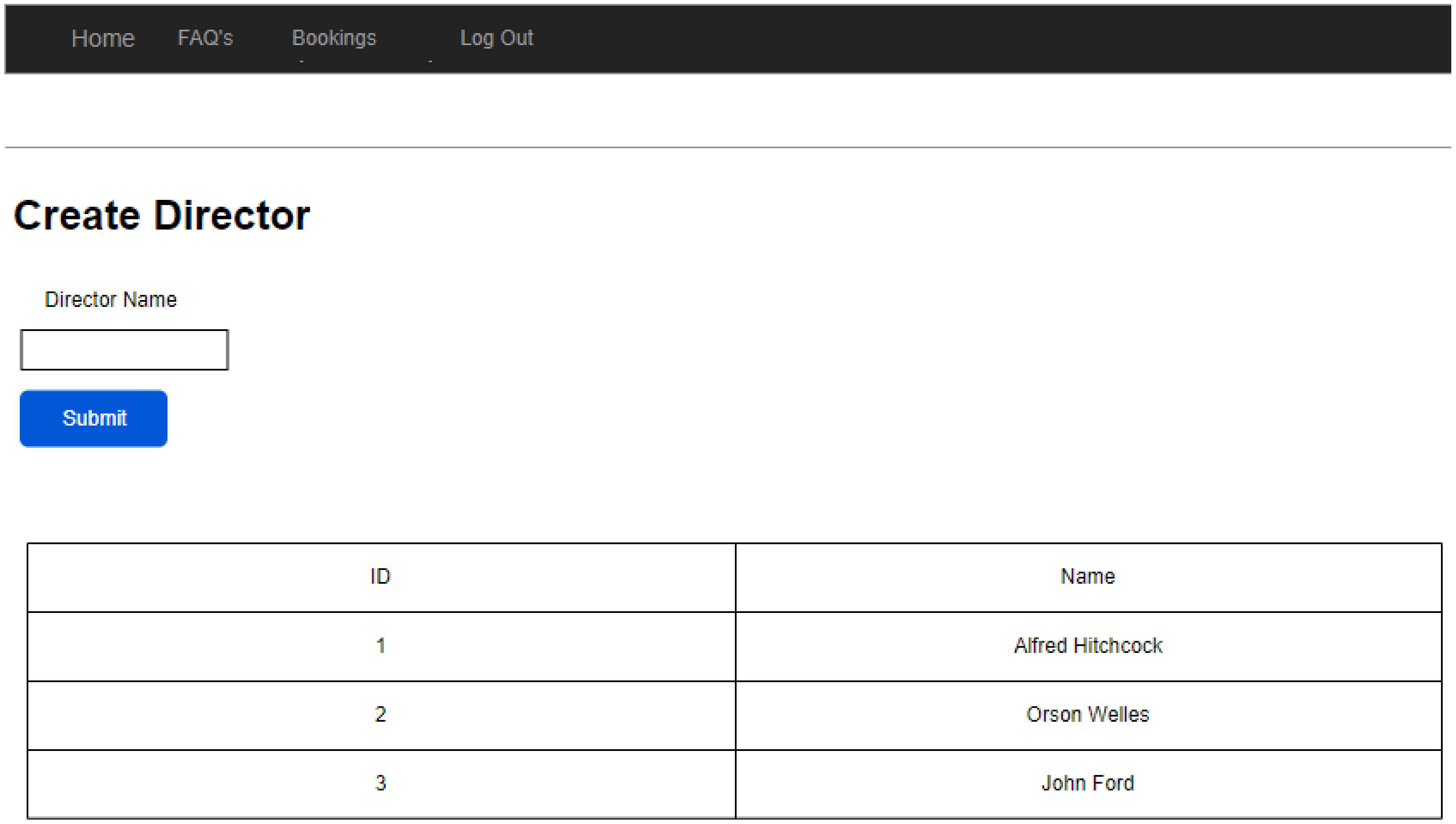


Create Actor

Actor Name:

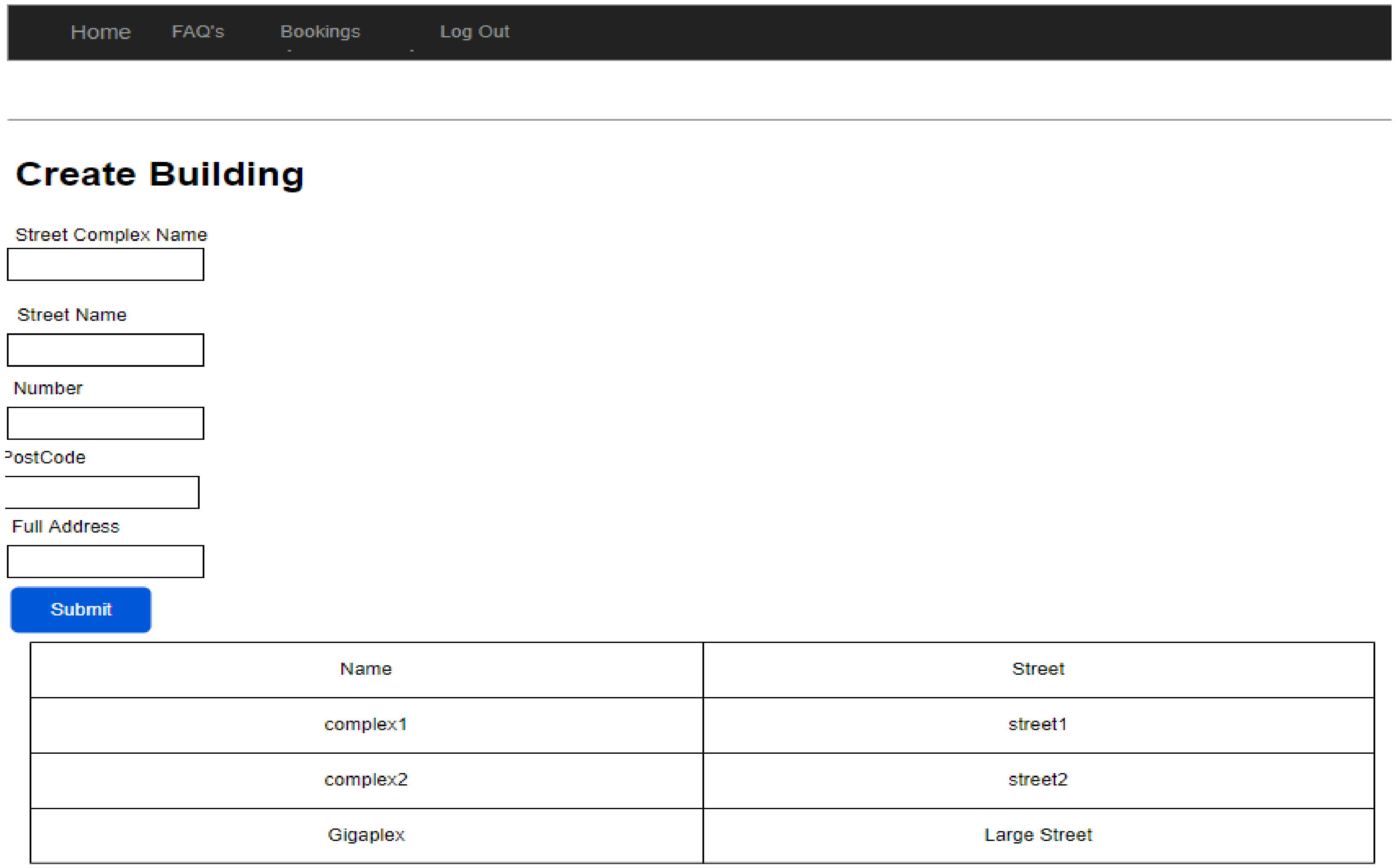
ID	Name
1	Chris Evans
2	Johnny Depp
3	Margot Robbie

