Annabel Kimber 864191

1. Development Models

a) In the scrum agile mythology the meetings are consistent keeping the team focuses, but also allows for changed to be made. Extreme Programming requires every stage of the system to run/compile before being added to the final project. Within waterfall the previous stage of the product life cycle needs to be completed before the next one can commence, meaning the team have clear milestones. An advantage of waterfall is that it is an easy and logical development model, and due to the cascading nature of it, each phase has to be signed off before another can start, meaning there is no overlap within it rigidity. However, because of this it is very difficult to go back and change previous mistakes.

(120 words)

b). There are many factors that go into choosing which software development model a company will choose, including the size of the company, the stability of the requirements, and the amount of money the company has. In smaller companies, clients have more freedom to change their mind because agile methods are more commonly used. This is because they are cheaper to operate because if a mistake is made it doesn't cost the company in both time and money. The client is involved the whole way the development rather than just in the beginning like with the waterfall method.

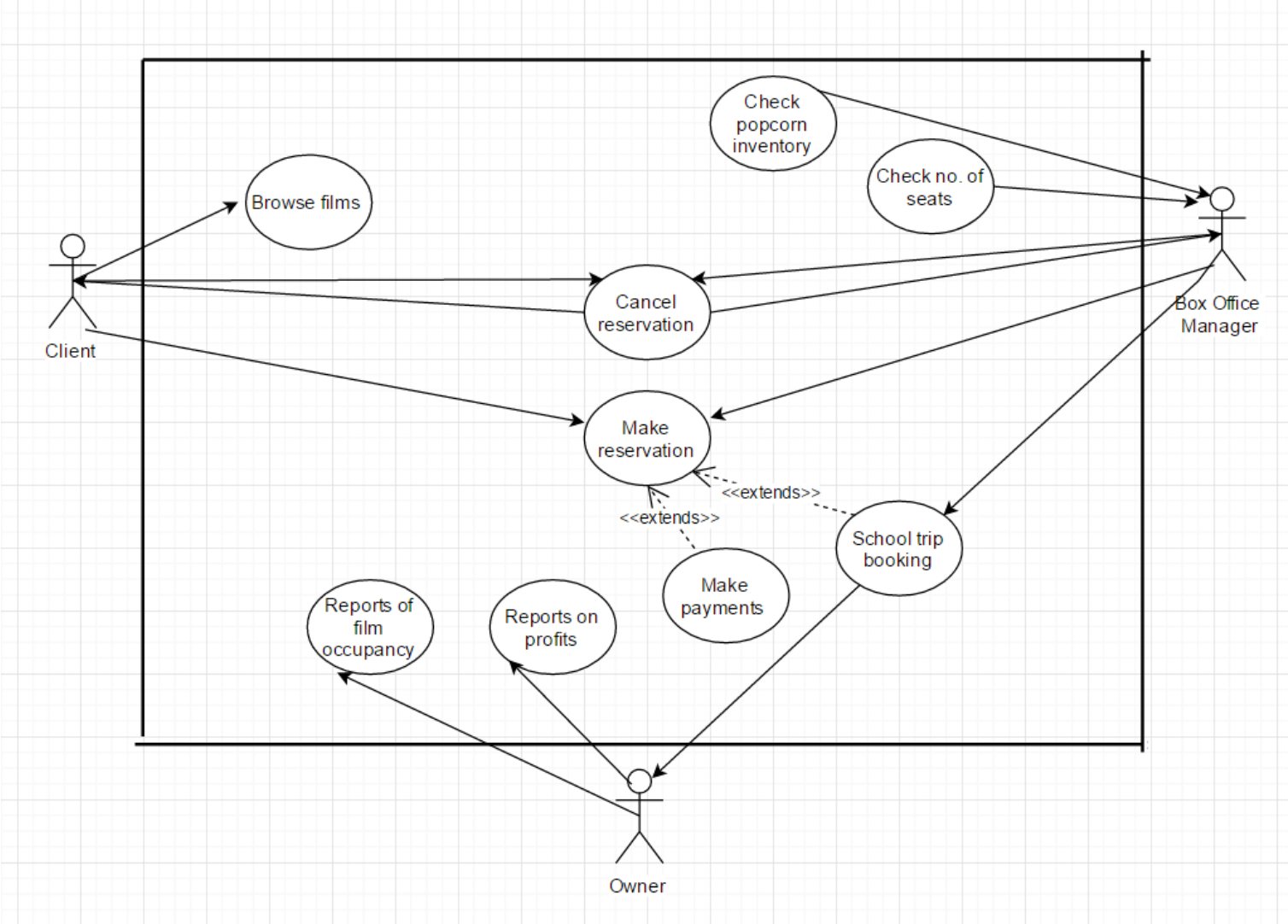
When working in smaller companies, like start-up companies, such as 'Enigma Recovery', because it is much easier to organise consistent meetings. With consistent meetings it is much easier to keep on top of the project. With agile methodologies, the client is at the centre, and the small regular meetings allow for the client to change their mind and changes to be implemented throughout the whole development process.

