CSCI222 Assignment 3

@ease Neighbourhood Entertainment System

Wireless Media Distribution

Glenn Harper, Ian Mckay, Michael Boge, Tiaki Rice

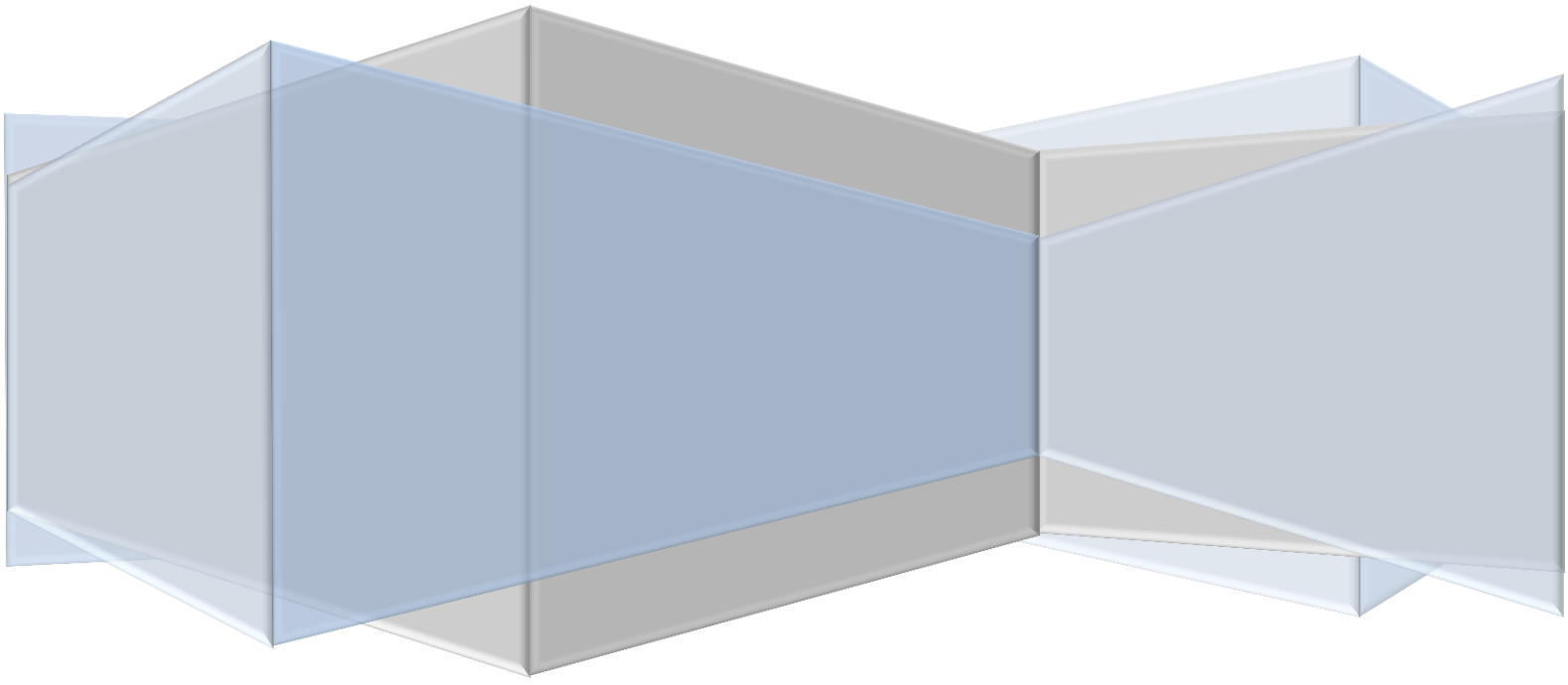


Table of Contents

Purpose 6
Scope 6
Introduction 7
Business Needs/Requirements 7
Users and User Needs 8
    Customers 8
    Vendor Staff 8
    Service Administrator 8
Product / Solution Overview 9
Assumptions and Dependencies 9
Major Features 10
Customer Interface 10
Staff Interface 11
Scope and Limitations 11
Cost and Pricing 12
Quality Ranges 12
Precedence and Priority 12
Other Needs 13
Network 13
Documentation Requirements 13
    Technical User Manual 13
    Online Help 13
    Installation and Configuration Guides and Read Me Files 13
Problem Statement 14
Position Statement 15
Overview 16
Market Demographics 16
Stakeholders 17
Users 17

Business (and User) Benefits 18
Functional Requirements 19
Customer Usage/Interface 19
Access/Storage 19
Security 20
Billing 20
Pricing 20
Rewards 21
Staff Usage/Interface 21
Non-Functional Requirements 22
Ease of Use 22
Platform 22
Interface (and Programming Language) 22
Storage 22
Customer Use Case Diagram 24
Service Administrator Use Case Diagram 24
Staff Use Case Diagram 25
Use Case Scenarios 26
Software Architecture 38
Performance 39
Security 39
Availability 39
Server Architecture 39
Master-slave architecture 39
Two-tier client-server architecture 39
Multi-tier client-server architecture 40
Peer-to-peer architecture 40
Technologies 40
Wireless network 40
Smart card technology 40
RAID disk arrays 40
Class Diagram 42

Login as Customer 45
Edit Account Settings 46
Browse Catalogue / Organise Catalogue 47
Purchase Media / Earn Reward Points / Use Reward Points 48
Create Playlist 49
Schedule Playback 50
Watch Media 51
Watch Preview 52
Login as Staff 53
Create Customer 53
Update Customer 54
Delete Customer 55
Create Recording 56
Update Recording 57
Delete Recording 58
Browse Billing 59
Mark Bill Paid 59
Apply Special Billing Provisions 60
Alter Category Pricing 60
Alter Individual Pricing 61
Start Service (for individual customer) 62
Stop Service (for individual customer) 63
Resume Service (for individual customer) 63
Start All Services 64
Stop All Services 64
Resume All Services 65
Alter Pricing Function 65
State Diagrams 66
Customer 66
Service Settings 66
Bill 67
Construction Iterations 74

Iteration 1 74
Iteration 2 75
Iteration 3 75
Iteration 4 75
Iteration 5 76
Iteration 6 76
Group Work 77
Meetings 77
Role Allocation 77
Version Control 78
Specification Coverage 79
Decisions 79
Architecture 79
Data Persistence (database) 79
Classes 80
Users 81
Use Cases 81
Interfaces 81
Appendix 1: Meeting Minutes 82
Appendix 2: Work Diary Samples 87
Tiaki's Work Diary 87
Ian Mckay's Work Diary 88

Glossary Definitions

@ease: The company @ease Inc

@ENES: The @ease Neighbourhood Entertainment System

Customers: The end-users of the system, users who use the system at home to access media.

Media: A piece of multimedia that can be from a number of types available typically available, e.g. movies, music, video games

Recording: Literally the a video/audio recording of the media stored in the system

Staff: Employees of the vendors of the system (both the @ease company and otherwise), they manage the system

Vendors: Companies and merchants that use existing methods of media distribution, who may want to switch to the @ease system.

Document Overview

Purpose

The purpose of this report is to document the implementation of the @ease Neighbourhood Entertainment System, specifically the inception and elaboration phases. This documentation displays these phases with the artifacts created in each section (the diagrams), as well as summaries describing the planning of the system and outlining how it will be developed/implemented.

The intended audience of this document are the stakeholders involved with the system, namely the media companies (i.e. “video stores”) that will be likely to use this system and the @ease company itself developing the system (as a record of the process). The customers are also partly the intended audience, as they will also be using the system as end-users.

Scope

The system to be produced is the @ease Neighbourhood Entertainment System (referred to as @ease throughout the document).

@ease is a service that is intended to replace the typical 'video store' in terms of its role in providing media to a neighbourhood. It allows customers to browse and select media to view, without the need to physically travel to a vendor ('video store'), the media instead being delivered to their home via a network connection. The system also allows the vendor staff to manage the media available to customers (both in terms of the actual media, and things like pricing), as well as the actual customers and their access to the service.

Vision

Introduction

@ENES is a service that is to perform the role of a neighbourhood video store by allowing the browsing and delivering of media to customers, as well as the managing of this content by vendor staff. It will perform these tasks over a simple browser interface using a stand-alone wireless network.

Business Needs/Requirements

Currently customers wanting to view/use certain media need to travel to a 'video store' in order to browse and borrow it. The staff working in in these media related businesses (such as video stores) need to manage this physical media, both in organising, pricing and returning.

In order to simplify this process a computer system is to be developed, which will allow for the easy management and borrowing of this media through a network. This system must specifically be able to:

- Store all this information, and have it available for quick and easy access
- Effectively use a network to transfer the information/media
- Allow customers to browse the media catalogue (including previews), and sort this information
- Allow customers to purchase the viewing of the media and schedule it
- Provide reward points that can be used in the system to customers
- Provide security by confirming a customer's identity using a smart card
- Allow customers to set billing information (such as period and method)
- Allow staff to change and set the pricing for the media, taking into account things like formulas and service tariffs, as well as allowing it to be set individually or en masse
- Allow staff to create, edit and delete customers
- Allow staff to view customer billing information
- Allow staff to create, edit and delete media records
- Allow staff to start, stop or resume tasks

All staff and user tasks must be performed using a internal interface, which will be built in Java, and can run in web-browsers.

Users and User Needs

There are 3 main users of the @ENES service:

- **Customers**: the main users, and end users, of the service. Their role is to use for browsing through and getting the media delivered to them at home. They are expected to have basic computer knowledge (sufficient for using a web-browser)
- **Vendor Staff (such as video store staff):** the staff who use the service to display their media and manage some features of it (such as pricing, content, etc). They are expected to have sufficient computer knowledge and access to the service.
- **Service Administrator:** a staff member who manages the actual @ENES service. The main role they play is altering the functions used to pricing.

These users all have the following specific needs:

Customers

- Able to browse through a media catalogue
- Can search/sort the catalogue (such as get a certain genre, or sort by name)
- Able to view previews (some of which may need to be purchased) through the catalogue
- Can create a 'playlist' of frequently viewed media
- Able to schedule playback of media or a playlist for a later time
- Able to manage their account, mainly to set billing settings (such as period)
- Able to obtain, through purchases, and use 'reward points'

Vendor Staff

- Able to manage services, starting, stopping and resuming them
- Able to browse user billing information
- Can create, edit and delete customers from the system
- Can create, edit and delete media records from the system
- Able to alter pricing tariffs

Service Administrator

- Able to edit the formula used to calculate pricing

Product / Solution Overview

@ENES is a service which provides customers with access to large amounts of media (of many different types), which will be stored in databases and distributed (i.e. streamed) to the customer via a built-in network connection. The customer will be able to browse the media before selecting what they want to view/stream, via a Java web-based catalogue viewable in their browser. This catalogue will also support searching and sorting to allow easier browsing.

The customer is also able to change their billing options, set up playlists and schedule playback times via this web-based interface, though each is done in its respective view of the interface.

The system also caters for the vendors, specifically allowing the staff to manage the media, customers and services via a Java interface (separate from the customer's interface). The staff are able to add, delete and edit both media recordings and customers, as well as access billing information and start, stop and resume services using this interface. Each of these tasks are performed in a different section/view relative to that action.

The system also provides staff and service administrators with the ability to modify the pricing of media (both in batches or individually).

These interfaces, both the customer and staff ones, are accessed by first logging into the system. This is performed via a smart-card.

Assumptions and Dependencies

The following assumptions and dependencies relate to the capabilities of the @ENES service:

- It is assumed that @ease and the vendor companies run compatible database and server architectures.
- The current content of vendor companies (i.e. the media and their customers) can be integrated into the new system.
- It is assumed the media to be put in the system is capable of streaming (otherwise the playing will not work).

Major Features

The structure of this service can be broken down into a few parts:

- **The Database:** The back-end of the service, providing the information used by the system, namely the customer/staff records and the media recordings.
- **The Web-based Customer Interface**: The interface the customer uses to browse and access media. It is accessed through a web-browser and is Java based.
- **The Staff Interface**: A Java application the staff and service administrators use to manage the system.

The interfaces outlined above can be further described by their different views used to perform the different actions available.

Customer Interface

The following views are contained within the customer interface, and can be easily navigated between using tabs at the top of each view.

- **Login View:** This is where the customer is prompted to enter their smart-card, so they can be checked for validity and then logged into the system.
- **Catalogue View:** This is where the customer browses the media and selects records to view previews for and purchase. The media records are displayed as a list in this view, displaying key information (such as title, author/director, genre, etc) and an image. This list can be scrolled through, or sorted via clicking on the list headings. The customer can also search here using a search bar at the top of the page. If the customer clicks on a record they will be shown full information on that film, as well as being given the option of viewing a preview for that selection, and an option to purchase it.
- **Customer Account Settings View:** This is where the customer can select their billing options (such as when they are billed) through inputs (both text-based and drop-down lists). They can also view their account information from this view.
- **Playlist View:** This is where the customer can see their favourite media, as well as media they currently have access to (have purchased) and arrange it into playlists. They can also schedule the playback of an item or a playlist for a certain time in this view. The playlists are shown again in a list, similar to the form they are in the catalogue view.

Staff Interface

The following views are contained within the staff interface, and can be easily navigated between using tabs at the top of each view.

- **Login View:** This is where the staff member is prompted to enter their smart-card, so they can be checked for validity and then logged into the system.
- **All Services View:** This is a simple view where the staff member can start, stop and resume the services to all customers (via a button menu).
- **Customer Service View:** This is where the staff member can start, stop and resume services to an individual customer. The customers are displayed in a list with buttons reflecting the available actions positioned next to each customer.
- **Billing View:** Here the staff can see the billing options set for each customer via a spreadsheet like tabular view.
- **Media Management View**: This view allows a staff member to add, remove and edit media records. The view provides a list of all media records (to allow them to be edited and removed via buttons), as well as input fields for adding a media record.
- **Customer Management View:** Similar to the prior view, except it is based on adding, removing and editing customer records instead. It again provides a list of customer records (with buttons for editing and removing), as well as input fields for adding the record.
- **Pricing Management View:** The staff can use this view to change the pricing for media in this view. It provides lists for selecting firstly what method of price setting is to be used (either by selecting a group, then modifying all records of that group, or selecting an individual media record). The price is then changed by using a slider.
- **Pricing Function View:** This view is not accessible to staff, but only by system administrators. They can change the function used for calculating prices in this view.

For images of these view designs please see the User Interface section of this report

Scope and Limitations

This product will cover all of the requirements mentioned above (and in the requirements section of this report). The initial product will focus on providing a method for the customer to access media from home, with the service (and its media) being managed by staff.