10.1.10 Figure 10: Create Director Wireframe



ID	Name
1	Alfred Hitchcock
2	Orson Welles
3	John Ford

10.1.11 Figure 11: Create Building Wireframe



Create Building

Street Complex Name

Street Name

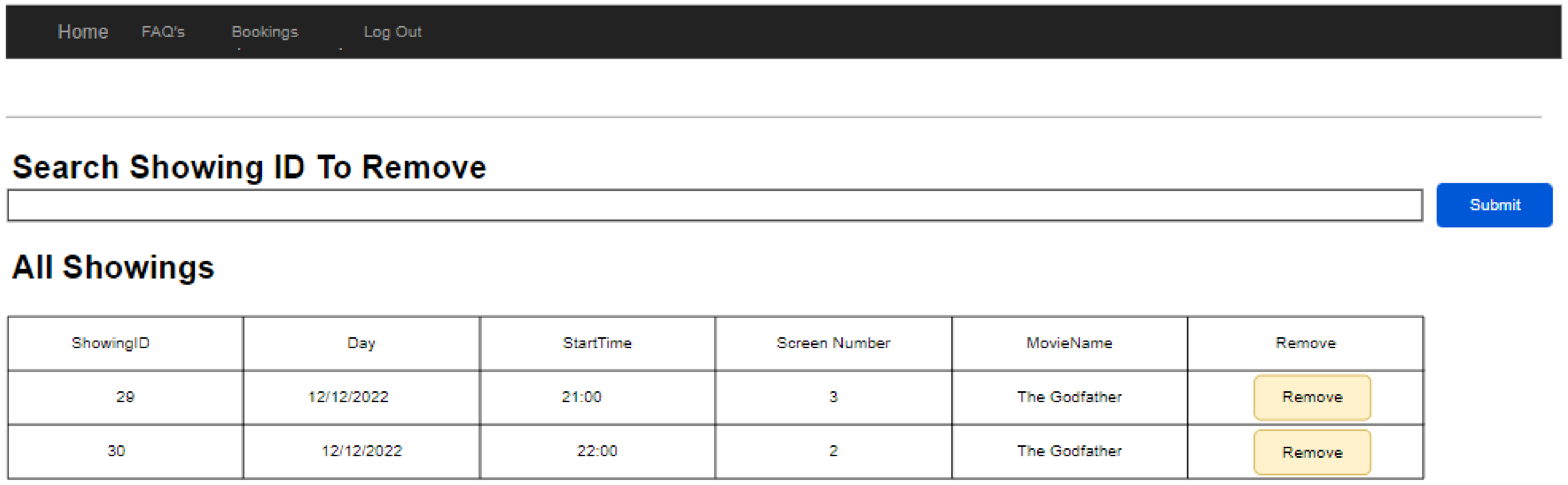
Number

PostCode

Full Address

Name	Street
complex1	street1
complex2	street2
Gigaplex	Large Street

10.1.12 Figure 12: Remove Showing Wireframe

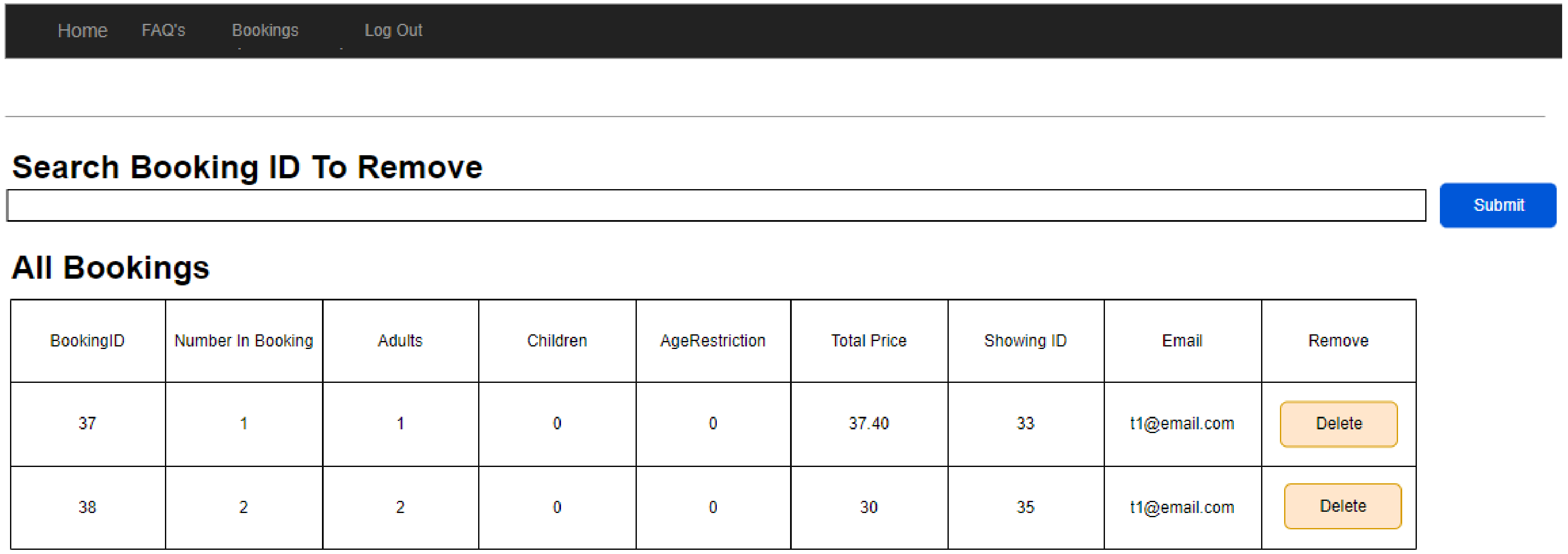


Search Showing ID To Remove

All Showings

ShowingID	Day	StartTime	Screen Number	MovieName	Remove
29	12/12/2022	21:00	3	The Godfather	Remove
30	12/12/2022	22:00	2	The Godfather	Remove