In larger companies, this isn't as easy as there tend to be a larger number of employees which make having regular meetings much harder, this is why larger companies tend to use the waterfall method. In car manufacturing companies such as 'Toyota', because an estimate can be given of a finishing date and final cost. Using the waterfall method patience has to be taken since there is no room for error as once a stage has been completed there is no going back. Being meticulous in prototyping and in earlier stages significantly reduces the cost of errors than in the final stages, and they are much easier to find when working through the cascading development model and signing of previous stages, rather than working on all stages simultaneously like in agile methods.

(296 words)

Annabel Kimber 864191

2. Use Case Diagram



3. Requirements

1. User and System Requirements

- User Requirement – 'Clients can browse films to see what is playing. At any time, they can also make reservations'. If the client can't see what films are playing, they then can't make bookings to see the films which loose the company. Browsing films and making bookings is why the client is using the system, so if doesn't work, it's not doing what the user expects it to do.
- System Requirement – 'Strings entered through text boxes in the interface should not break the system'. If entering text through text boxes on the interface does break the system, then the system is not functioning as expected.

2. Functional and Non- Functional Requirements

- Functional Requirement – 'They can also run statistics on the percentage occupancy of a particular film.' The system should allow the owners to produce reports on the occupancy so they can see how popular a film is so they can adjust the number of viewings accordingly.
- Non-Functional Requirements – 'The company's colours are yellow and blue.'. These are the colours that the company have specified, and having them on every page allows for continuity and professionalism.

(182 words)

Annabel Kimber 864191

4. Prototype

4.1 Scenario

Henry, the client was browsing films that were offered at theatre, he finds one that he likes and makes a booking for he and his wife. He enters his payment information. Henry then tells his wife that he has made the booking. She then informs him that they cannot attend the moving so he cancels the booking. Henry then creates a second viewing for a time that he and his wife can attend.

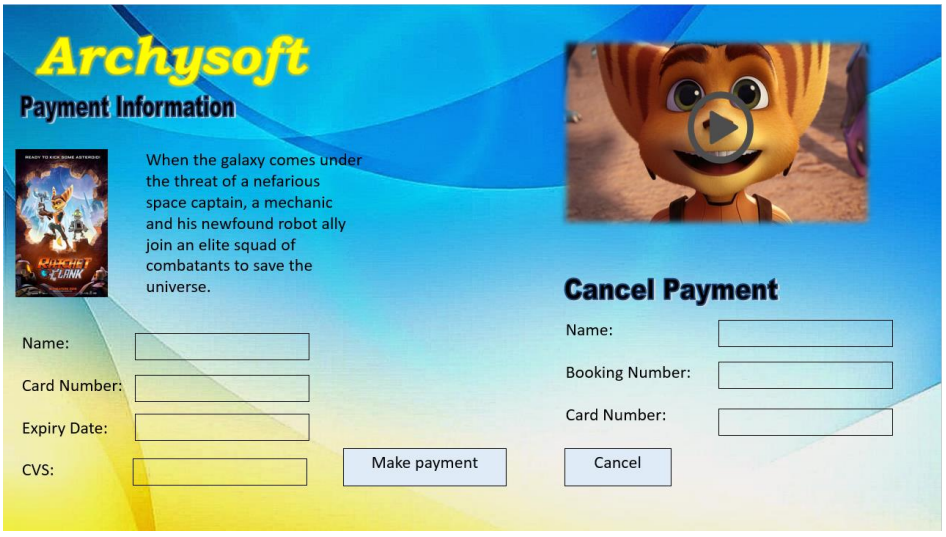
(73 words)