In terms of limitations;

- The media related data will be limited by what the databases can contain, meaning some media types may not be able to be implemented within the system
- The system may not be able to display all data on the media (such as all the actors), as this would over-complicate the media database.

Cost and Pricing

The customers will have to pay to use the system, this will most likely be in the form of paying for a piece of media (the money going to @ease or the relevant vendor), this pricing will be determined by the pricing formula put into place by the vendor's system administrator. The customer may also be expected to pay a small fee to keep subscription in the service.

The vendors will have to pay a few at set intervals (monthly, weekly or yearly) to @ease in order to have a licence to use the system.

In addition to the development costs of the system, for @ease, or the cost of the licence, for vendors, there will be the additional cost for the hardware necessary to run the service (i.e. the databases, servers, RAID-array drives, etc).

Quality Ranges

The following define the quality ranges for the @ease system in terms of performance, usability and other factors:

- **Availability**: the system will be available 24 hours a day, 7 days a week, with the possible exception of small amounts of downtime for management and maintenance.
- **Usability:** the system shall be easy to use for both the customers who are expected to have only small amounts of technical knowledge and for the staff who are expected to have suitable knowledge.
- **Maintenance/Management:** the system and its content (such as the pricing formula) is able to be changed and used without the need to reconstruct the system.
- **Accessibility:** the system shall provide help, both in the forms of online help for the customer, and more technical help in the internal interface for the staff.

Precedence and Priority

The full functionality discussed in this vision should be available in the first actual release of the system (as the vendors and other customers are to be provided full functionality without the need to considerably update soon after release).

For detailed descriptions of the priority of the functionalities and the order they are to be implemented in construction please see the 'Requirements' section and the 'Construction Iterations' section respectively.

Other Needs

- **Accessibility:** @ENES is to be as compatible as possible, as such it needs to be able to be used on as many browsers as possible (probably IE6+, Firefox, Safari and Chrome).
- **Performance:** The customer must be able to access the media records as rates suitable for browsing through them.
- **Back-end:** The vendors must have suitable back-end systems (databases and servers) in order to run the service, so the data can be stored and accessed by customers.

Network

In order for the clients to be able to access the service they must be able to receive reasonable access to the wireless network it runs on (with reasonable access being a connection that allows media to be streamed at a reasonable rate through it).

Documentation Requirements

The following documents are required for the full system release:

Technical User Manual

Describes the use of the internal interface for managing the system (such as adding/removing/editing records and starting/stopping/resuming services). The descriptions will be from the Staff and System Administrator's viewpoints.

The manual is to be provided both as a hard-copy with the system, but more importantly as a soft copy document linked to via the internal interface help.

Online Help

Each view/action of the online customer interface should provide a help option. This help option will briefly describe the parts of the view and the steps required to perform the actions that view provides. This help shall be from the viewpoint of the customer.

Installation and Configuration Guides and Read Me Files

The system shall come with a number of files describing the basic installation and configuration process. However, it is assumed a professional will be present for the installation of the system for vendors, so this documentation is only needed for re-installing/re-configuring.

The customer only needs a brief guide explaining how to set up the smart-card reader needed to use the system.

Problem and Position Statements

Problem Statement

The problem of	browsing and distributing movies, as well as organising the movies in the current 'video store' methods
Affects	the customers who wish to browse and get the media, as they have to go out to 'video stores' to get the media, and the management staff of these stores, who as they have to account for the distribution of their stock (such as ensuring it is returned);
the impact of which is	the customers have to go all the way out to 'video stores' to get the media (and return it) and they may not have all the information they want on it. Which results in dissatisfied customers who are less likely to make this trip. For the staff it means much of their time must be taken up managing this.
A successful solution would be	a system which allows customers to access the media (for both browsing through and actual watching) from home, through some form of network. Meaning they would no longer have to travel and would be more likely to use the system, and the staff would no longer have to manager physical stock.

Position Statement

For	customers and current media vendors (such as video stores)
Who	need a method to easily access(customer) or deliver(company) media content.
@ENES is a	system consisting of a network and web-based applications
That	stores and transmits/streams media to allow customers to securely browse this media through a catalogue as well as view it, and allows staff to manage the media (such as adding stock and pricing)
Unlike	the alternative of storing physical copies of the media and making customers go to a store to browse and borrow it (taking it home), as is done in currently in video stores.
our product	provides access to a web-based catalogue which allows customers to select movies (by searching, showing previews, etc) and stream them to watch at home using a wireless network, without the need to leave the house.

Business Context

Please note that much of this sections content is discussed in the Vision section above, please read it before this section.

Overview

@ease wishes to implement new service called the @ease Neighbourhood Entertainment System. This service (and the devices it includes) will be sold by @ease and other vendors and will perform the perform the role of a neighbourhood video store by allowing customers to browsing and watch media via streaming over a stand-alone wireless network connection. This media and customer information will be stored on a database-server back-end which is managed by the vendor staff through their own specific interface.

This service will bring @ease to the fore-front of home entertainment systems thus improving their image, attracting more customers and vendors. These vendors and customers will in turn receive the benefits of using the new system.

Market Demographics

The target market for this system is both the vendors who wish to sell and manage the product, but most importantly the customers who wish to purchase/view media from home without needing to travel to a vendor.

Stakeholders

Name	Represents	Role
@ease Inc	The company (and team) developing @ENES, as well as the head managers of the service	To develop the system and provide support to the vendors, as well as managing the system themselves
Vendor (and Staff)	Company and staff who pay to use and sell the @ENES package. They are responsible for managing the content and services of the @ENES service	Add, remove, edit customer and media records. Start, stop and resume services, set media pricings
Customers	End users who pay to access the content of the @ENES system.	Browse media, purchase media, view previews, view media, create playlists, schedule playback of media, edit account settings (mainly billing)

Users

Name	Description	Stakeholder
Customer	Browses the online catalogue for media, viewing previews as the do this, purchases media and views it via streaming. Also creates playlists of this media and schedules the playback times.	Self-represented
Staff	Adds content (namely customer and media records) to the system, manages the customers services by starting, stopping or resuming them, also sets the prices of media in the system.	Represented by @ease Inc or Vendor (as is a member of staff at one of these)
Service Administrator	Manages the pricing function used to calculate appropriate media prices given certain service tariffs and criteria (such as media runtime, time of day, etc).	Represented by @ease Inc or Vendor (as is a member of staff at one of these)

Business (and User) Benefits

The system will provide a number of benefits to the stakeholders and users.

Benefit	Supporting Features
@ease Inc will appear as a better company in terms of home entertainment systems, bringing in more interested vendors and customers.	The system as a whole
Vendor can manage their whole media stock and attract customers not interested in the old system	The online customer interface and the media management in the internal interface
Staff can manage media without the need to physically manage it as well (such as ensuring it is returned and not overdue)	Media recordings are stored in database servers which have an internal interface to easily add/remove/edit media and customer records
Service administrators can change the way all media is priced without needing the reconstruct the pricing functionality.	Interface where a new pricing formula can be entered
Customers can browse and receive media content at home without the need to go to a vendor (with more information available due to the ability to view previews)	Online browsable catalogue and stream-able media of system

Overall, the system will benefit the stakeholders and users by providing a new kind of service which will attract paying customers (for the vendors and @ease) and allow easier media browsing, management and delivery for the end users (customers and staff).

Requirements

Functional Requirements

Customer Usage/Interface

Requirement #: FuncReq_1
Description: The system should provide a catalogue of media/content for the customer to browse through, in a UI. The elements of the catalogue must be able to be organised/sorted using common groupings (such as title, genre, etc)
Rank of Importance: High

Requirement #: FuncReq_2
Description: The system must allow the customer to create a 'playlist' of media/content they frequently view
Rank of Importance: Medium

Requirement #: FuncReq_3
Description: Allows the customer to view a streaming preview of the media (if available), taking into account any costs that may be involved.
Rank of Importance: High

Requirement #: FuncReq_4
Description: The customer must be able to schedule playback of media (or playlists) at times they specify.
Rank of Importance: Medium

Access/Storage

Requirement #: FuncReq_5
Description: Customer, media, staff and other information is to be securely and logically stored (with the ability to allow the logical location of the media recordings to be changed). The system must provide easy and efficient access to this data through a network.
Rank of Importance: High

Requirement #: FuncReq_6
Description: The system is to securely encapsulate the media/content streamed to the customer to prevent unauthorised capture/copying of the content (be this by external unauthorised users, or inappropriate use by the customers).
Rank of Importance: High

Security

Requirement #: FuncReq_7
Description: The system is to store information enabling it to identify customers/staff and provide appropriate access, using a 1024 bit DES key certificate, provided by a smart card.
Rank of Importance: High

Billing

Requirement #: FuncReq_8
Description: The system must allow the customer to select the billing period used for the billing purposes related to the system. The periods will be defined from the anniversary of when the customer subscribed to @ease
Rank of Importance: High

Requirement #: FuncReq_9
Description: The system should provide the ability to set special billing provisions for a customer or request.
Rank of Importance: Medium

Pricing

Requirement #: FuncReq_10
Description: The system's pricing methods must be based on Service Tariffs
Rank of Importance: High

Requirement #: FuncReq_11
Description: The pricing can be modified or set either en-masse (by selecting all new releases, or media of a certain genre, etc) or individually. Pricing performed individually will override the en-masse general pricing, if set.
Rank of Importance: High

Requirement #: FuncReq_12
Description: Pricing must be calculated and set using a function or data rate, time of day, duration of the recording, special rates and loyalty program status. This function is to be alterable by the system administrator.
Rank of Importance: High

Rewards

Requirement #: FuncReq_13
Description: The system must provide a loyalty program, where customers who frequently use the system are able to earn reward points. These points should be redeemable for access to the system or other merchandise.
Rank of Importance: Low

Requirement #: FuncReq_14
Description: The loyalty program must also support partnerships, where points earned in these partnership services can be transferred and redeemed by the @ease service.
Rank of Importance: Low

Staff Usage/Interface

Requirement #: FuncReq_15
Description: The staff must be able to add, edit and delete customers from the service, using the interface.
Rank of Importance: High

Requirement #: FuncReq_16
Description: The staff must be able to add, edit and delete media recordings from the service, using the interface.
Rank of Importance: High

Requirement #: FuncReq_17
Description: The stuff must be able to start, stop and resume services via this interface. This must be both as a whole and for the service for individual customers.
Rank of Importance: High

Requirement #: FuncReq_18
Description: The system must allow staff to browse customers' billing details.
Rank of Importance: Medium

Non-Functional Requirements

Ease of Use

ID	NonFuncRec_1
Description	The service should be simple for customers to use, not requiring any special training (other than how to use a web-browser).
Priority	High

Platform

ID	NonFuncRec_2
Description	The service is to be usable on any major operating system (Windows, Mac, Linux), and the client side browsable in any modern browser (IE, Firefox, Chrome, Safari).
Priority	High

Interface (and Programming Language)

ID	NonFuncRec_3
Description	The system provides a customer interface. It is to be a web-based GUI interface accessible through a browser. The interface will be written in Java
Priority	High

ID	NonFuncRec_4
Description	The system provides an internal interface for staff use is to be written in Java.
Priority	High

Storage

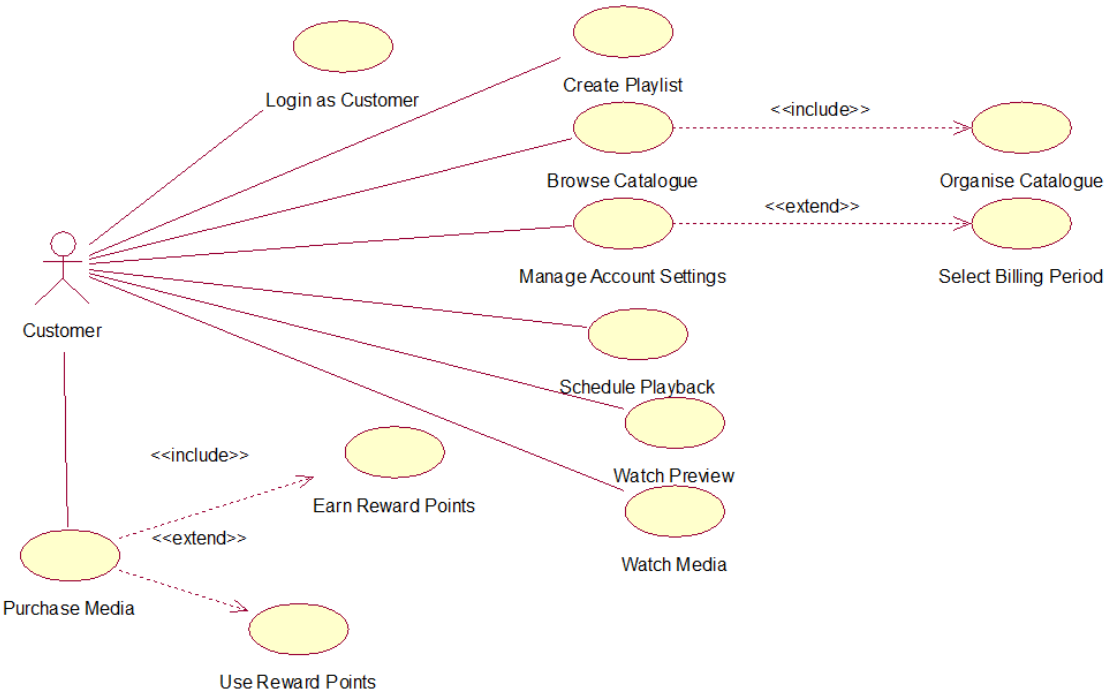
ID	NonFuncRec_5
Description	The media and content recordings are to be stored in farms of RAID arrays.
Priority	High

ID	NonFuncRec_6
Description	There can be an unlimited number of items in the catalogue (limited only by storage space)
Priority	Medium