10.1.13 Figure 13: Remove Booking Wireframe

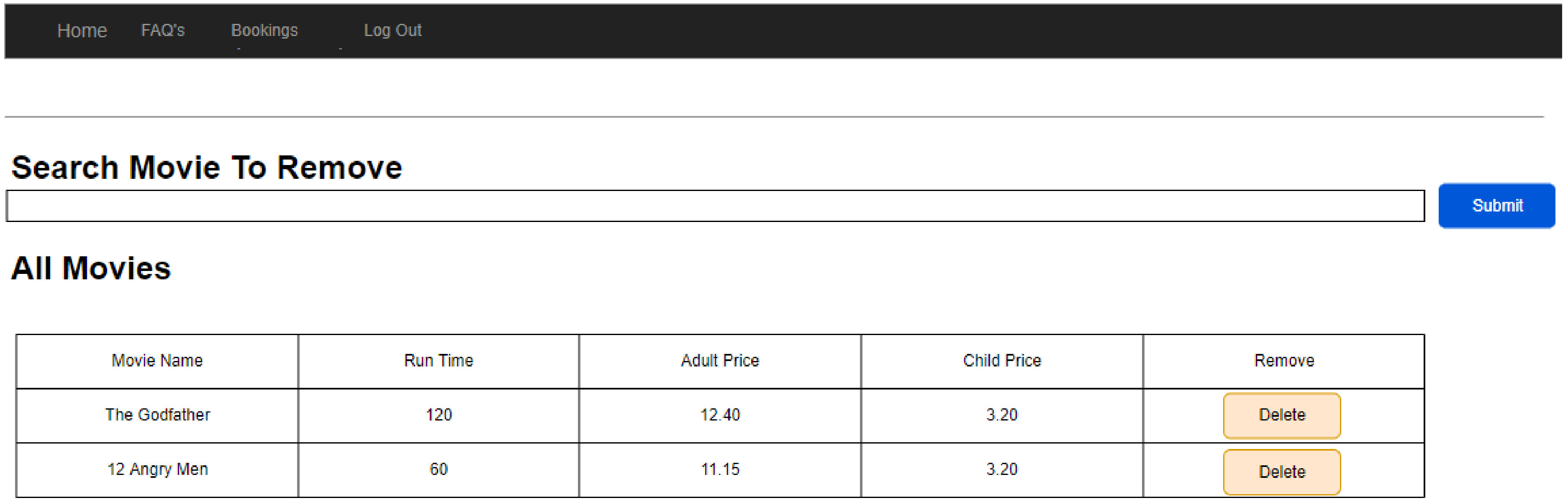


Search Booking ID To Remove

All Bookings

BookingID	Number In Booking	Adults	Children	AgeRestriction	Total Price	Showing ID	Email	Remove
37	1	1	0	0	37.40	33	t1@email.com	Delete
38	2	2	0	0	30	35	t1@email.com	Delete

10.1.14 Figure 14: Remove Movie Wireframe



Search Movie To Remove

All Movies

Movie Name	Run Time	Adult Price	Child Price	Remove
The Godfather	120	12.40	3.20	Delete
12 Angry Men	60	11.15	3.20	Delete

10.1.15 Figure 15: Testing Table

Key:

E = Edge (Data that is at the edge of expected inputs)

N = Expected (Normal and the most common type of data)

X = Exceptional (Data that is not expected)

Case	Test Title	Test Steps	Test Data	Expected Result	Actual Result	Status	Fixed
E	Create Account	Create account and input two different passwords	input1 = password input2 = Password	Passwords do not match	Passwords do not match	✓	
N	Create Account	Create account normally	email: email@gmail.com name:connor dillon username: cd password: passsword	Account Successfully created	Account Successfully created	✓	
X	Create Account SQL Injection attempt	Create Account, SQL injection	email: tz@gmail.com name: 105 OR 1=1 username: SQL password: passsword	Account submitted, injection failure.	Account submitted, injection failure.	✓	

X	Create Account with the same unique identifier PK	Use the same email twice	Email:connor@gmail.com	Save to database	Unhandled error	X	Added additional error handling
E	Login	Login with non-registered email	email: noemail@gmail.com password: password	Access denied page	Access Denied Page	✓	
N	Login	Login with registered email	email: email@gmail.com password: password	Proceed to application	Proceed to application	✓	
X	Login	Login with SQL code attempt	email:' or 1=1 – password:password	Must contain “@” in email	Must contain “@” in email	✓	
N	Select Showing Time	Select a time from the film	movie: the godfather time:14:56	Taken to booking page	Taken to booking page	✓	
E	Select Tickets	Select tickets at 0 for both	tickets:0 adult, 0 children	Input must be greater than 0	proceeds to show booking confirmation	X	Corrected, if child's film one child should be preset, else if adult film one adult will be selected
N	Normal ticket selection	select 1 adult for batman	tickets: adult 1	Confirm booking page	Confirm booking page	✓	

X	Select 18+ Ticket	Dark Knight, aged 18 & select children ticket	tickets: 0 adults, 1child	Child tickets should not be visible	Child tickets not be visible	✓	
N	Cancel booking	Remove booking	remove the godfather booking	Booking should be removed and displayed new list of customer bookings	Worked as expected	✓	
E	Staff account tries to manage accounts	login to t1@email.com, and attempt to access account management	email:t1@email.com password:password	access denied as restricted to superuser account	Worked as expected	✓	
N	Superuser Account access manage accounts	Log into superuser and attempt to access account management	email:SuperUser996@cinema.com password:password	Access granted	Access granted	✓	
E	Customer account tries to access staff protected pages	Login to customer account and attempt to access account management or any staff pages such as create movies	email:old@gmail.com password:password	Access denied as restricted to Staff account	Worked as expected	✓	
N	Staff account attempts to access staff protected pages	log into staff account and attempt to access all staff protected pages	email:t1@email.com password:password	Access granted	Access granted	✓	
N	Create Movie	Create a Movie using the staff account	Movie: The Imitation Game Runtime: 120 Director: Alfred	Added to the table. Image automatically added to the listing in index	Worked as expected	✓	

			Actors: Chris Evans Adult:12.40 Child:4.20 Genre:Thriller Age:15 Description: The imitation games description				
E	Create Movie with spelling mistake in title	Create a movie with a misspelt title	Movie: The pipitation Game Runtime: 120 Director: Alfred Actors: Chris Evans Adult:12.40 Child:4.20 Genre:Thriller Age:15 Description: The imitation games description	Added to table, may affect the image that is added to the listing through API	Index page throwing: IndexOutOfRangeException: Index was outside the bounds of the array.	X	Added image not found text. When adding movies, staff should ensure the film has been added properly
E	Create Movie that is already present in the database	Using the same PK so create another duplicate	Movie: The Imitation Game Runtime: 120 Director: Alfred Actors: Chris Evans Adult:12.40 Child:4.20	Error message correctly displayed	Works as intended	✓	

			Genre:Thriller Age:15 Description: The imitation games description				
X	Create Movie that does not exist	Create a movie which name does not exist	Movie: 123214!!	Image not found displayed in index.	Image not found displayed in index.	✓	
N	Create Showing	Create a standard movie showing	Date: 10/10/2022 start time: 13.10 Movie: Shawshank ScreenID: 1	Added to table successfully	Works as intended	✓	
E	Create Showing	Create a showing with a date that is in the past	Date:10/10/1990 start time: 12:00 Movie: ShawShank ScreenID:2	Error displayed that date has passed	Showing is successfully submitted	X	Implemented JQUERY date picker with minimum date
X	Create Showing	Create a showing with no date selected	Date: start time: 12.20 Movie: ShawShank ScreenID:2	Error displayed missing input	submits to database	X	Error message added and prevent database submission if blank
N	Create Actor	Create an Actor	name : Katharine Hepburn	Successfully added to the table and database	Works as expected	✓	