4.2 Design

GUI Design 1

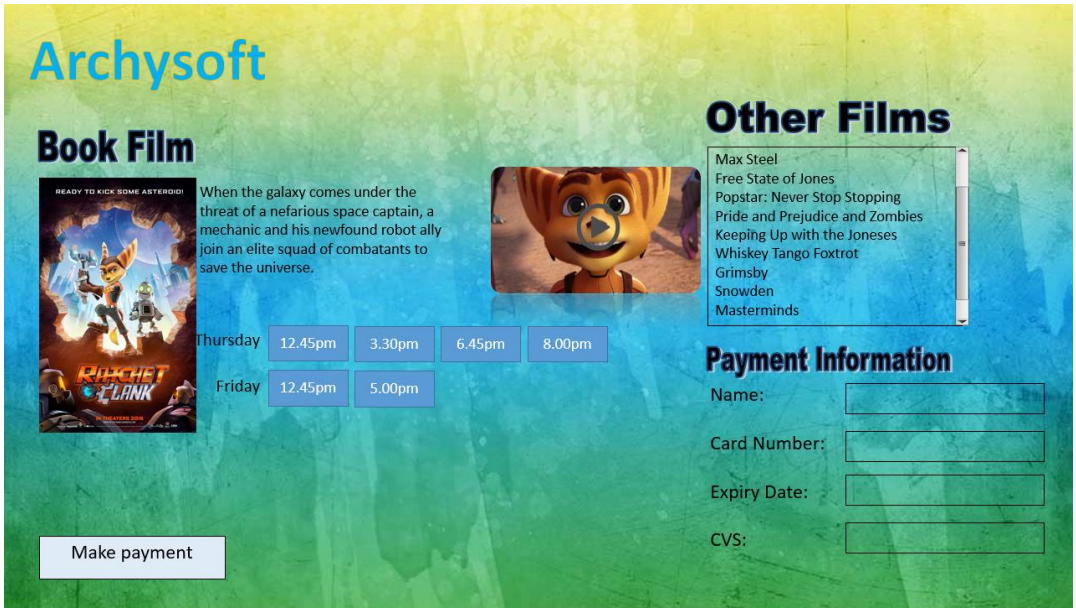
There is a scroll through the movies on offer allowing the user to browse. Once they have been selected, the user can find out more information about the film and add their payment information if they wish, watch the trailer or cancel their reservation.

(44 words)



Annabel Kimber 864191

GUI Design 2



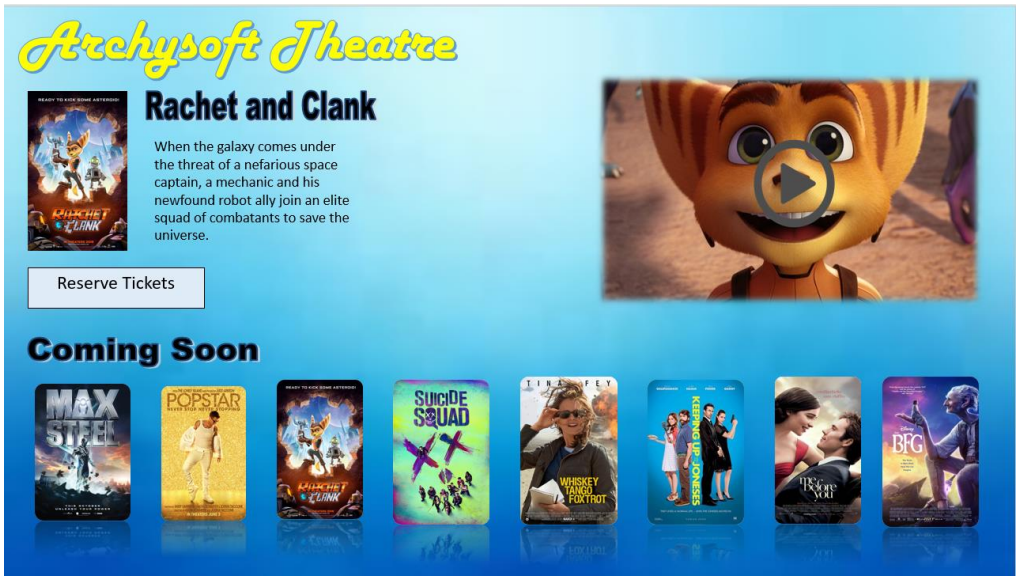
There is a list of films, which when one is selected, more information appears, showing bookings and prompting payment.