Use Cases

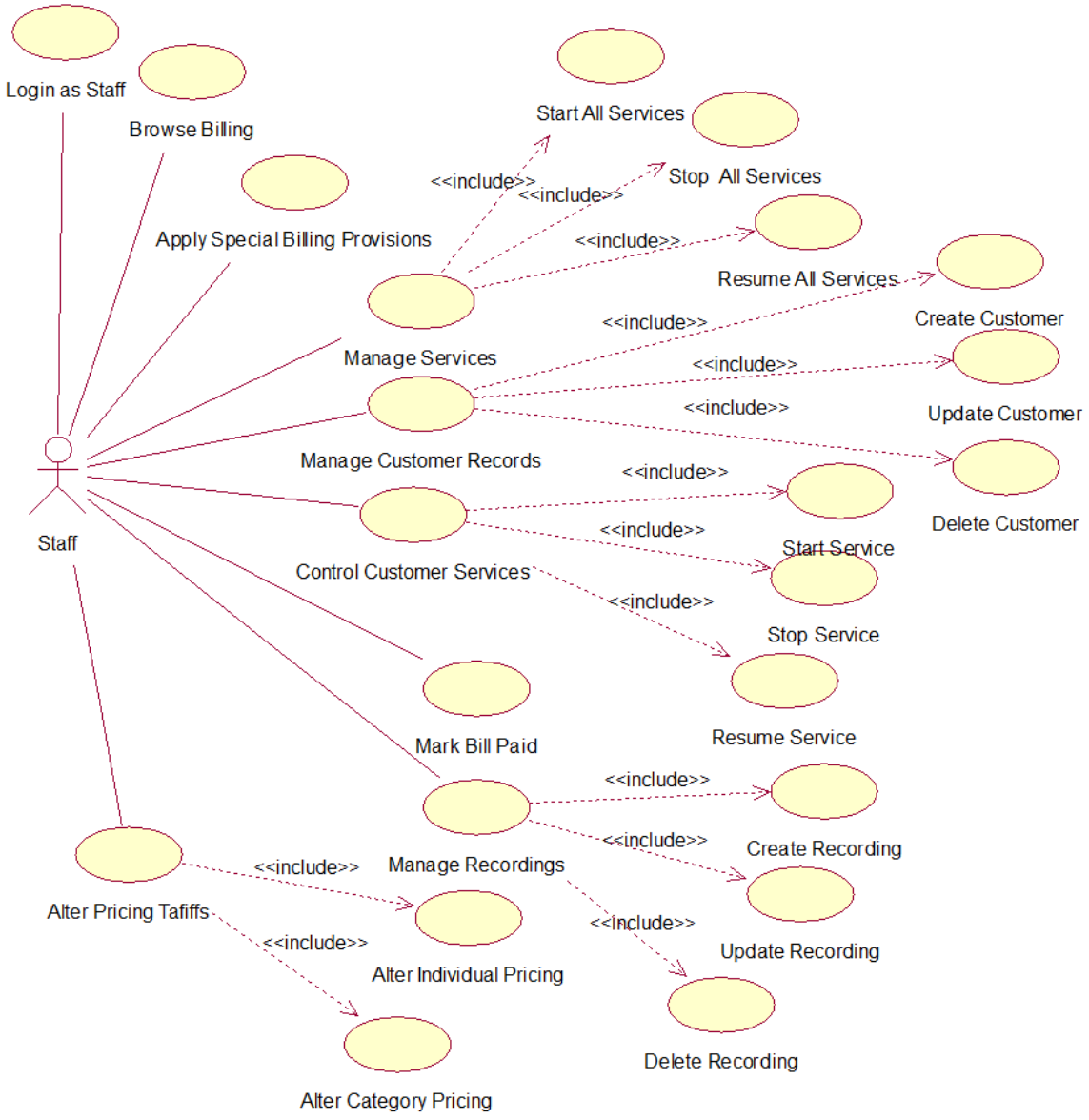
After considering the list of requirements that we produced from the description of the system we settled on three actors; Staff, Service Administrator and Customer, and quite a large number of use cases between them.

Customer Use Case Diagram



Service Administrator Use Case Diagram

Staff Use Case Diagram



Use Case Scenarios

The following are the scenarios we defined for the above use cases:

Use Case Name	Login as Customer
Primary Actor	Customer
Summary	To allow the customer to login to the system
Pre-Conditions	The customer has a valid account The customer enters the correct username and password
Normal Flow of Events	1. Customer opens the login interface 2. Customer enters username and password 3. Customer selects “Login” option
Extensions	N/A
Post-Conditions	Customer is successfully logged in

Use Case Name	Create Playlist
Primary Actor	Customer
Summary	To allow the customer to create playlists of media which they access frequently
Pre-Conditions	The customer has media in their library
Normal Flow of Events	1. Customer selects “Create playlist” option 2. Customer adds media one at a time to a list 3. Customer names the list 4. Customer saves the list
Extensions	N/A
Post-Conditions	A playlist is saved and is able to be loaded at a later date

Use Case Name	Browse Catalogue
Primary Actor	Customer
Summary	The Customer is able to view a list of available media content for purchase and is able to organise the catalogue
Pre-Conditions	N/A
Normal Flow of Events	1. Customer selects browse catalogue option 2. Customer is able to browse and filter all available content
Includes	Organise Catalogue
Post-Conditions	N/A

Use Case Name	Organise Catalogue
Primary Actor	Customer
Summary	Customer is able to sort the catalogue by various parameters
Pre-Conditions	Customer is browsing the catalogue
Normal Flow of Events	1. Customer selects a specific filter
Extensions	N/A
Post-Conditions	Content is filtered appropriately

Use Case Name	Manage Account Settings
Primary Actor	Customer
Summary	Customer is able to tweak their settings according to preference
Pre-Conditions	N/A
Normal Flow of Events	1. Customer selects “Manage Account Settings” from menu 2. Customer changes settings accordingly 3. Customer selects save button
Extensions	Select billing period
Post-Conditions	Customers settings are altered and saved

Use Case Name	Select Billing Period
Primary Actor	Customer
Summary	Customer is able to select the period on which they are billed.
Pre-Conditions	Customer is currently in account settings
Normal Flow of Events	1. Customer selects billing period 2. Customer selects save button
Extensions	N/A
Post-Conditions	Customers billing period is altered

Use Case Name	Schedule Playback
Primary Actor	Customer
Summary	Customer is able to schedule playback of media at times specified by the customer without limit
Pre-Conditions	Customer has purchased media
Normal Flow of Events	1. Customer selects “Schedule Playback” option from the menu 2. Customer selects media to be played 3. Customer selects time at which this media will be played 4. Customer selects a “Schedule” button
Extensions	Modify Playback
Post-Conditions	Media will be played at the time specified by the user

Use Case Name	Watch Preview
Primary Actor	Customer
Summary	Customer is able to watch a preview of specific media
Pre-Conditions	Customer must have purchased the preview if required
Normal Flow of Events	1. Customer selects preview for playback from their library 2. Preview is played in the customers window 3. Customer exits from the preview
Extensions	N/A
Post-Conditions	N/A

Use Case Name	Watch Media
Primary Actor	Customer
Summary	Customer is able to watch purchased media in the browser
Pre-Conditions	Customer must own at least a single piece of media
Normal Flow of Events	1. Customer selects “Watch Media” from the menu 2. Customer selects media to be viewed
Extensions	N/A
Post-Conditions	N/A

Use Case Name	Purchase Media
Primary Actor	Customer
Summary	Customer is able to purchase media using previously set billing methods
Pre-Conditions	Customer has a piece of media they wish to purchase
Normal Flow of Events	1. Customer selects “Purchase Media” from the menu 2. Customer enters media details and presses confirm 3. Customer accepts presented confirm screen
Extensions	Use Reward Points
Includes	Earn Reward Points
Post-Conditions	Media is added to the customer’s account Customers account is charged for purchase

Use Case Name	Use Reward Points
Primary Actor	Customer
Summary	Customer is able to use accumulated points to purchase a variety of rewards
Pre-Conditions	Customer must have reward points
Normal Flow of Events	1. Customer selects “Use Reward Points” from the menu 2. Customer selects rewards to claim 3. Customer confirms claim
Extensions	N/A
Post-Conditions	Reward points are removed from the users account equal to points spent

Use Case Name	Earn Reward Points
Primary Actor	Customer
Summary	Customer earns reward points when media is purchased
Pre-Conditions	Customer must purchase media
Normal Flow of Events	1. Customer purchases media
Extensions	N/A
Post-Conditions	Reward points are added to the users account

Use Case Name	Login as Staff
Primary Actor	Staff
Summary	To allow the staff member to login to the system
Pre-Conditions	The staff member has a valid account The staff member enters the correct username and password
Normal Flow of Events	1. Staff member opens the login interface 2. Staff member enters username and password 3. Staff member selects “Login” option
Extensions	N/A
Post-Conditions	Staff member is successfully logged in

Use Case Name	Manage Customer Records
Primary Actor	Staff
Summary	Staff Members are able to manage fields in customers records
Pre-Conditions	N/A
Normal Flow of Events	1. Staff member selects “Manage Customer Records” from the menu 2. Staff member selects appropriate option
Extensions	Create Customer Update Customer Delete Customer
Post-Conditions	N/A

Use Case Name	Create Customer
Primary Actor	Staff
Summary	Staff members are able to create a new customer record
Pre-Conditions	Staff member has appropriate details to put into the new customer record Staff member is currently in the “Manage Customer Records” menu
Normal Flow of Events	1. Staff member selects “Create Customer” from the menu 2. Staff member inputs data into appropriate fields of the data form 3. Staff member selects “Create New Customer” button at the bottom of the data form
Extensions	N/A
Post-Conditions	A new customer record is created and added to the database

Use Case Name	Update Customer
Primary Actor	Staff
Summary	Staff members are able to modify customers details
Pre-Conditions	Customer record to modify must first exist in the database Staff member is currently in the “Manage Customer Records” menu
Normal Flow of Events	1. Staff member selects “Update Customer” from the menu 2. Staff member modifies appropriate fields in the data form 3. Staff member selects the “Save Customer” button at the bottom of the data form
Extensions	N/A
Post-Conditions	Customer record in the database is updated

Use Case Name	Delete Customer
Primary Actor	Staff
Summary	Staff members are able to remove a user record from the database
Pre-Conditions	Customer record to modify must first exist in the database Staff member is currently in the “Manage Customer Records” menu
Normal Flow of Events	1. Staff member selects “Delete Customer” from the menu 2. Staff member enters required customer details to be deleted 3. A confirmation message is shown and “Confirm” is selected by the staff member
Extensions	N/A
Post-Conditions	The account is removed from the database

Use Case Name	Browse Billing
Primary Actor	Staff
Summary	Staff members are able to browse customers billing details
Pre-Conditions	N/A
Normal Flow of Events	1. Staff member selects “Browse Billing” from the menu 2. Staff member enters an account name or ID to browse the billing of 3. A “Confirm” option is selected
Extensions	N/A
Post-Conditions	Billing details for the specified customer is displayed on screen

Use Case Name	Control Customer Services
Primary Actor	Staff
Summary	Staff members are able to start, stop and resume services of a specific customer
Pre-Conditions	N/A
Normal Flow of Events	1. Staff member selects “Control Customer Services from the menu
Extensions	Start All Services Stop All Services Resume All Services
Post-Conditions	N/A

Use Case Name	Start All Services
Primary Actor	Staff
Summary	Allows a staff member to start all of a customer’s services
Pre-Conditions	Staff member must be in the “Control Customer Services” menu
Normal Flow of Events	1. Staff member selects “Start All Services” from the menu
Extensions	N/A
Post-Conditions	All of the customers services are started

Use Case Name	Stop All Services
Primary Actor	Staff
Summary	Allows a staff member to stop all of a customer’s services
Pre-Conditions	Staff member must be in the “Control Customer Services” menu
Normal Flow of Events	2. Staff member selects “Stop All Services” from the menu
Extensions	N/A
Post-Conditions	All of the customers services are stopped

Use Case Name	Resume All Services
Primary Actor	Staff
Summary	Allows a staff member to resume all of a customer’s services
Pre-Conditions	Staff member must be in the “Control Customer Services” menu
Normal Flow of Events	3. Staff member selects “Resume All Services” from the menu
Extensions	N/A
Post-Conditions	All of the customers services are resumed

Use Case Name	Manage Recordings
Primary Actor	Staff
Summary	Allows a staff member to create, update or delete a recording from the system
Pre-Conditions	N/A
Normal Flow of Events	1. Staff member selects “Manage Recordings” from the menu
Extensions	Create Recording Update Recording Delete Recording
Post-Conditions	N/A

Use Case Name	Create Recording
Primary Actor	Staff
Summary	Allows a staff member to add a recording to the database
Pre-Conditions	Staff member must have all the required files and data for the recording Staff member must be in the “Manage Recording” menu
Normal Flow of Events	1. Staff member selects “Create Recording” from the menu 2. Staff member fills out the required fields in the data form 3. Staff member selects the “Create Recording” option
Extensions	N/A
Post-Conditions	A new recording is added to the database

Use Case Name	Update Recording
Primary Actor	Staff
Summary	Allows a staff member to update a recording currently in the database
Pre-Conditions	The recording to modify must exist in the database Staff member must be in the “Manage Recording” menu
Normal Flow of Events	1. Staff member selects “Update Recording” from the menu 2. Staff member updates the required fields in the data form 3. Staff member selects the “Update Recording” option
Extensions	N/A
Post-Conditions	An existing recording in the database is modified

Use Case Name	Delete Recording
Primary Actor	Staff
Summary	Allows a staff member to delete a recording currently in the database
Pre-Conditions	The recording to delete must exist in the database Staff member must be in the “Manage Recording” menu
Normal Flow of Events	1. Staff member selects “Delete Recording” from the menu 2. Staff member enters the required data into the data form 3. Staff member selects the “Delete Recording” option 4. The staff member selects the “confirm” option
Extensions	N/A
Post-Conditions	An existing recording in the database is deleted

Use Case Name	Alter Pricing Tariffs
Primary Actor	Staff
Summary	Staff members are able to alter tariffs
Pre-Conditions	N/A
Normal Flow of Events	1. Staff member selects “Alter Pricing Tariffs” from the menu 2. Staff member fills out the appropriate data form 3. Staff member confirms the change
Inclusions	Alter Individual Pricing Alter Category Pricing
Post-Conditions	The prices for the affected items are changed.
Use Case Name	Alter Individual Pricing
Primary Actor	Staff
Summary	Staff members can individually re-cost items which overrides the default price
Pre-Conditions	Item exists Staff member is in the “Alter Pricing Tariffs” menu
Normal Flow of Events	1. Staff member selects “Alter Individual Pricing” from the menu 2. Staff member enters appropriate item details and new price 3. Staff member selects the confirm change option
Extensions	N/A
Post-Conditions	The items price is changed in the database

Use Case Name	Alter Category Pricing
Primary Actor	Staff
Summary	Staff members can set the default price for specific categories
Pre-Conditions	Category must exist Staff member is in the “Alter Pricing Tariffs” menu
Normal Flow of Events	1. Staff member selects “Alter Category Pricing” from the menu 2. Staff member enters appropriate category details and new price 3. Staff member selects the confirm change option
Extensions	N/A
Post-Conditions	Cost of all items in the category specified are modified

Use Case Name	Manage Services
Primary Actor	Staff
Summary	Staff able to start, stop and resume services
Pre-Conditions	N/A
Normal Flow of Events	1. Staff selects “Manage Services” from menu
Extensions	Start Services Stop Services Resume Services
Post-Conditions	N/A