E	Create no named Actor	Create an Actor with no name	name :	Cannot submit empty field error	Works as expected	✓	
X	Create actor SQL	Create an actor using SQL injection	name: ' or 1=1 –	Submit as regular actor name, without compromising the database	Works as expected	✓	
N	Create Director	Create Director	name: FEDERICO FELLINI	Successfully added to the table and database	Works as expected	✓	
E	Create no named Director	Create Director with blank name	name:	Cannot submit empty field error	Works as expected	✓	
X	Create SQL Director	Create Director with SQL injection	name: ' or 1=1 –	Submit as regular director name, without compromising the database	Works as expected	✓	
N	Manage Bookings	Search booking ID 32 and delete	search ID: 32	Successfully deleted and clearly removed from the list	Works as expected	✓	
E	Manage Booking	Search for an already deleted booking	search ID:32	No results displayed	Works as expected	✓	
X	Manage Booking	Search for negative number ID of booking	search ID: -1	Must be a positive number message	Works as expected	✓	
N	Manage Showings	Remove showing ID 25	search ID: 25	Successfully removed from the list	Works as expected	✓	

E	Manage Showings	Search for an already deleted showing	search ID: 25	Successfully removed from the list	Works as expected	✓	
X	Manage Showings	Search for negative showing ID of booking	search ID: -1	Must be a positive number Must be a positive number message	Works as expected	✓	
N	SuperUser Account Management	Update user account password	email: old@gmail.com password:letmein	updates password successfully	password invalid on login	X	Updated password Bcrypt logic
E	SuperUser Account Management	Submit all fields as blank	all fields:	Display error	NOT NULL constraint error	X	Add empty field validation to all input fields.
X	SuperUser Account Management Attack	Name submitted as SQL attack	name: ' or 1=1 –	Submit without compromising database	Works as expected	✓	
N	SuperUser Account Management	Change name	Name: connor2	Submit and only update name	Hashed password was hashed again, locking account	X	Updated logic for account management.
N	Python, scan a QR code	Scan a booking in a user account	QR code	Beep and present the booking in the database if present	Works as expected	✓	
E	Python, scan a booking for a cancelled showing	Scan a booking for a showing that has been cancelled	QR Code	QR code should not be shown	FK constrain failed	X	Added cascade on delete to FK for ShowingID in booking

X	Scan a SQL malicious code QR code	Scan a QR code which would demonstrate an exploit in the codebase.	QR code = <script>alert(“test”)</script>	No response from the program	Works as expected	✓	
N	Create Building	Standard building creation	Name:trioplex Street: wallaceST Number: 12 Postcode: PA12UR	Added Successfully, displayed with confirmation message	Works as expected	✓	
E	Create a Building with the post code test	Create a Building with the post code containing a additional string	Name:trioplex2 Street: wallaceST Number: 12 Postcode: PA12UR UNIT2	Added Successfully, displayed with confirmation message	Works as expected	✓	
X	Creating a building with duplicate primary key	Creating a building with the same complex name	Name:trioplex Street: wallaceST Number: 12 Postcode: PA12UR	Warning message displayed and not added to the database	Works as expected	✓	
N	Create a theatre	Creating a theatre	Screen Number:10 Complex = complex1	Successfully added message	Works as expected	✓	
E	Create a theatre with the same screen	Using the same screen number for creation (which is the PK)	Screen Number:10 Complex = complex1	Warning message, not added	Works as expected	✓	
X	Negative screen	Create theatre with a	Screen Number:-	Not possible in range	Submitted	X	Created range

	number	negative screen number	10 Complex = complex1				logic on the input
N	Remove Theatre	Remove screen number 1	Screen number:10	Successfully removed, with message	Works as expected	✓	
E	Search for theatre	Search for theatre that does not exist	Screen Number: 10	Not found message	Works as expected	✓	
X	Remove negative Screen Number	Editing the html client side to accept negative numbers and attempt to search for it	Screen Number: -1	Not found message	Works as expected	✓	
N	Search and remove	Search for screen and then remove it from the search results	Screen Number: 9	Successfully removed, with success message	Works as expected	✓	
E	Check that the FAQ works	Click on the FAQ section in the navbar	N/A	Successfully loaded and displaying data	Works as expected	✓	
E	Logout working	Click on logout and check browser cookies to ensure the “MyCookieAuth” has been removed	N/A	Successfully removed and redirected to login page	Works as expected	✓	
E	Remove the pipitation games	Manage movies and remove The pipitation Game	Movie Name: The pipitation Game	Successfully removed and redirected to login page	Works as expected	✓	

10.3 Appendix 2: Data Dictionary

Buildings

Column Name	Meaning	Data Type	Nulls	PK / FK	Default Value	Constraint /Column Property (including Data Classification)
complexName	Uniquely identifies which site holds the theater.	varchar(40)	No	PK	N/A	(Public)
street	Name of the street the complex is located.	varchar(40)	No		N/A	(Public)
streetNum	Number of the property on the street	smallint(10000)	No		N/A	(Public)
postCode	A worldwide recognized location code for the property.	varchar(20)	No		N/A	(Public)
address	Full address: street name, number, postcode and complex name	varchar(200)	No		N/A	(Public) (Computed) complexName + street+streetNum + postCose

Theatres

Column Name	Meaning	Data Type	Nulls	PK / FK	Default Value	Constraint /Column Property (including Data Classification)
screenNum	Uniquely identifies a showing via screen number.	smallint(700)	No	PK	N/A	(Public)
capacity	Capacity of seats in the theater.	smallint(500)	No		100	(Public)
complexName	Uniquely identifies which site holds the theater.	varchar(40)	Yes	FK	N/A	(Public) ON UPDATE CASCADE ON DELETE NO ACTION

Showings

Column Name	Meaning	Data Type	Nulls	PK / FK	Default Value	Constraint /Column Property (including Data Classification)
showingID	Uniquely identifies which showing is being played.	INTEGER	No	PK	N/A	(Public) (Computed) Set identity specification seed as 1 and increment as 1.
preBookedAmount	Capacity of seats in the theater that are prebooked.	smallint(500)	No		N/A	(Public)
day	The date/day that the showing is due to play	nvarchar(100)	Yes		N/A	(Public)
startTime	What time the showing starts.	decimal(10,2)	Yes		N/A	(Public)
screenNum	Uniquely identifies a showing via screen number.	smallint(500)	No	FK	N/A	(Public) ON UPDATE CASCADE ON DELETE NO ACTION
remaingSeats	Amount of seats available after including bookings	smallint(100)	No		100	(Public)
movieName	Movie name	varchar(60)	No	FK	N/A	(Public) ON UPDATE CASCADE ON DELETE NO ACTION

Movies

Column Name	Meaning	Data Type	Nulls	PK / FK	Default Value	Constraint /Column Property (including Data Classification)
movieName	Uniquely identifies the movie via name.	varchar(60)	No	PK	N/A	(Public)
runTime	How long the movie lasts.	smallint(500)	No		N/A	(Public) Stored in minutes
image	Image of the movie to be used as a thumbnail	blob()	No		N/A	(Public)
description	Small description of the movie's plot.	nvarchar(200)	No		N/A	(Public)
adultPrice	Price of one adult ticket	decimal(10,2)	No		12.40	(Public)
childPrice	Price of one child ticket	decimal(10,2)	Yes		4.20	(Public)

genreID	genreID linking to the genre of the movie in the separate table	INTEGER	No	FK	N/A	(Public) ON UPDATE CASCADE ON DELETE NO ACTION
ratingID	ratingID links to the age rating table to identify the age rating of the movie	INTEGER	No	FK	N/A	(Public) ON UPDATE CASCADE ON DELETE NO ACTION