The user can also cancel their payment, where other films on offer are also shown.

(34 words)

Annabel Kimber 864191

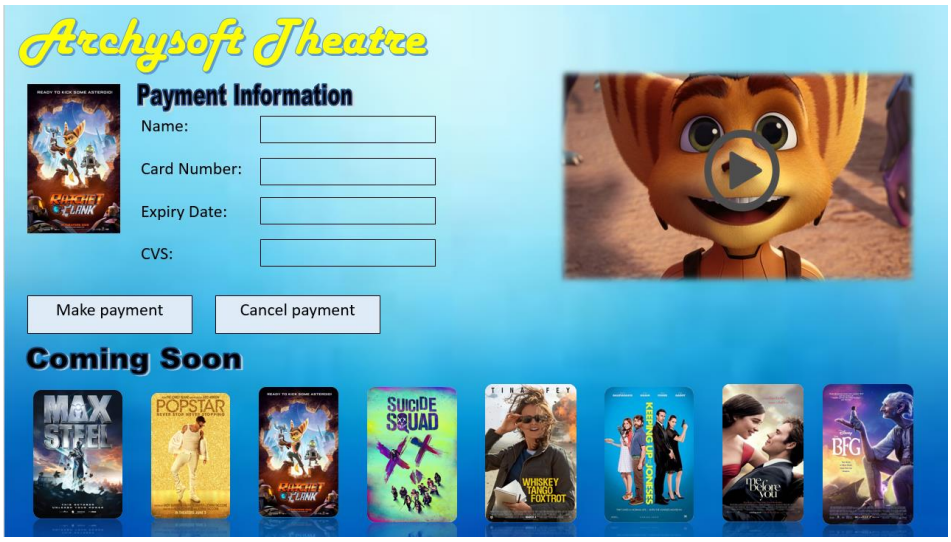
GUI Design 3



The user can browse films that are coming soon, watch the trailer and read information about the selected film.

A second window prompts the user to input their information, and then select whether they are making a payment or cancelling their booking.

(42 words)



Annabel Kimber 864191

GUI Design 2 UML:

GUIBrowse
- panelMain: JPanel - archysoftTextField: JTextFild - bookfilmTextField: JTextField - otherfilmsTextField: JTextField - paymentInformationTextField: JTextField - thursdayTextField: JTextFiled - fridayTextField: JTextFiled - 12.45pmButton: JButton - 3.30pmButton: JButton - 6.45pmButton: JButton - 8.00pmButton: JButton - nameTextField: JTextField - cardnumberTextField: JTextField - expirydateTextField: JTextField - cvsTextField: JTextField - image: Image - video: Video
+ makepayment(film: reservation: ArrayList)

GUIPayments
- archysoftTextField: JTextFild - cancelpaymentTextField: JTextField - otherfilmsTextField: JTextField -cancelButton: JButton - nameTextField: JTextField - bookingnumberTextField: JTextField - cardnumberTextField: JTextField -otherfilmsScroll: JScrollPane
+ cancel(film: reservation: ArrayList)

Annabel Kimber 864191

4.3 Prototype

Film
- title: String - director: String - duration: int
+ Film (in title:String, director:String, duration:int)

FilmSort
- filmSort: ArrayList<Film>
+addFilm (in Film)

Booking
- customer: Client - movie: Film
+ (in customer:Client, movie:Film)

BookingList
- bookings: ArrayList<Booking>
+ makeBooking(in Booking) + cancelBooking(in Booking)

Client
- name: String - surname: String - email: String - phoneNo: int - cardNo: int - expiryMonth: int - expiryYear: int - CSV: int
+ getName() + setName(name: String) + getSurname() + setSurname(surname:String) + getCardNo() + setCardNo(cardNo:int) + getExpiryMonth() + setExpirtyMonth(expiryMonth:int) + getExpiryYear() + setExpiryYear(expirtyYear:int) + getCSV() + setCSV(cSV:int)

Collaboration Diagrams