Use Case Name	Start Services
Primary Actor	Staff
Summary	Allows the Staff to start all services
Pre-Conditions	Staff must be in the “Control Customer Services” menu
Normal Flow of Events	1. Staff selects “Start Services” from the menu
Extensions	N/A
Post-Conditions	All services are started

Use Case Name	Stop Services
Primary Actor	Staff
Summary	Allows the Staff to stop all services
Pre-Conditions	Staff must be in the “Control Customer Services” menu
Normal Flow of Events	1. Staff selects “Stop Services” from the menu
Extensions	N/A
Post-Conditions	All services are stopped

Use Case Name	Alter Pricing Function
Primary Actor	Service Administrator
Summary	The service administrator is able to change the formulae used to calculate the pricing of services
Pre-Conditions	N/A
Normal Flow of Events	1. The admin selects “Alter Pricing Function” option from the menu 2. The admin enters the new formulae in the data form 3. The admin selects the “Alter Pricing Function” option
Extensions	N/A
Post-Conditions	The pricing function for all services is changed and all relevant services are updated

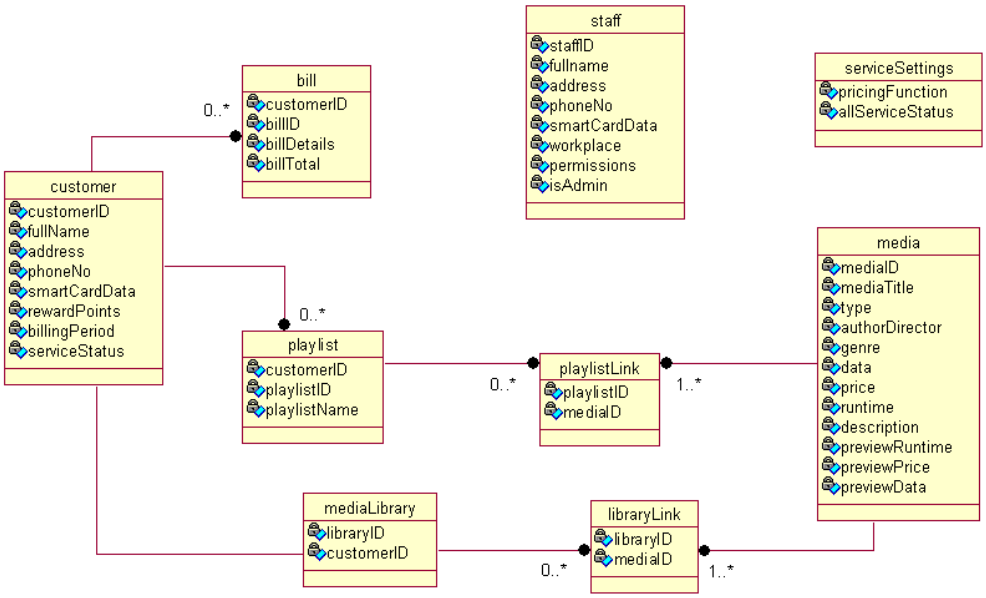
Use Case Name	Resume Services
Primary Actor	Staff
Summary	Allows the Staff to resume all services
Pre-Conditions	Staff must be in the “Control Customer Services” menu
Normal Flow of Events	1. Staff selects “Resume All Services” from the menu
Extensions	N/A
Post-Conditions	All services are resumed

Risks

The following table describes the possible risks which could affect the system. The table is ordered in order of the 'Risk Ranking/Magnitude', meaning more serious risks (either because they are more likely to occur or will have serious consequences) are at the top.

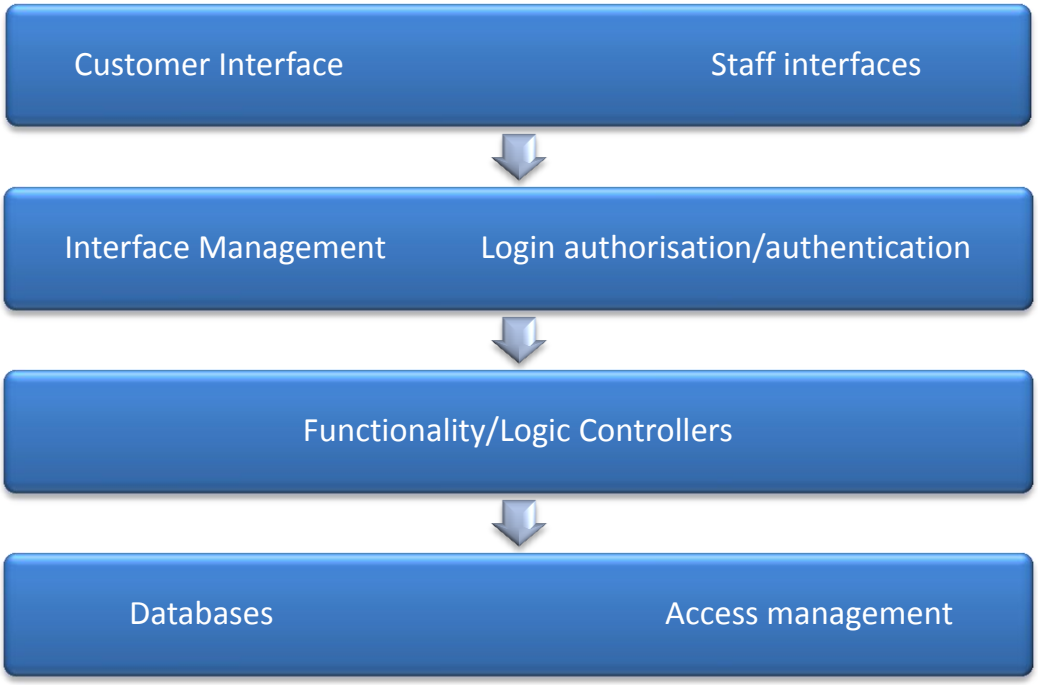
Risk Ranking/Magnitude	Risk Description and Impact	Solution Strategy and/or Contingency Plan
7	May be incompatible with certain browsers/operating systems or their configurations.	Addressed by using Java, which is cross-platform in terms of operating systems. To be further addressed in Elaboration phase, ensure it meets as many browser standards as possible.
6	Large numbers of customers using system at peak times may cause performance issues (slow performance).	Assess/test heavy user load (both with practical tests and predictions) in early phases, paying attention to response time.
6	Large, high quality media may be transferred slowly, again causing performance issues.	Assess/test large media files in early phases, paying attention to response/buffering time.
4	Requires specific knowledge to run the media databases, which video store staff would most likely have been untrained in.	Provide ample training for these staff, with an easy to understand interface for the staff to use. Addressed in the UI design in the Elaboration phase.
3	Customers may have little technical knowledge and be unable to convert to the new system.	Provide simple browser interface, with plenty of help functionality. Addressed in the UI design in the Elaboration phase.
3	Storing billing information online could be insecure, resulting in fear of loss of details	Ensure smart card system is effective and the system (and this information) cannot be accessed or changed by other methods.
3	Possibility of other parties gaining access to customer's smart cards. Meaning they can access their information.	Company support telephone number.

Data Persistence Diagram



Architecture

Software Architecture



The system is designed using a 'Layered Pattern' architecture. This pattern is used as it best fits the situation, with the system being organised into a set of layers, namely:

- The customer and staff interfaces
- The management of the customer and staff interfaces and the login authorisation/authentication
- The controllers providing the main functionality and logic of the program
- The databases and access management

This pattern is used as it follows the flow of the data in the system.

1. Firstly the databases will be accessed to retrieve the necessary information (be it a media recording or customer information, etc), using the access management sections.
2. This data will then be passed up through the main controllers, which will ensure it is valid and perform any necessary arrangements (such as updating prices).
3. The data will then be passed to the interface management layer where it will be changed into the correct format for display.

Performance

The performance of the @ENES system will largely depend on the most processor dependant operations and the architecture of the system. The largest performance issue for the @ENES system will be the ability to seamlessly stream media. A critical decision will be the addition of a streaming media buffer, and how large the buffer may be. This will create a trade-off between the time it takes to start media, and the quality of the media. Media playback operations should be done on the client-side to eliminate unnecessary network throughput.

Security

Security will be an essential part of the @ENES system. Personal data as well as the use of customer rewards points will be essential intellectual property to secure. Since accounts will have the ability to charge customers for media consumption, it is essential to validate users before gaining access to the system. Media distribution must be controlled and localized to the @ENES system only. The ability to access media to download for personal use without correct accounting can have effects on the relevant media industries. Media licensing laws should also be closely followed.

Availability

The availability of the @ENES system will be crucial for customer satisfaction. Should media servers fail without an appropriate mechanism for recovery; the customer will not be able to use the system. Redundant distribution servers will create a recovery mechanism to allow media to be available to customers even if one server fails.
This pattern is used because it gives the following advantages to the system:

- It provides a clear separation between sections, meaning each section can be considered easy to build and test, focusing on one layer each iteration
- Allows replacement of entire layers so long as the interface is maintained.
- Redundant facilities (e.g., authentication) can be provided in each layer to increase the dependability of the system.

Server Architecture

Master-slave architecture

The use of master-slave architecture with any system would have both advantages and disadvantages. Advantages include the ease of control of services and configuration with a centralised server. A major disadvantage of this would be the dependence of the system on the slave processes that control the data for the master. The @ENES system does not have a need for any externally controlled processes and therefore would render the Master-slave architecture inappropriate.

Two-tier client-server architecture

Two-tier client-server architecture allows the system to become distributed to an indefinite number of clients, depending upon the performance of the central server. A thin-client approach would direct all media distribution, application logic and processor intensive work to the central server, leaving the thin-client with the bare minimum of hardware required. A fat-client approach eases

some of the workload on the server as some processor intensive work would be allocated to the client end. For implementation in the @ENES system, a thin-client approach would be better from a design point of view.

Multi-tier client-server architecture

A multi-tier client-server approach would involve the client connecting to a web front-end which derives it's content from a database server. This has advantages for deployment of systems and media distribution. It protects the database server from invalid input as the web front-end would be able to validate the data before passing it to the database-server. This also allows better systems management as services can be manipulated from the web front-end whilst maintenance may be performed on the database server. These factors made a multi-tier client-server approach the most appropriate architecture for the @ENES system.

Peer-to-peer architecture

A peer-to-peer architecture would not be appropriate for the @ENES system. Peer-to-peer systems require computations to be done at any node of the network which would amount to each client hardware to be of high-end hardware. Storage and security of the system would be threatened as a decentralised content database would be unreliable and insecure.

Technologies

Wireless network

System will utilize localized wireless networks to transfer media content quickly and securely. The system will not be presented over existing internet service providers in order to minimize costs.

Smart card technology

Authentication to the @ENES system will use smart card technology to authenticate users into the system. A 1024-bit DES key will provide sufficient security for users of the system.

RAID disk arrays

Redundant arrays of independent disks are used to store details about staff, customers and the media itself. This ensures appropriate data recovery should a disk fail. It may also add speed to disk transfers and hence the system overall.

Deployment

The initial system is to be implemented using the pilot deployment model. There are a number of reasons why this deployment model is appropriate:

- It allows the current system (i.e. regular video stores) to continue being used (without needing to either stop them as a whole, or phase in different parts for all regions)
- It is suitable for the 'neighbourhood' style of the system, namely the service will be local to a particular region, so one region (i.e. the pilot) can be updated to the new service and evaluated before other regions are.

Another key feature in the deployment is the nature of the service's users/stakeholders, particularly the vendors, as they have to completely change their methods to use the service. This means that they will only start to use the system if they believe it will provide potential benefits to them, which could be shown by the pilot model. Also, because they will be the main means of use of the system and giving customers access to the service, without them first converting there is no possible way to do a complete change over (or in fact even just phasing in one part at a time).

Thus, the pilot deployment model is appropriate for the initial implementations.

In terms of the deployment of the physical parts of the system, it will follow the following structure:

For the vendors (and for the @ease company providing the service)

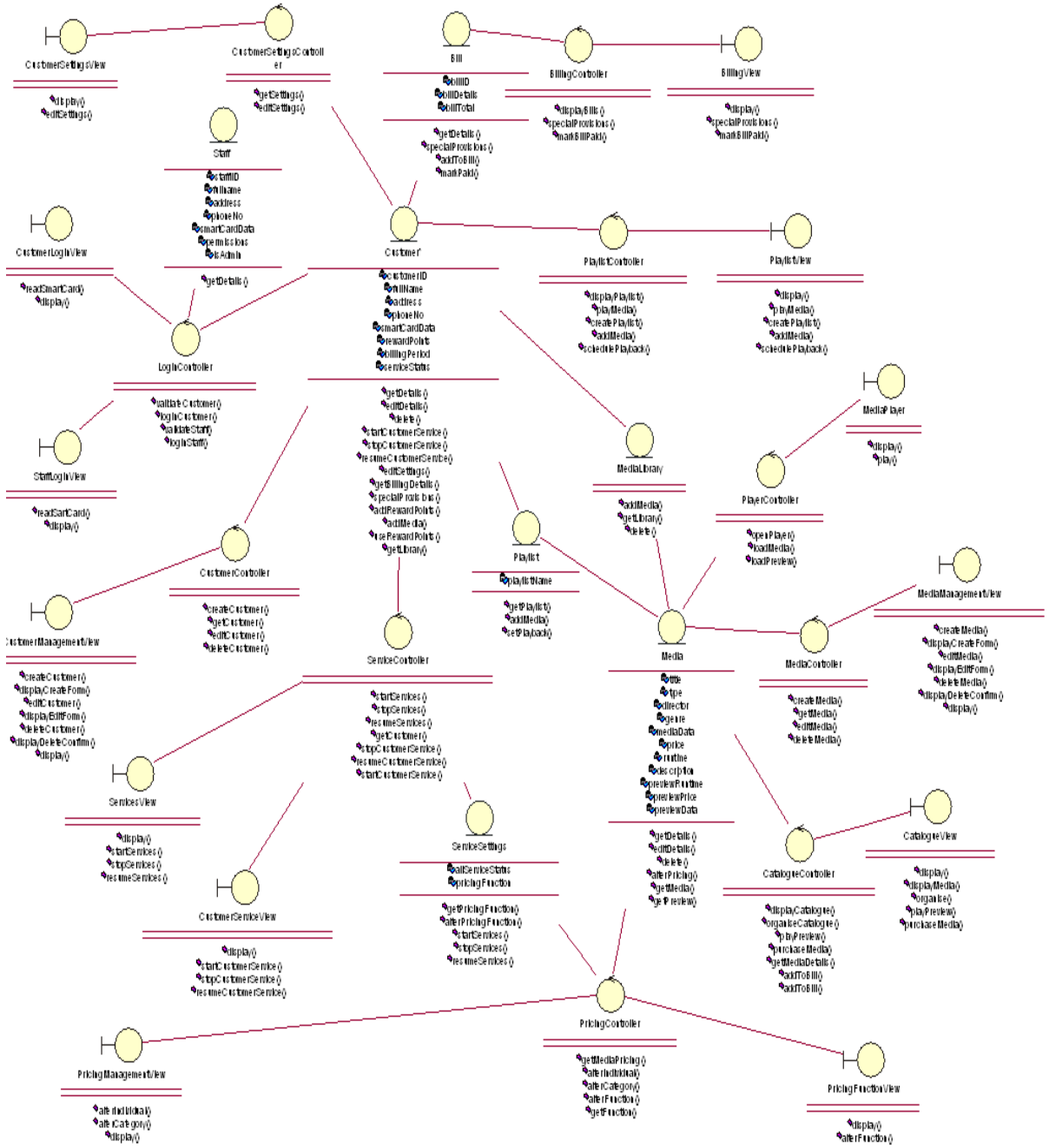
- Deploy the server(s) and database(s) needed to run the system (i.e. physically set up the server and create the necessary databases and database tables to store the required data, and run the system)
- Implement the internal vendor/staff interface for accessing the databases/servers (i.e. install and configure the Java application on the internal server the staff will use)
- Deploy the equipment necessary for providing the wireless network customers use to access the service (i.e. configure the server to use a network connection, possibly one that is physically deployed itself in this phase)
- Implement the customer interface for externally using the system (i.e. install the customer interface software on the web-service to provide the customers access to the system of the previously set up network)
- Add data (media and new customer) to finish the deployment

The customers themselves require no special deployment to use the system, simply requiring a connection to the @ease network, a browser to access the interface and a smart card reader.