Directors

Column Name	Meaning	Data Type	Nulls	PK / FK	Default Value	Constraint /Column Property (including Data Classification)
directorID	Uniquely identifies the director in the table via the unique id.	INTEGER	No	PK	N/A	(private) (Computed) Set identity specification seed as 1 and increment as 1.
directorName	String literal name of the director	varchar(20)	No		N/A	(Public)

DirectorsMovieJunction

Column Name	Meaning	Data Type	Nulls	PK / FK	Default Value	Constraint /Column Property (including Data Classification)
directorID	Uniquely identifies the director in the table via the unique id.	INTEGER	No	PK	N/A	(private) ON UPDATE CASCADE ON DELETE CASCADE
movieName	String literal name of the director	varchar(20)	No	PK	N/A	(Public) ON UPDATE CASCADE ON DELETE CASCADE

Actors

Column Name	Meaning	Data Type	Nulls	PK / FK	Default Value	Constraint /Column Property (including Data Classification)
actorID	Uniquely identifies the actor in the table via the unique id.	INTEGER	No	PK	N/A	(Public) (Computed) Set identity specification seed as 1 and increment as 1. CASCADE ON DELETE & ON UPDATE
actorName	String literal name of the actor	varchar(20)	No		N/A	(Public)

ActorsMovieJunction

Column Name	Meaning	Data Type	Nulls	PK / FK	Default Value	Constraint /Column Property (including Data Classification)
actorID	Uniquely identifies the actor in the table via the unique id.	INTEGER	No	PK	N/A	(Public)
movieName	String literal name of the movie title	varchar(20)	No	PK	N/A	(Public)

Genres

Column Name	Meaning	Data Type	Nulls	PK / FK	Default Value	Constraint /Column Property (including Data Classification)
genreID	Uniquely identifies the genre in the table via the unique id.	INTEGER	No	PK	N/A	(Public) (Computed) Set identity specification seed as 1 and increment as 1.
Name	Understandable meaning of genre such as “action”	varchar(20)	No		N/A	(Public)

AgeRatings

Column Name	Meaning	Data Type	Nulls	PK / FK	Default Value	Constraint /Column Property (including Data Classification)
ratingID	Uniquely identifies the age rating in the table via the unique id.	smallint(5)	No	PK	N/A	(Public) (Computed) Set identity specification seed as 1 and increment as 1.

Age	Understandable meaning of age rating such as “18”	varchar(5)	No		N/A	(Public)

Bookings

Column Name	Meaning	Data Type	Nulls	PK / FK	Default Value	Constraint /Column Property (including Data Classification)
bookingID	Uniquely identifies the booking.	INTEGER	No	PK	N/A	(Public) (Computed) Set identity specification seed as 1 and increment as 1.
numberInBooking	Amount of seats reserved in the booking	smallint(20)	No		N/A	(Public) (Computed) adultTickets + childTickets (Constraint) adultTickets + childTickets < 21. Maximum tickets 20 per adult selection or child selection.
adultTickets	Amount of adults attending booking.	smallint(20)	Yes		N/A	(Public)
childTickets	Amount of children attending the booking.	smallint(20)	Yes		N/A	(Public)
ageRestriction	Check to ensure party are old enough for showing.	smallint(2)	No			(Public) (Computed) If childTickets & (ageRating > 12) then 1 else 0 If a child is trying to book for a movie that has an age rating of > 12 then flag.
totalPrice	Price of all bookings	smallint(1000)	No		N/A	(Public) (Computed) adultTickets * adultPrice + childTickets * childPrice
suppliedDiscountCode	The user entered discount code	CHAR(8)	Yes			(Public)
showingID	Unique identifier of the showing.	INTEGER	No	FK		(Public)

						ON UPDATE CASCADE ON DELETE CASCADE
email	Customer accounts email address	nvarchar(30)	No	FK		(Public) ON UPDATE CASCADE ON DELETE NO ACTION

Accounts

Column Name	Meaning	Data Type	Nulls	PK / FK	Default Value	Constraint /Column Property (including Data Classification)
email	Uniquely identifies the user account.	nvarchar(30)	No	PK	N/A	(Public) CASCADE ON DELETE & ON UPDATE
username	The account username	varchar(12)	No		N/A	(Public)
passwordHash	Hashed users password.	char(60)	No		N/A	(Public) (computed) Password must be hashed using bcrypt. BCryptHelper.HashPassword(mySalt,"password");
passwordSalt	Used to strengthen password.	char(60)	Yes		N/A	(Public) (computed) Random unique string mixed with password – then hashed. BCryptHelper.GenerateSalt(10)
firstName	Users first name.	nvarchar(30)	No		N/A	(Public) (constraint) No special character
lastName	Users last name.	nvarchar(30)	No		N/A	(Public) (constraint) No special character
fullName	Users' full name computed.	nvarchar(61)	No		N/A	(Public) (computed) firstname + “ ” + lastname
isStaff	The user is staff and not consumer	tinyint(3)	No		N/A	(Public) 1 represents staff account

						0 represents consumer account

possibleDiscounts

Column Name	Meaning	Data Type	Nulls	PK / FK	Default Value	Constraint /Column Property (including Data Classification)
discountID	Uniquely identifies the discount in the table via the unique id.	INTEGER	No	PK	N/A	(Public) (Computed) Set identity specification seed as 1 and increment as 1.
discountCode	Unique stings that can activate a user discount	CHAR(8)	No		N/A	(Public)
discountOffer	The actual discount such as “50” will be 50% off total booking price.	varchar(20)	No		“Expired”	(Public)

possibleDiscountShowingJunction

Column Name	Meaning	Data Type	Nulls	PK / FK	Default Value	Constraint /Column Property (including Data Classification)
discountID	Uniquely identifies the discount in the table via the unique id.	INTEGER	No	PK	N/A	(Public) (Computed) Set identity specification seed as 1 and increment as 1. ON UPDATE CASCADE ON DELETE CASCADE
showingID	Uniquely identifies the discount in the table via the unique id.	INTEGER	No	PK	N/A	(Public) ON UPDATE CASCADE ON DELETE CASCADE

10.4 Appendix 3: Meeting Forms

10.4.1 Meeting 1

FORM A

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING AGENDA
(To be completed ***before*** *the scheduled meeting)*

Student: B00355490	**Supervisor: Henry Hunter**
Meeting Number: 1	**Date/Time: 29/09/2021 9:00**

PROGRESS

Over the last month, the following tasks have been completed:

- Finding a supervisor.
- Finding a moderator.
- Deciding on a project.
- Creating a project specification list.
- Arranging a meeting.

The following tasks identified last month have not been completed or problems/issues have emerged that require attention:

AGENDA FOR FORMAL MEETING

1. Discussion of progress so far/since last meeting
2. Discussion of how specific tasks will be undertaken
3. Reading/investigative work undertaken so far and planned for the future
4. Discussion of problems or potential risks that have been identified (questions)
5. Discussion of work to be undertaken towards formal submissions (e.g. Interim Report)
6. Setting of tasks and planned targets before next formal meeting
7. Any other business?
8. Date of next formal meeting.

FOR NEXT MEETING:

i. GANTT Chart
ii. Start Interim Report

Questions:

I. Seek approval for camera QR scanning ethical approval?
II. Should a fictitious project specification be created from the client?

FORM B

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING MINUTES AND PLAN

*(To be completed **after** the scheduled meeting)*

Student: Connor Dillon **Supervisor: Henry Hunter**

Meeting Number:1 **Date/Time: 29/09/21 12:00**

MINUTES

The following tasks and issues were discussed, and specific actions agreed upon:

1. Seek approval for camera QR scanning – ethical approval (not needed).
2. Should a fictitious project specification be created by the client? No “pilot” project to avoid ethical issues.
3. Reviewed project specifications.

PLAN

The following tasks and timelines have been agreed, both for the next month and beyond:

For the next month:

- Project Specification Submitted
- GANTT Chart created
- Interim report Draft 1

- Find more literature to review

Beyond the next month

- Submit interim report

10.4.2 Meeting 2

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING AGENDA

*(To be completed **before** the scheduled meeting)*

Student: B00355490	**Supervisor: Henry Hunter**
Meeting Number: 2	**Date/Time: 27/10/2021 10:00**

PROGRESS

Over the last month, the following tasks have been completed:

- Creating a GANTT chart.
- Starting Interim report.
- Submission of Project Spec.
- Getting Project Spec approved.

The following tasks identified last month have not been completed or problems/issues have emerged that require attention:

AGENDA FOR FORMAL MEETING

1. Discussion of progress so far/since last meeting – Review GANTT CHART & Interim
2. Discussion of how specific tasks will be undertaken
3. Reading/investigative work undertaken so far and planned for the future
4. Discussion of problems or potential risks that have been identified (questions)
5. Discussion of work to be undertaken towards formal submissions (e.g. Interim Report)
6. Setting of tasks and planned targets before next formal meeting

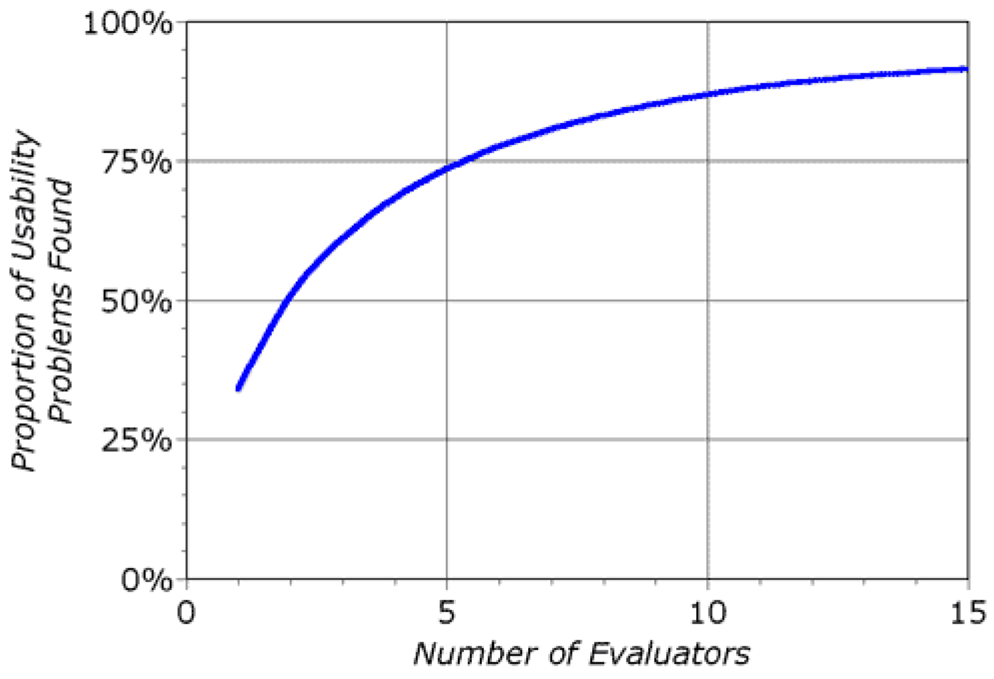
7. Any other business?
8. Date of next formal meeting.

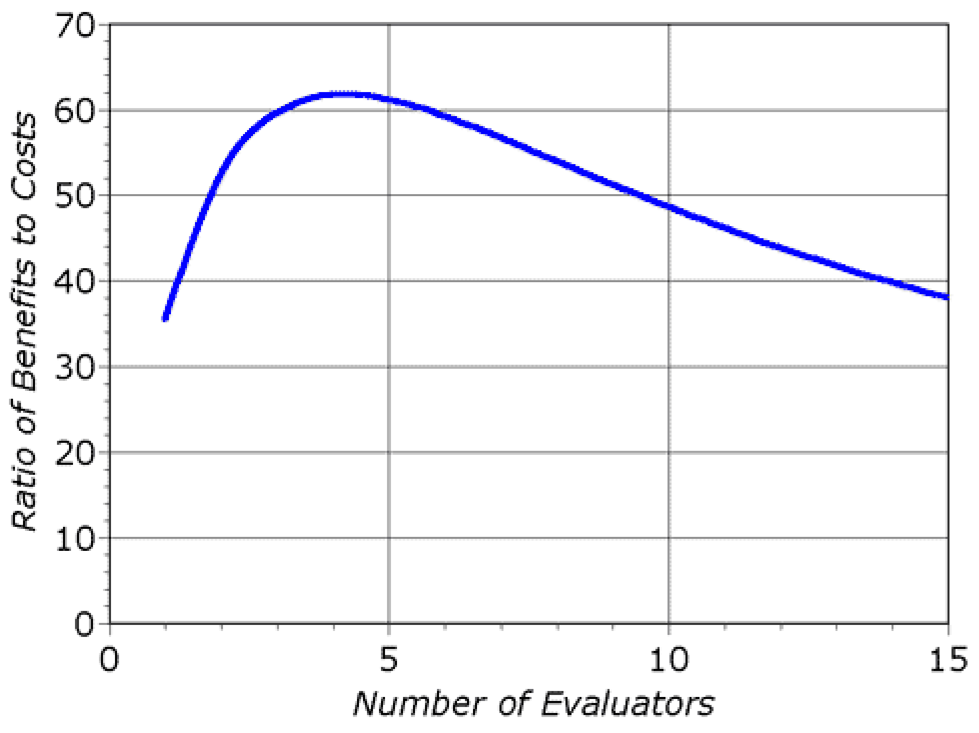
FOR NEXT MEETING:

i. First Draft of Interim Report
ii. GANTT chart draft

Questions:

Questions about Nielsen's Evaluation.





FORM B

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING MINUTES AND PLAN

*(To be completed **after** the scheduled meeting)*

Student: Connor Dillon **Supervisor: Henry Hunter**

Meeting Number:2 **Date/Time: 27/10/21 10:00**

MINUTES

The following tasks and issues were discussed, and specific actions agreed upon:

1. Gantt Chart – Add development phase
2. Reduce words, in order to stay within word count
3. Reviewed chapter headings.
4. Gantt Chart-Fix literature review dates + time
5. 5 UI testers, Nielson's no ethical approval needed.