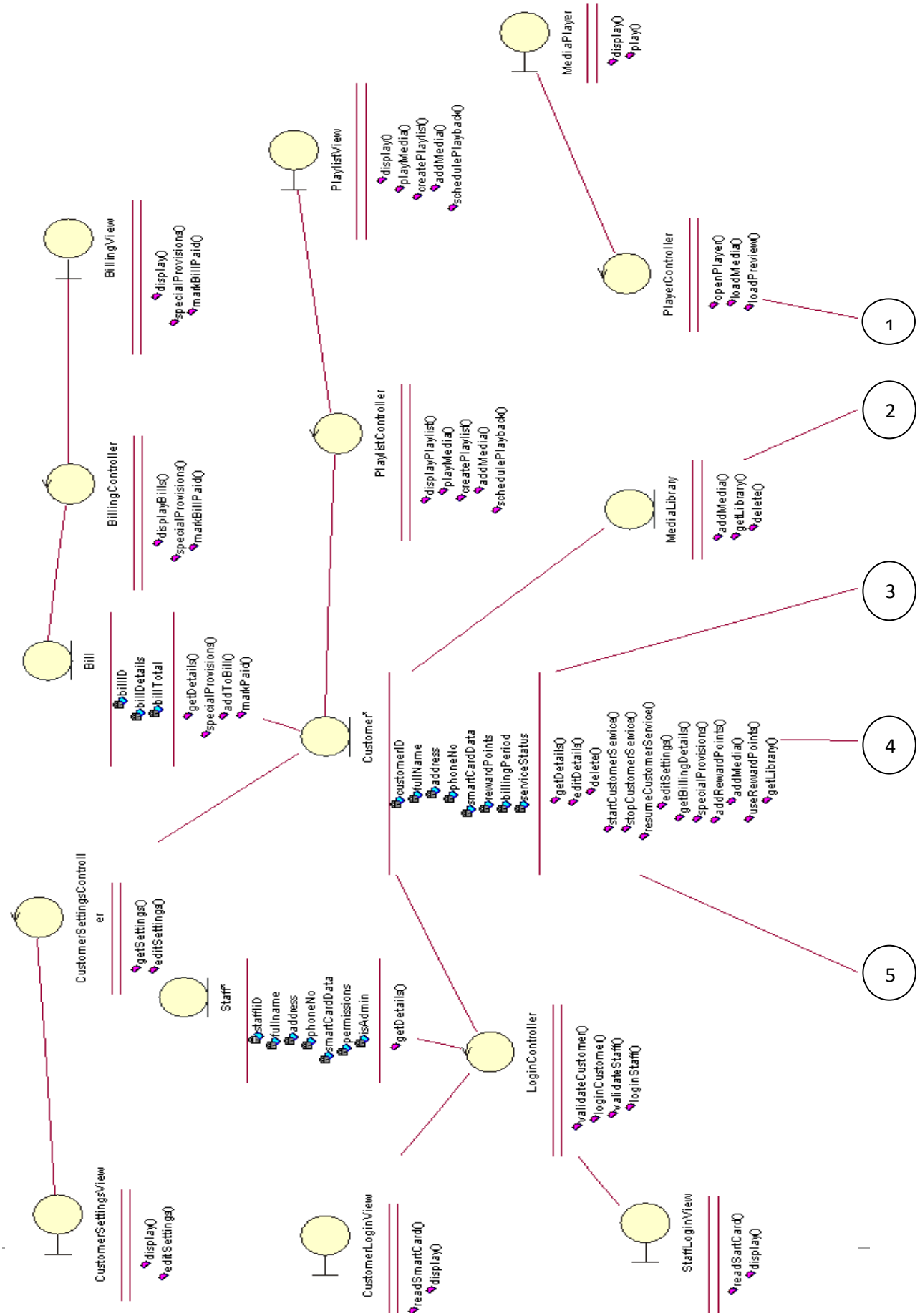
UML Diagrams

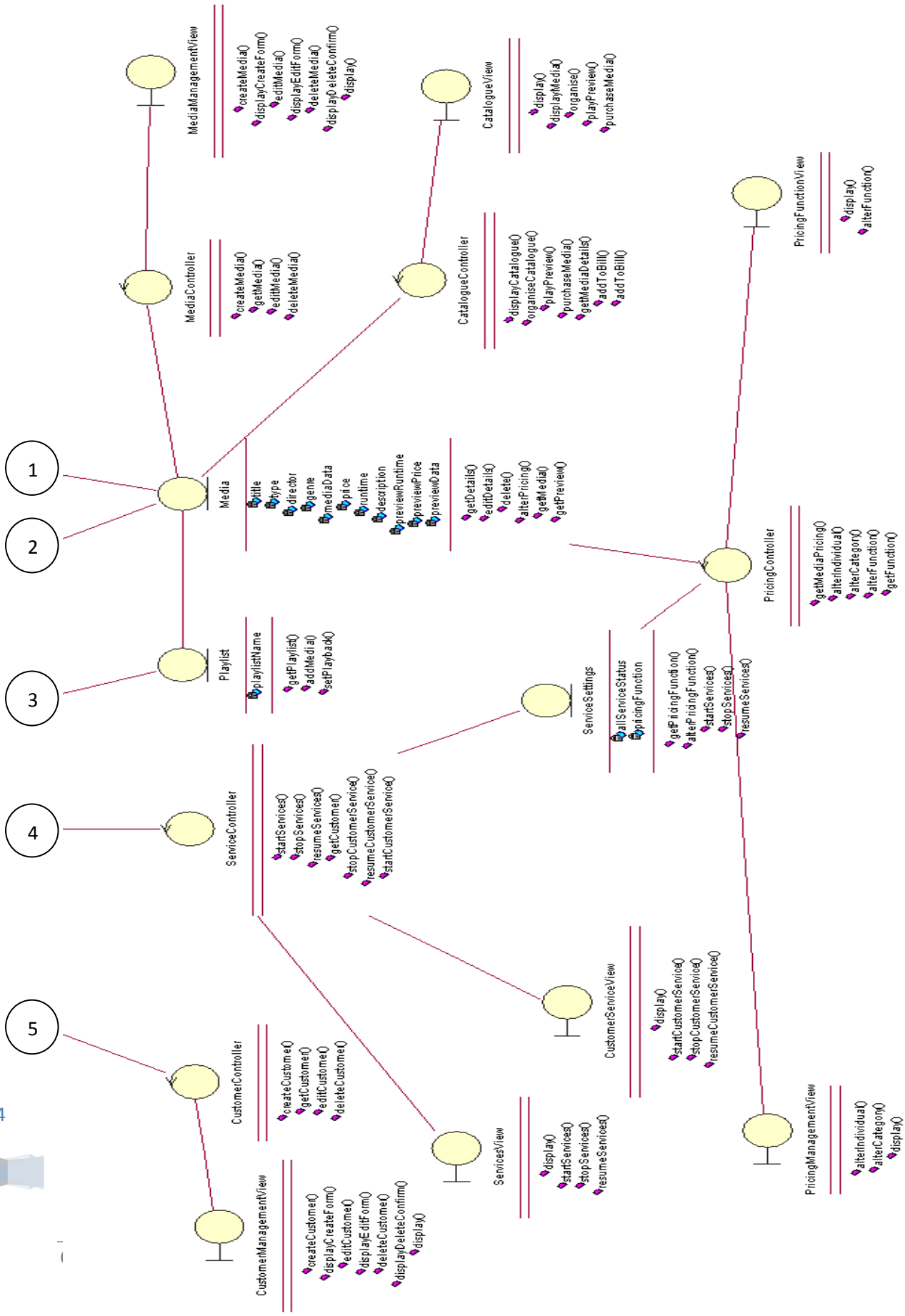
Class Diagram

The following shows an overview of the class diagram which we have designed for the @ease Neighbourhood Entertainment System.



The following two images show the same diagram enlarged for easier reading.

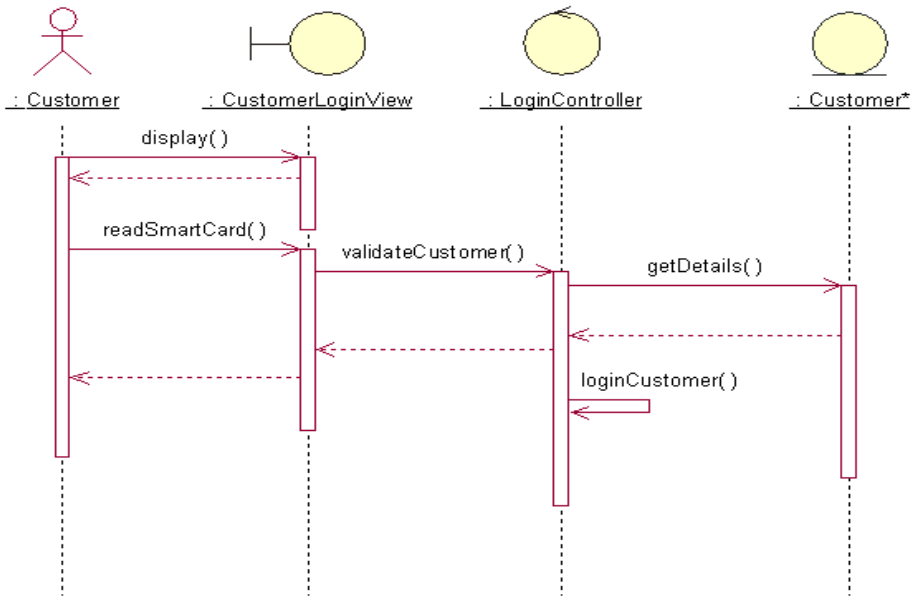

MediaPlayer
display()
play()
PlaylistView
display()
playMedia()
createPlaylist()
addMedia()
schedulePlayback()
BillingView
display()
specialProvisions()
markBillPaid()
PlayerController
openPlayer()
loadMedia()
loadPreview()
1
PlaylistController
displayPlaylist()
playMedia()
createPlaylist()
addMedia()
schedulePlayback()
BillingController
displayBills()
specialProvisions()
markBillPaid()
2
MediaLibrary
addMedia()
getLibrary()
delete()
Bill
billID
billDetails
billTotal
getDetails()
specialProvisions()
addToBill()
markPaid()
3
Customer*
customerID
fullName
address
phoneNo
smartCardData
rewardPoints
billingPeriod
serviceStatus
getDetails()
editDetails()
delete()
startCustomerService()
stopCustomerService()
resumeCustomerService()
editSettings()
getBillingDetails()
specialProvisions()
addRewardPoints()
addMedia()
useRewardPoints()
getLibrary()
4
CustomerSettingsControll
er
getSettings()
editSettings()
Staff*
staffID
fullname
address
phoneNo
smartCardData
permissions
isAdmin
getDetails()
LoginController
validateCustomer()
loginCustomer()
validateStaff()
loginStaff()
5
CustomerSettingsView
display()
editSettings()
CustomerLoginView
readSmartCard()
display()
StaffLoginView
readSartCard()
display()


1
2
3
4
5
Media
title
type
director
genre
mediaData
price
runtime
description
previewRunTime
previewPrice
previewData
getDetails()
editDetails()
delete()
alterPricing()
getMedia()
getPreview()
Playlist
playlistName
getPlaylist()
addMedia()
setPlayback()
MediaController
createMedia()
getMedia()
editMedia()
deleteMedia()
MediaManagementView
createMedia()
displayCreateForm()
editMedia()
displayEditForm()
deleteMedia()
displayDeleteConfirm()
display()
CatalogueController
displayCatalogue()
organiseCatalogue()
playPreview()
purchaseMedia()
getMediaDetails()
addToBill()
addToBill()
CatalogueView
display()
displayMedia()
organise()
playPreview()
purchaseMedia()
PricingController
getMediaPricing()
alterIndividual()
alterCategor()
alterFunction()
getFunction()
PricingFunctionView
display()
alterFunction()
ServiceSettings
allServiceStatus
pricingFunction
getPricingFunction()
alterPricingFunction()
startServices()
stopServices()
resumeServices()
ServiceController
startServices()
stopServices()
resumeServices()
getCustomer()
stopCustomerService()
resumeCustomerService()
startCustomerService()
CustomerController
createCustomer()
getCustomer()
editCustomer()
deleteCustomer()
CustomerManagementView
createCustomer()
displayCreateForm()
editCustomer()
displayEditForm()
deleteCustomer()
displayDeleteConfirm()
display()
CustomerServiceView
display()
startCustomerService()
stopCustomerService()
resumeCustomerService()
ServicesView
display()
startServices()
stopServices()
resumeServices()
PricingManagementView
alterIndividual()
alterCategor()
display()

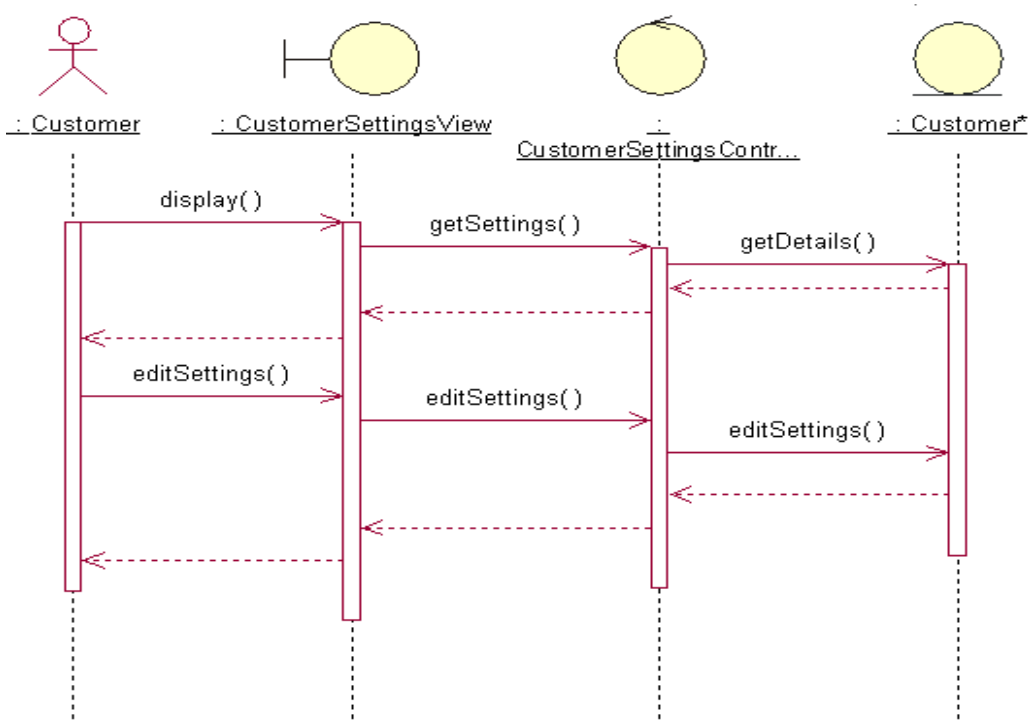
Sequence Diagrams

The following are the sequence diagrams corresponding to the use cases associated with the customer actor.

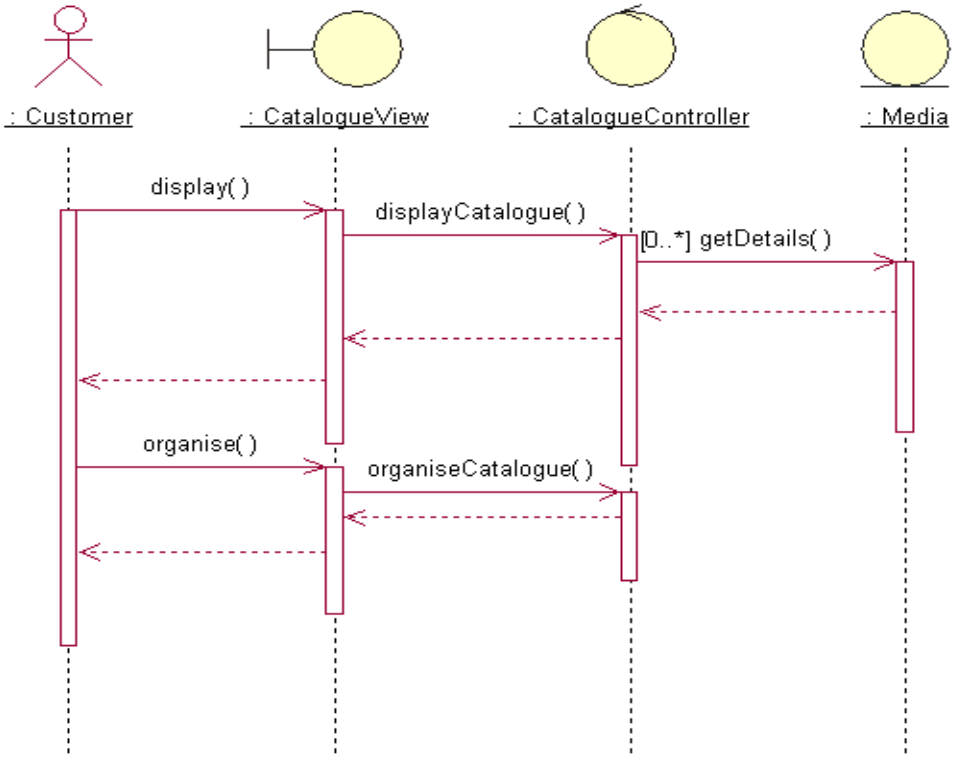
Login as Customer



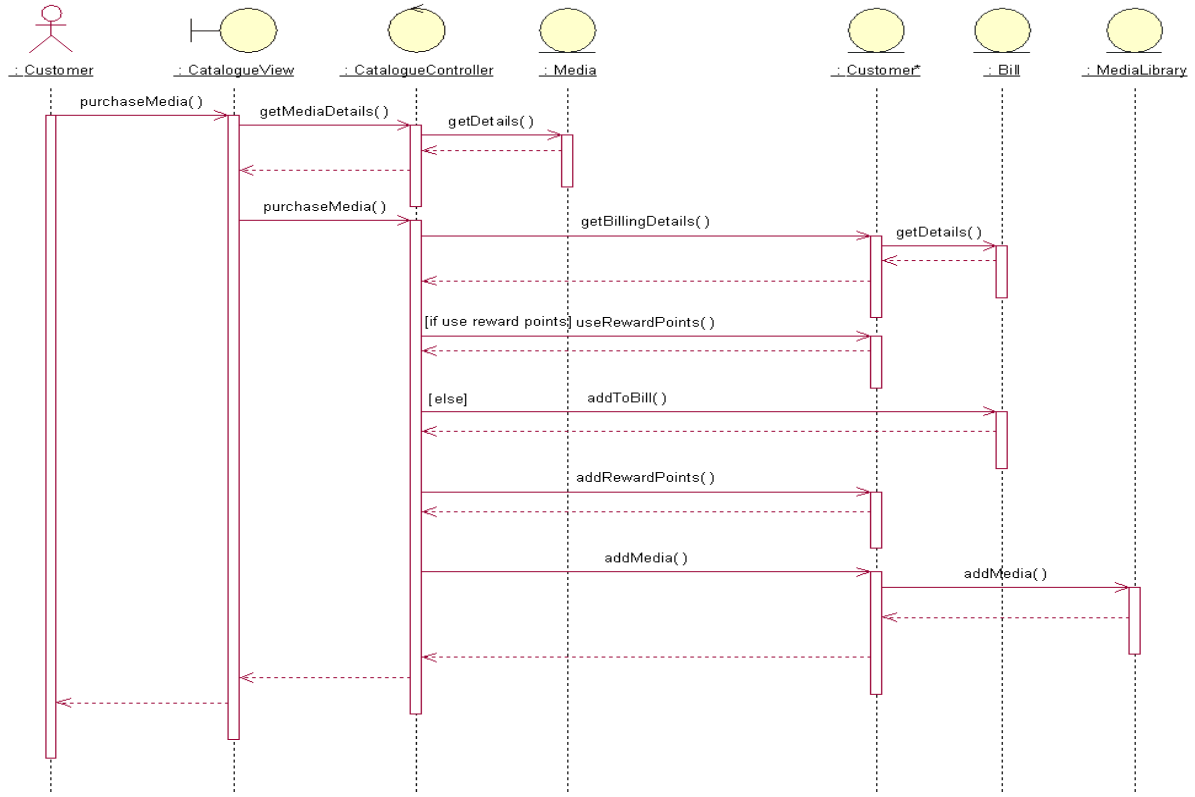
Edit Account Settings



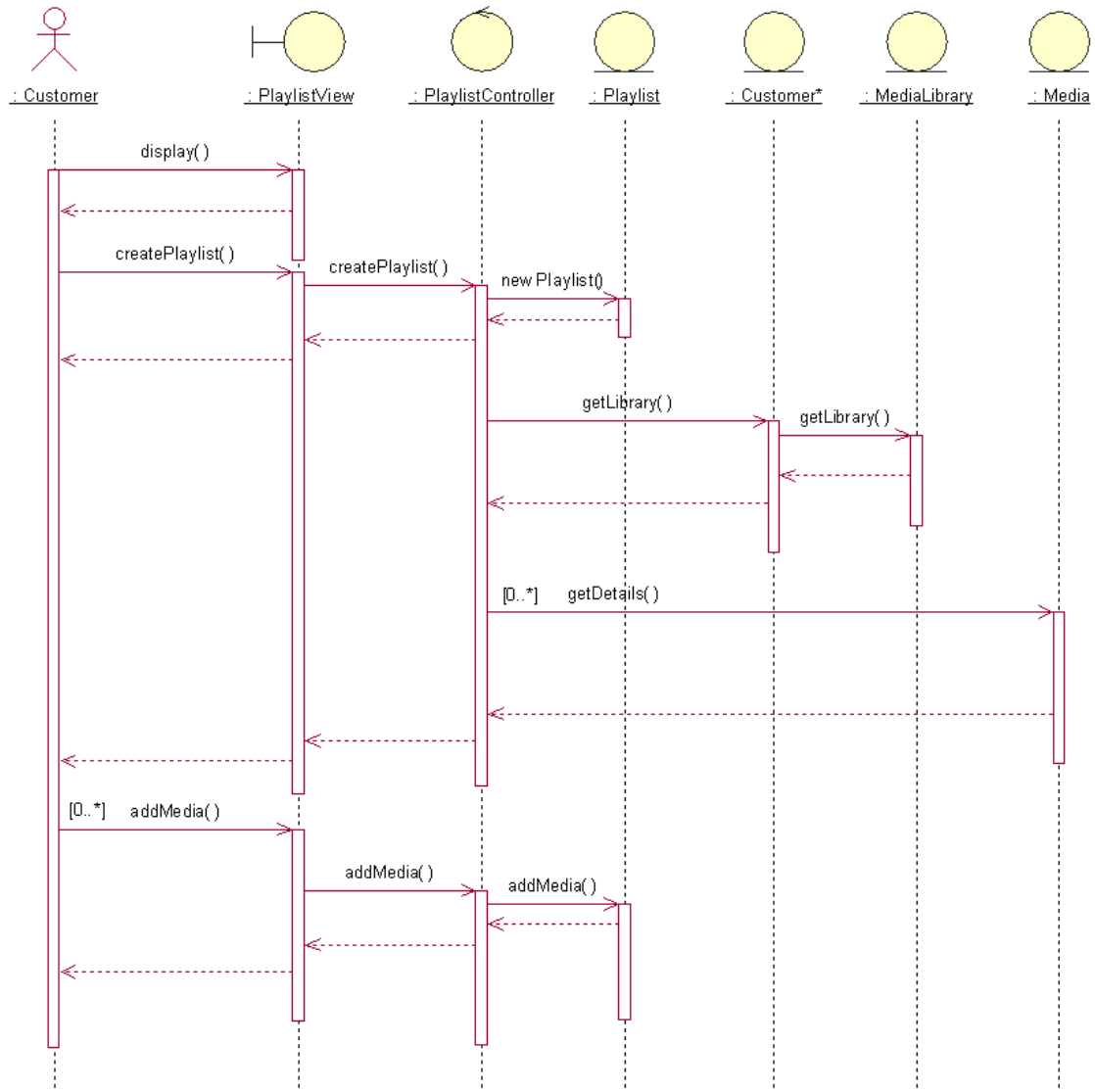
Browse Catalogue / Organise Catalogue



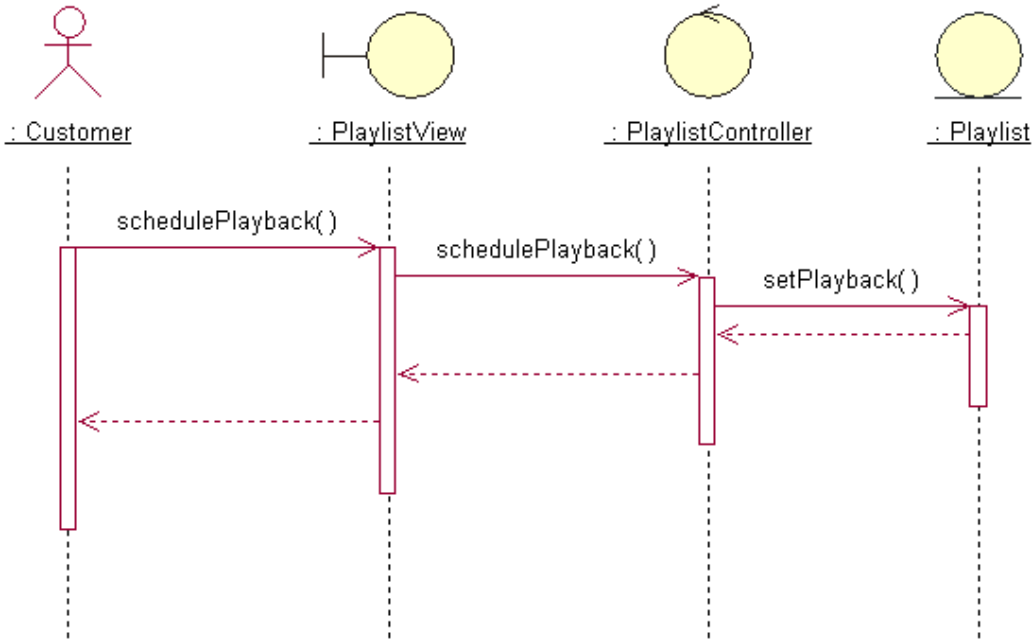
Purchase Media / Earn Reward Points / Use Reward Points