PLAN

The following tasks and timelines have been agreed both for the next month and beyond:

For the next month:

- Send a copy of GANTT and Interim for review, at least one week before submission

Beyond the next month

- Start Development

10.4.3 Meeting 3

FORM A

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING AGENDA

*(To be completed **before** the scheduled meeting)*

Student: B00355490	**Supervisor: Henry Hunter**
Meeting Number: 3	**Date/Time: 23/11/2021 9:00**

PROGRESS

Over the last month, the following tasks have been completed:

- Interim Report
- GANTT Chart

The following tasks identified last month have not been completed or problems/issues have emerged that require attention:

AGENDA FOR FORMAL MEETING

1. Discussion of progress so far/since last meeting

2. Discussion of how specific tasks will be undertaken
3. Reading/investigative work undertaken so far and planned for the future
4. Discussion of problems or potential risks that have been identified (questions)
5. Discussion of work to be undertaken towards formal submissions (e.g. Interim Report)
6. Setting of tasks and planned targets before next formal meeting
7. Any other business?
8. Date of next formal meeting.

FOR NEXT MEETING:

- Start Application Development

Questions:

Can you please look at my Interim Report first draft?

FORM B

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING MINUTES AND PLAN

*(To be completed **after** the scheduled meeting)*

Student: Connor Dillon **Supervisor: Henry Hunter**

Meeting Number:3 **Date/Time: 23/11/21 10:00**

MINUTES

The following tasks and issues were discussed, and specific actions agreed:

1. Should the glossary of terms be at the beginning of the document?
2. Check all of your formatting! I see some fonts etc are different.
3. "published in 2021 Sharma, Ueno and Kingshott" would be better written, "published by Sharma etal, 2021".
4. Could put Nielsons ten useability heuristics as a heading
5. Check all paragraphs in the literature review and ensure you have at least one reference in each! e.g., last database paragraph!

6. Progress plan, put Gantt chart as an appendix and refer to it in the main report (not a list)!
7. Change Reflect heading to something like Self Reflection or Current project evaluation (something along those lines)
8. Maybe mention or reiterate in the conclusion what technologies you have decided to use (with references)
9. get someone to proofread it when you are finished as there are a few spelling/grammar mistakes to be fixed. You as the author may subconsciously skip over them.

PLAN

The following tasks and timelines have been agreed upon, both for the next month and beyond:

For the next month:

- Consider recommended changes and implement them where appropriate.
- Start database design and creation

Beyond the next month

- Start Development
- Start PowerPoint

10.4.4 Meeting 4

FORM A

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING AGENDA

*(To be completed **before** the scheduled meeting)*

Student: B00355490 **Supervisor: Henry Hunter**

Meeting Number: 4 **Date/Time: 20/01/2022 14:00**

PROGRESS

Over the last month, the following tasks have been completed:

- Database Diagram
- Created Database
- Received Interim feedback
- Started the Final Report
- Considered Ideas for the PowerPoint
- Application Connected to Database

The following tasks identified last month have not been completed or problems/issues have emerged that require attention:

AGENDA FOR FORMAL MEETING

1. Discussion of progress so far/since last meeting
2. Discussion of how specific tasks will be undertaken
3. Reading/investigative work undertaken so far and planned for the future
4. Discussion of problems or potential risks that have been identified (questions)
5. Discussion of work to be undertaken towards formal submissions
6. Setting of tasks and planned targets before next formal meeting
7. Any other business?
8. Date of next formal meeting.

FOR NEXT MEETING:

- Show application progress
- Show thesis progress

Questions:

Q. Thesis headings and subheadings:

Q. Entity-Relationship Model reviews (landscape in appendix, okay?)

Q. Use case review (UML)

Q. PowerPoint

prep 10%

style 20%

Content 50%

Questions 20%

10 slides

Proposed PP

SLIDE 1 (2 mins)
Intro

1. Cinema ticket booking system
    - Justification
        - Covid
        - Consumer benefits
        - Business Benefits

Slide 2 (2mins)
Aims

- Investigate the perceived experience customers have when using self-service systems.
- Create a simple and intuitive user interface.
- Select the most appropriate technologies for the application.
- Create a simple and robust database.
- Create a secure login system.
- Create an application that can interact and modify the database.
- Generate and integrate quick response (QR) codes.
- Integrate a third-party Application Programming Interface (API).
- A prototype which is suitable for the use case that it has been designed for.

Slide 3 (2mins)

Objectives

· Investigate the perceived experience customers have when using self-service systems through literature reviews.

· Develop a simple and intuitive user interface supported through iterations using Nielsen's Top Ten Heuristics.

· Identify, contrast, and compare possible technologies suitable for the application.

· Design, create and secure the applications database.

· Investigate and implement a secure login supported with existing literature.

· Create an application that can perform create, read, update, and delete (CRUD) functions on cinema showings and bookings.

· Research the most efficient and suitable way to integrate QR codes.

· Explore the possibilities of where the use of a third-party (API) may be suitable and enhance the final application.

· Adopt and integrate an agile development methodology.

· Develop a robust prototype through vigorous testing strategies.

Slide 4 (1mins)

Wireframes

supported with

Eye Tracking and Qualitative Page Evaluation. In a study conducted by Hernandez and Resnick using eye-tracking and page evaluation, their hypnosis was proved correct. The Guttenberg pattern was the most effective web page layout for the website landing page (2013).

Slide 5 (1 mins)

Wireframes Evaluated

using Nielsen’s top ten heuristics & heuristic evaluation (five evaluators) drop-off in benefits (graph)

Slide 6 (1 mins)

DBMS

- why SQLite (supported Literature (high level))
- Entity relationship model

Slide 7 (2mins)

why asp.net

ASP.NET being constantly maintained by Microsoft which is also open-source, we can conclude that the security in terms of security both frameworks are as secure as possible. Benefits drawn from the .NET environment include: “type-safety, multilanguage support, optimised memory management, just-in-time (JIT) compilation model that optimises the availability and performance of the application at runtime” (Hasan and Tu, 2003, p23)

Slide 8

small DEMO (4mins)

SQL creating database → showing asp.net connection → showing movies displayed on webpage

Slide 9 (5mins)

FORM B

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING MINUTES AND PLAN

(To be completed ***after*** *the scheduled meeting)*

Student: Connor Dillon **Supervisor: Henry Hunter**

Meeting Number:4 **Date/Time: 20/01/2022 14:0**

MINUTES

The following tasks and issues were discussed, and specific actions agreed:

- Thesis headings and subheadings
- PowerPoint structure
- Number all figures in thesis
- Work on demo for pp

PLAN

The following tasks and timelines have been agreed, both for the next month and beyond:

For the next month:

- Present PowerPoint
- Develop most of the application/prototype

Beyond the next month

- Complete thesis
- Proofread
- Test application

10.4.5 Meeting 5

FORM A

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING AGENDA

(To be completed ***before*** *the scheduled meeting)*

Student: B00355490	**Supervisor: Henry Hunter**
Meeting Number: 5	**Date/Time: 08/03/2022 11:00**

PROGRESS

Over the last month, the following tasks have been completed:

- CRUD added to prototype
- Thesis development stage first draft complete
- Python scanner created
- Application Connected to Database
- APIs Integrated

The following tasks identified last month have not been completed or problems/issues have emerged that require attention:

AGENDA FOR FORMAL MEETING

1. Discussion of PP feedback
2. Discuss testing table
3. Discussion of Essay Headings
4. Question at the end.

FOR NEXT MEETING:

- Completed Testing
- wireframes and ui
- testing complete

Questions:

1. Should I record a demo, one for each UML use case (Staff, Admin, Customer) (Create showing,Booking,QR scanner)

2. Are the screenshots in the development section appropriate, or do I need the full? page of code

3. Where should I reference the libraries that are needed in order for someone to operate the code in dotnet and python?

4. Example of the testing table.

A Passable Project will:

(i) A comprehensive literature and technical review.

(ii) Have a fully functional CRUD (create, read, update, and delete) ticket booking system.
    - Create Showing, Read Showings, Update accounts, Delete bookings

(iii) Research, design and implement a database (relational vs nonrelational)
    - Literature review EER model and implemented

(iv) Support customer and administrative authentication. (Login)
    - Levels and Cookie Claims Assigned

(v) The system will be controlled through an intuitive easy to use, intuitive user interface. Which will be iterated upon following Nielsen's Heuristic evaluation.
   - IN PROGRESS, WIREFRAMES AND NIELSENS

A First-Class Project will:

(i) Robust system because of testing.
   a. Testing Table, question below

(ii) Enhanced comprehensive and technical review.
   a. How to show an enhanced one extra heading?

(iii) Public API scraping OR private self-created API implemented on the webpage.
   a. QR CODE GEN, and openmovieDB

(iv) Ticket Creation (Generate and print QR code) QR code scanner.
   a. done python and API

(v) Research, compare and implement the most appropriate database protection measures with justification.
   a. Recommendations and password protection

need more comparison and justification

FORM B

COMPUTING HONOURS PROJECT (COMP10034)

PROGRESS AND FEEDBACK MEETING MINUTES AND PLAN

*(To be completed **after** the scheduled meeting)*

Student: Connor Dillon **Supervisor: Henry Hunter**

Meeting Number:5 **Date/Time: 08/03/2022 19:42**

MINUTES

The following tasks and issues were discussed, and specific actions agreed:

- Video UML is not needed
- PowerPoint feedback
- Code ScreenShots are fine the way they currently are
- Talked about Nelsons evaluation remotely as I have covid

- Agreed upon testing
- Ensure Library references are established in documentation

PLAN

The following tasks and timelines have been agreed, both for the next month and beyond:

For the 1/2 week away:

- Complete first draft
- Send Henry a copy of essay draft

Beyond the next month
NA

10.5 Appendix 4: Project Specification

COMPUTING HONOURS PROJECT SPECIFICATION FORM
(Electronic copy available on Moodle Computing Hons Project Site)

Project Title: Developing a Research Supported, Cinema Ticket Booking System

Student: Connor Dillon **Banner ID: B00355490**

Programme of Study: BSc (Hons) in Computer Science

Supervisor: Henry Hunter

Moderator: *Mark Stansfield*

Outline of Project:

The aim of this project will be to design and develop a cinema ticket booking system. This system will support Create, Read, Update and Delete functions (CRUD). This system should display different views to the user depending upon if they are staff or customers looking to book a ticket. Before any development takes place, all choices should be backed up with research explaining why the specific technologies were selected for this project. For the system to perform effectively, the database will be designed and documented before the *application moves into the development phase. The database will also be protected through a range of techniques most suitable for this case.*

Customers should be able to book their tickets and print off their tickets which will have a quick response code (QR) which can be scanned by a staff member of the cinema to verify that the ticket is legitimate (present in the database). Customers will have the ability to create, change, and remove bookings. Staff will be able to create cinema listings and edit customers bookings. Customer and staff permissions will be assigned via the implementation of a secure log in system. No payment gateway will be implemented in this application.

The results will hopefully show a fully functioning cinema ticket booking system, which is effective and efficient, and supported by research to show why the application was created the way it was.

A Passable Project will:

i. A comprehensive literature and technical review.

ii. Have a fully functional CRUD (create, read, update, and delete) ticket booking system.

iii. Research, design and implement a database (relational vs nonrelational)

iv. Support customer and administrative authentication. (Login)

v. The system will be controlled through an intuitive easy to use, intuitive user interface. Which will be iterated upon following Nielsen's Heuristic evaluation.

A First-Class Project will:

i. Robust system because of testing.

ii. Enhanced comprehensive and technical review.

iii. Public API scraping OR private self-created API implemented on the webpage.

iv. Ticket Creation (Generate and print QR code) QR code scanner.

v. Research, compare and implement the most appropriate database protection measures with justification.

Reading List:

Clarke, J., 2012. *SQL injection attacks and defense*. 1st ed. Waltham, MA: Elsevier, p.9.

Glez-Peña, D., Lourenço, A., López-Fernández, H., Reboiro-Jato, M. and Fdez-Riverola, F., 2013. Web scraping technologies in an API world. *Briefings in Bioinformatics*, [online] 15(5), pp.788-797. Available at: <https://academic.oup.com/bib/article/15/5/788/2422275 > [Accessed 2 October 2021].

Harrington, J., 2016. *Relational database design and implementation*. 1st ed. Amsterdam: Morgan Kaufmann/Elsevier, p.11.

Jatana, N., Puri, S., Ahuja, M., Kathuria, I. and Gosain, D., 2012. *A Survey and Comparison of Relational and Non-Relational Database*. [online] Citeseerx.ist.psu.edu. Available at: <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.678.9352&rep=rep1&type=pdf> [Accessed 25 September 2021].

Nielsen, J., 1994. *Heuristic Evaluation: How-To: Article by Jakob Nielsen*. [online] Nielsen Norman Group. Available at: <https://www.nngroup.com/articles/how-to-conduct-a-heuristic-evaluation/> [Accessed 2 October 2021].

Stone, D., 2005. *User interface design and evaluation*. 1st ed. Amsterdam: Morgan Kaufmann, pp.6-7.

Resources Required:

Hardware
Windows Machine
USB Camera (Scanning QR)

Software
Visual Studio
ASPNet core SDK
Python
SQL Management Studio
SQLite
Figma OR HTML + CSS

Marking Scheme:

Marking Scheme	Marks
Introduction	10
Literature Review	20

Requirements and Designs	15
Implementation and Evaluation	25
Testing	10
Conclusion	10
Critical Self-Appraisal	10

AGREED:

Student	**Supervisor**	**Moderator**
Name:	**Name:**	**Name:**
Connor Dillon	**Henry Hunter**	**Mark Stansfield**

IMPORTANT:

i.By agreeing to this form all parties are confirming that the proposed Hons Project will include the student undertaking practical work of some sort using computing technology / IT, most frequently achieved by the creation of an artefact as the focus for covering all or part of an implementation life-cycle.

i.By agreeing to this form all parties are confirming that any potential ethical issues have been considered and if human participants are involved in the proposed Hons Project then ethical approval will be sought through approved mechanisms of the School of CEPS Ethics Committee.