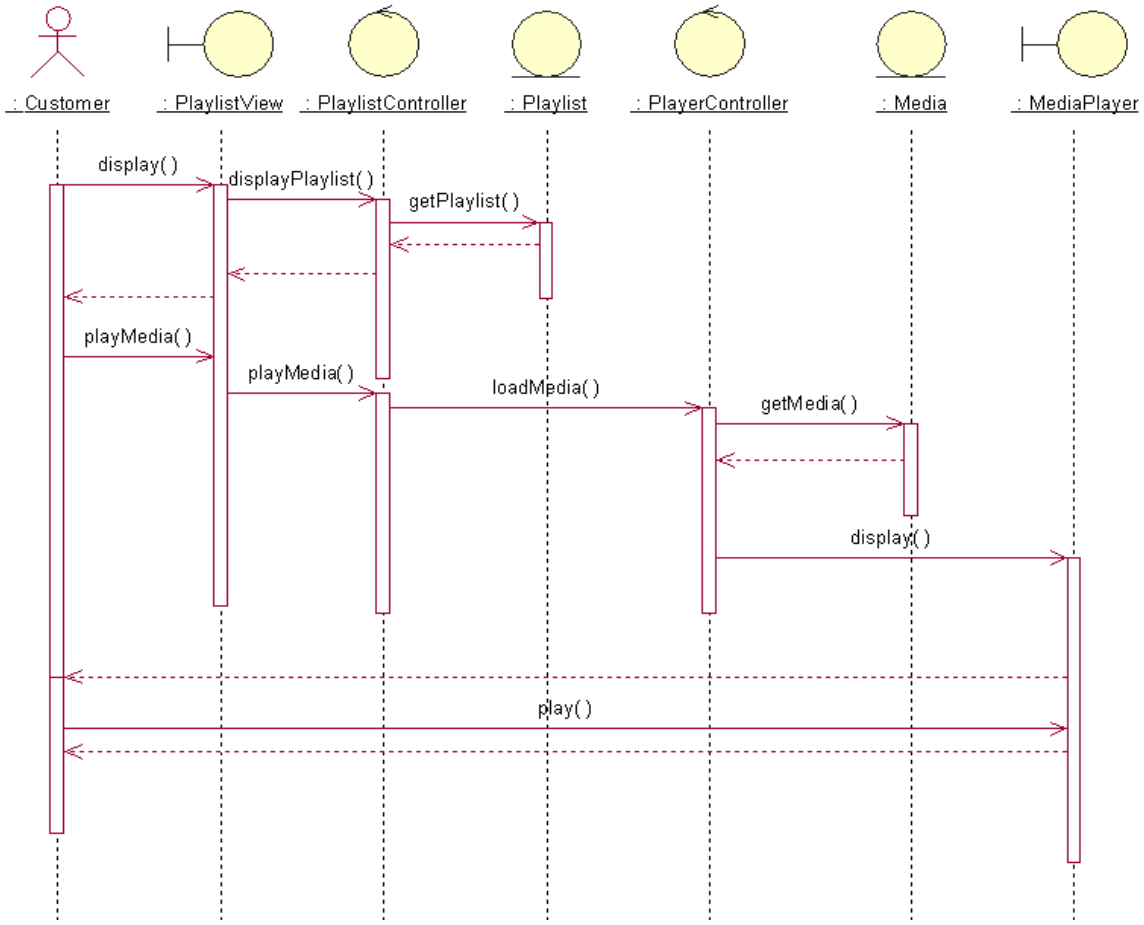
Create Playlist



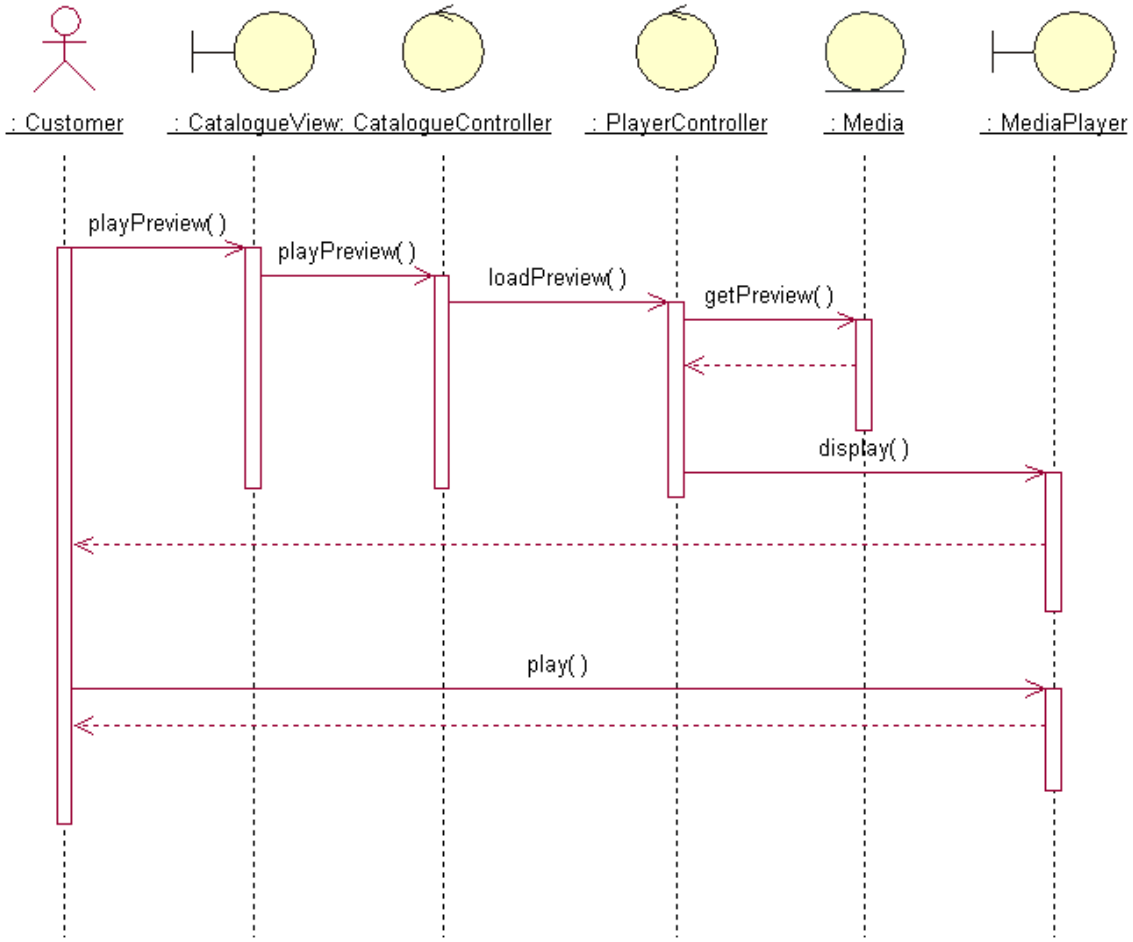
Schedule Playback



Watch Media

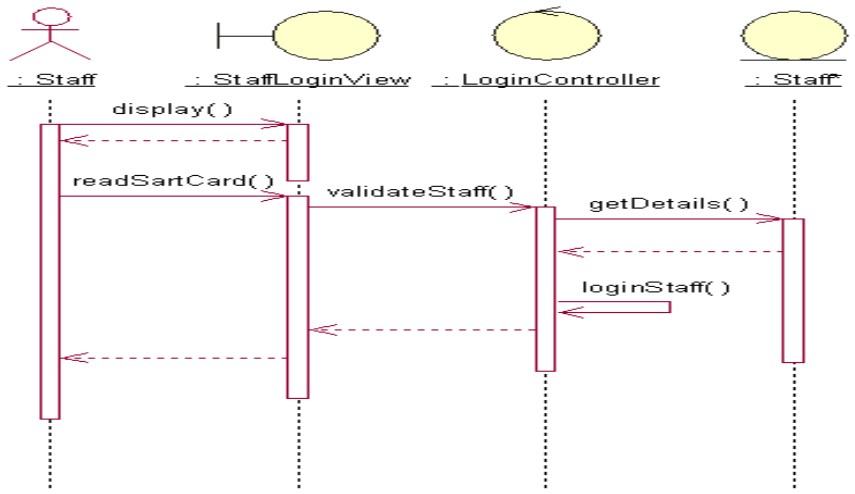

: Customer
: PlaylistView
: PlaylistController
: Playlist
: PlayerController
: Media
: MediaPlayer
display( )
displayPlaylist( )
getPlaylist( )
playMedia( )
playMedia( )
loadMedia( )
getMedia( )
display( )
play( )

Watch Preview

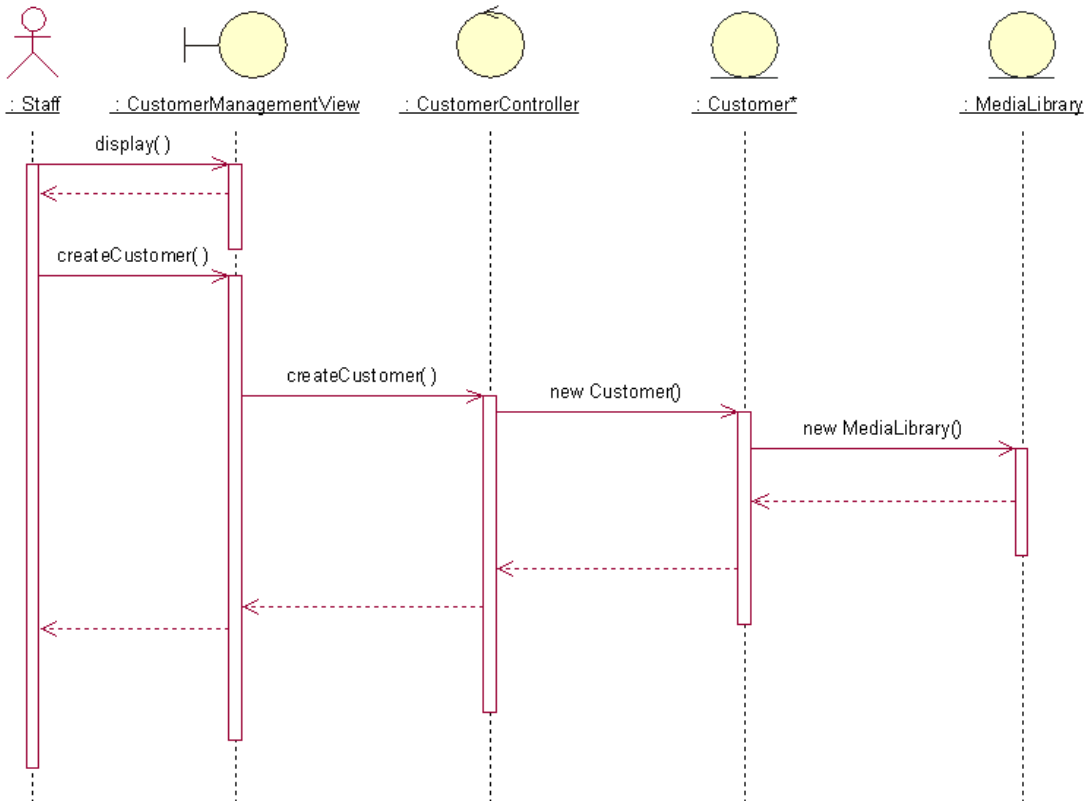


The following are the sequence diagrams corresponding to use cases associated with the staff actor.

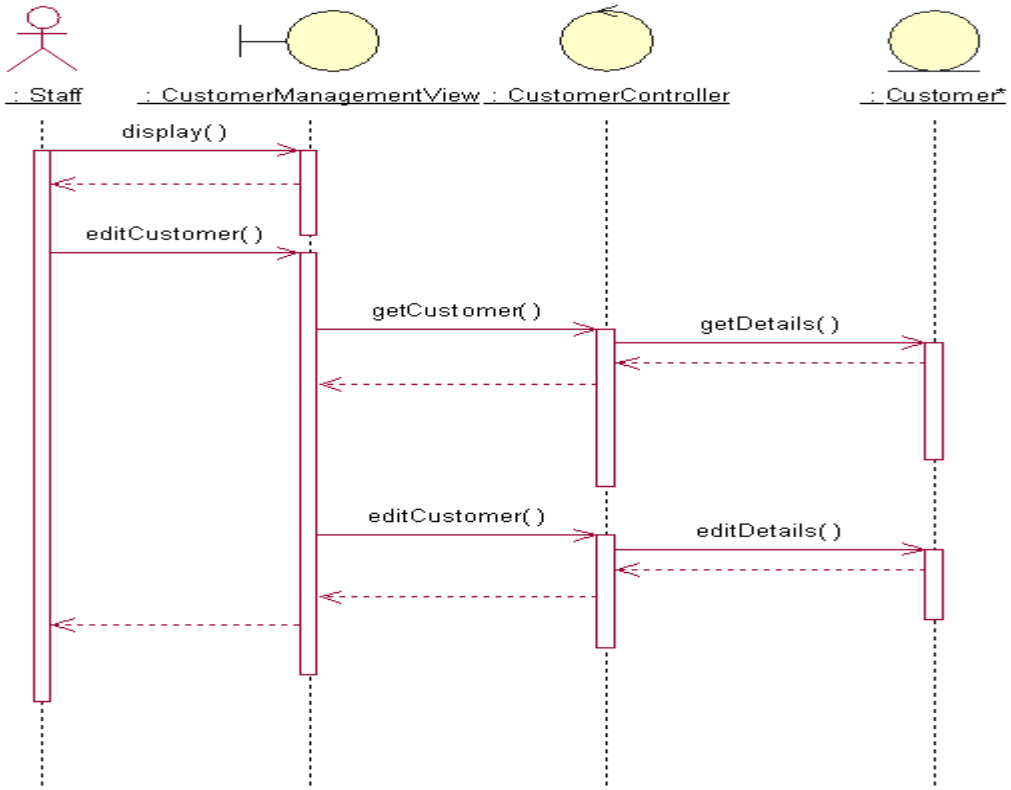
Login as Staff



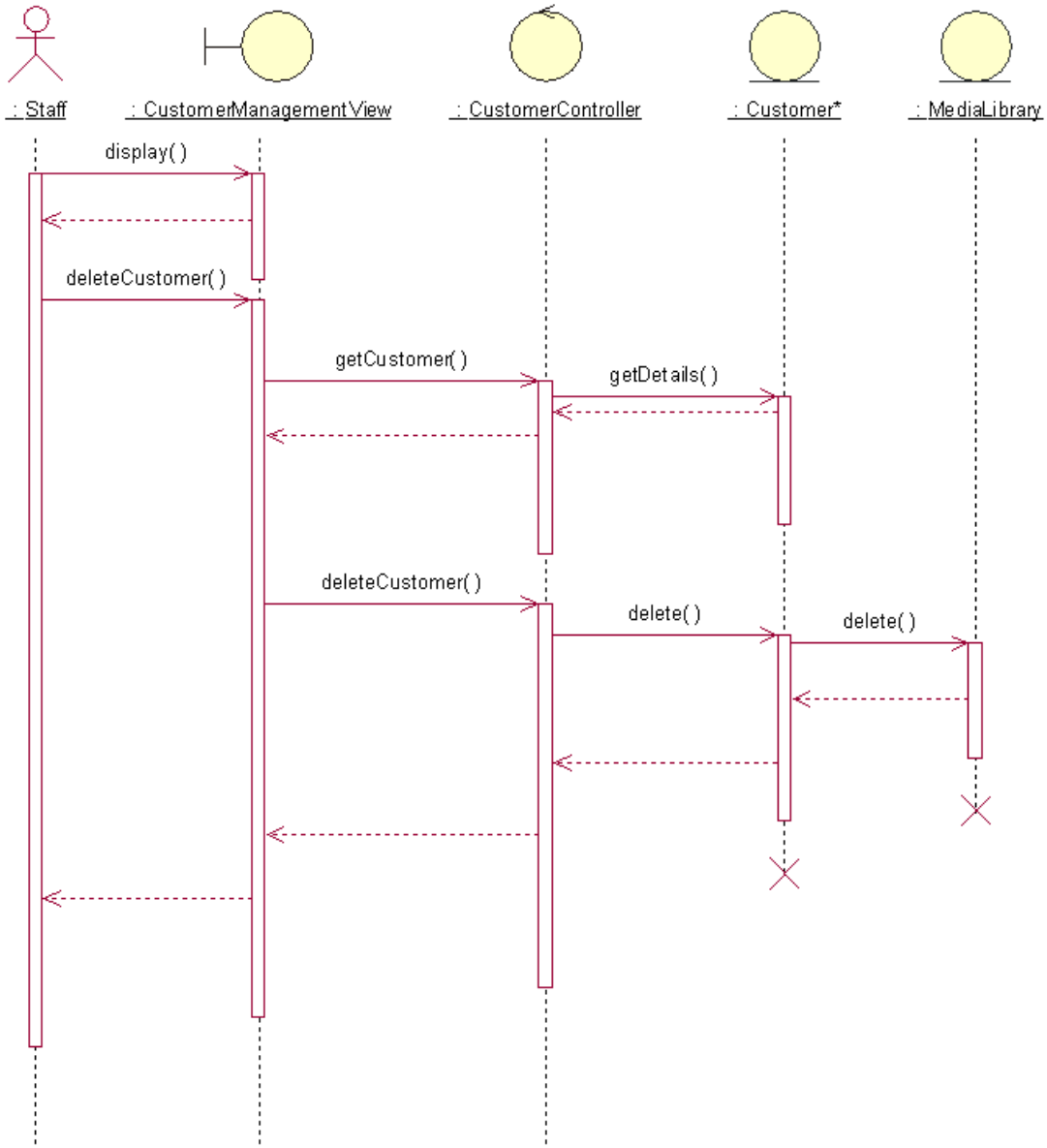
Create Customer



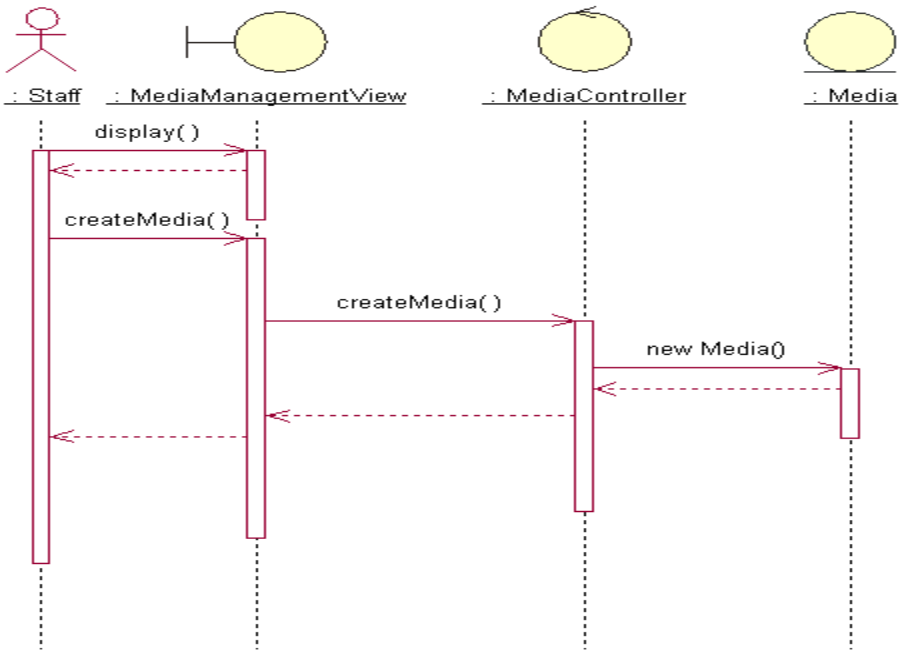
Update Customer



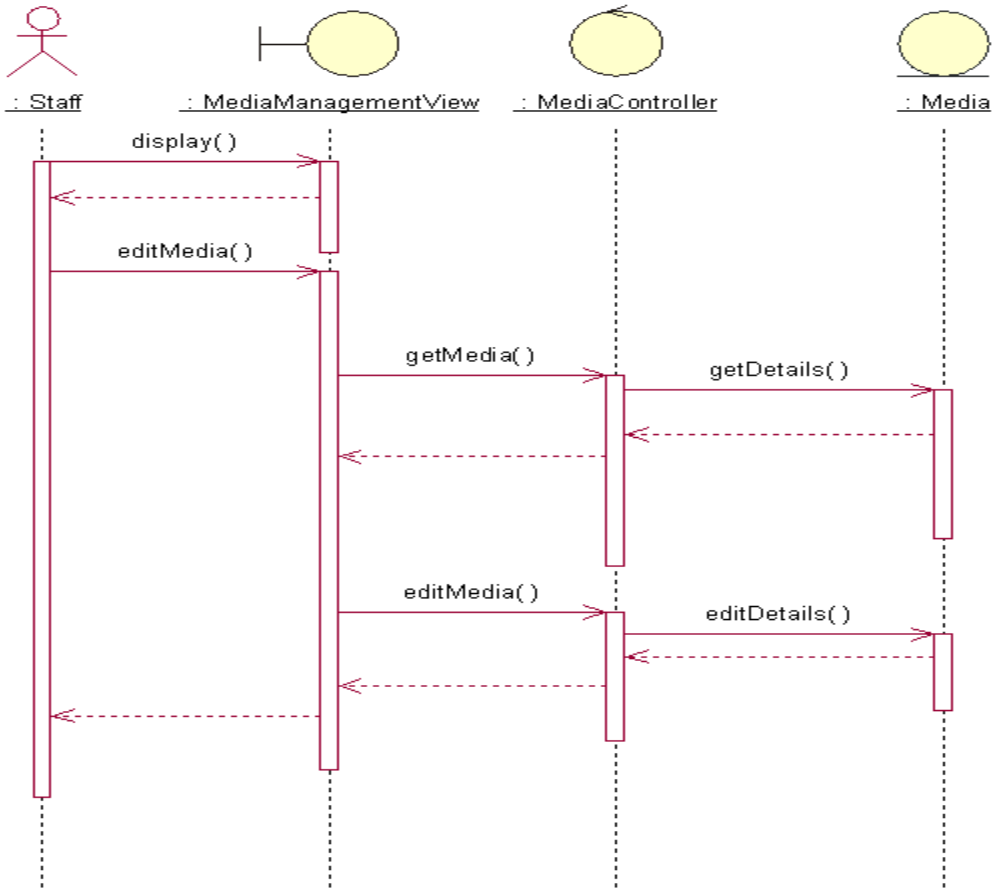
Delete Customer



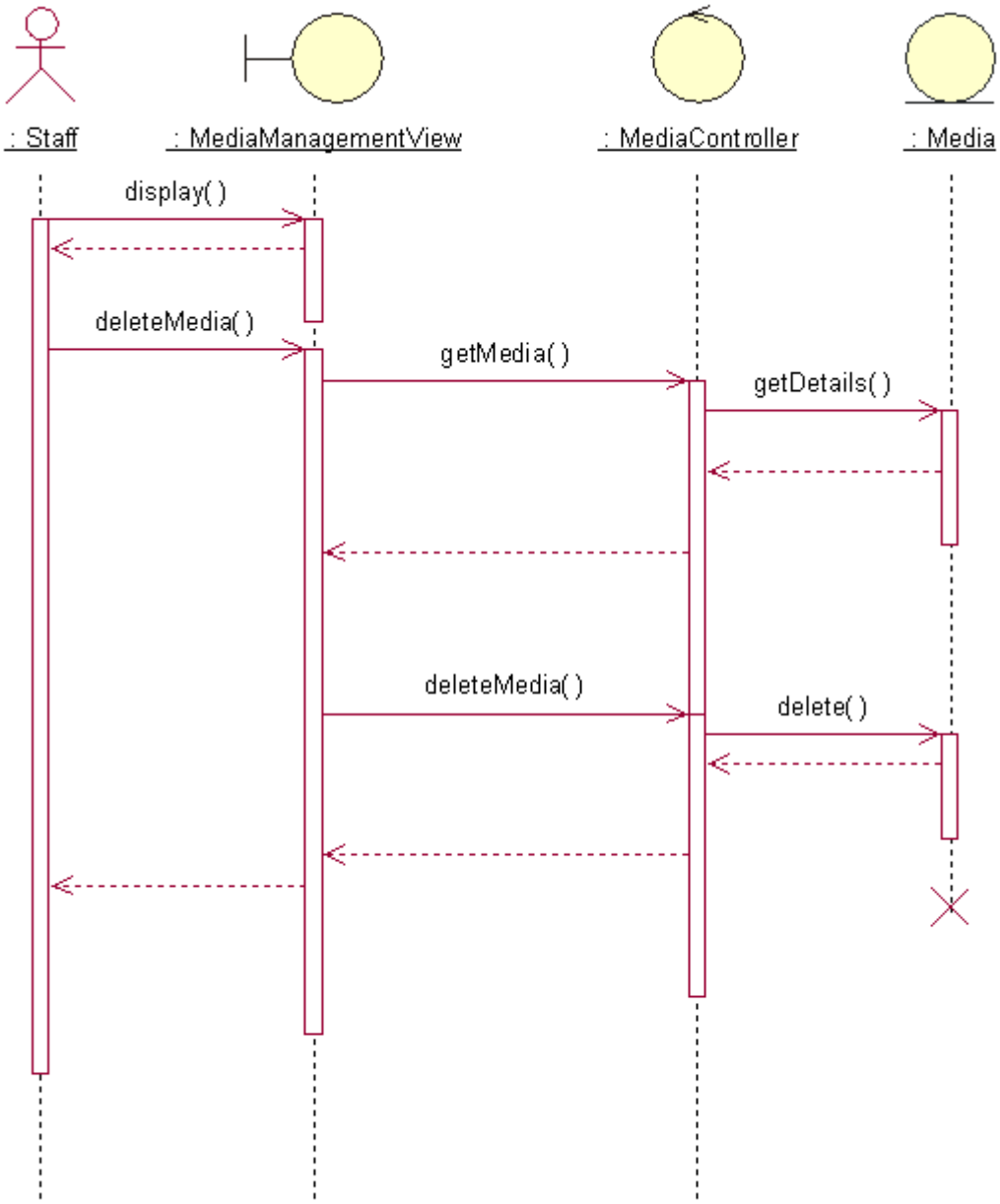
Create Recording



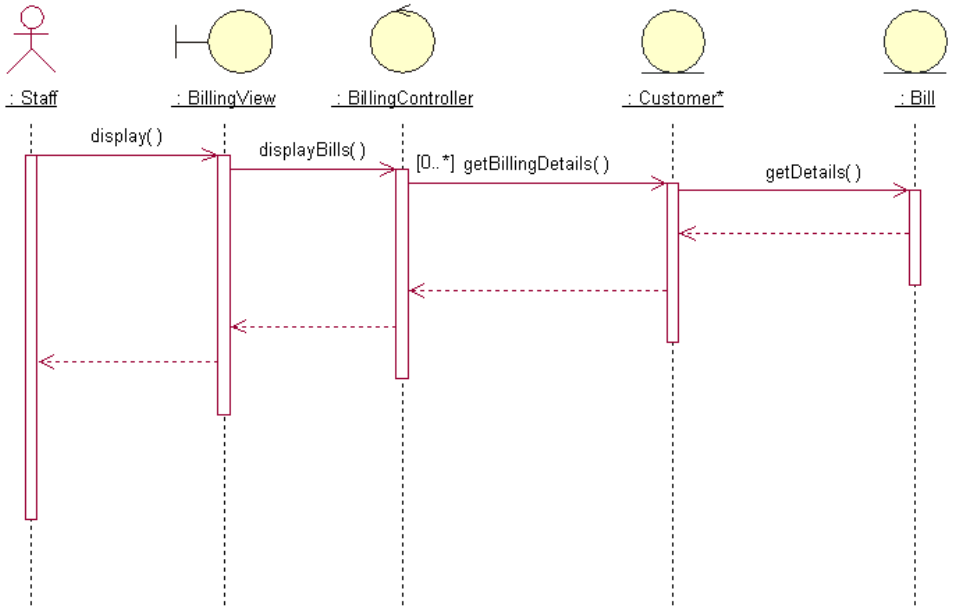
Update Recording



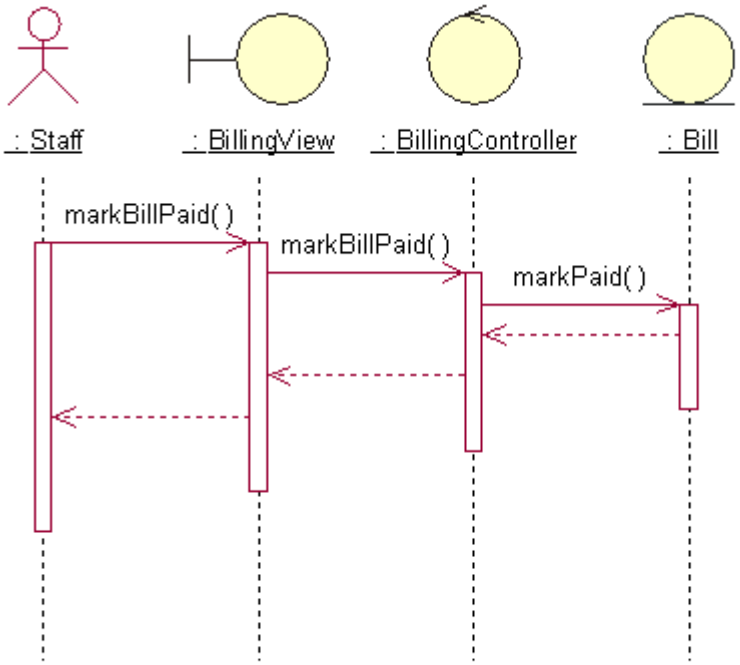
Delete Recording



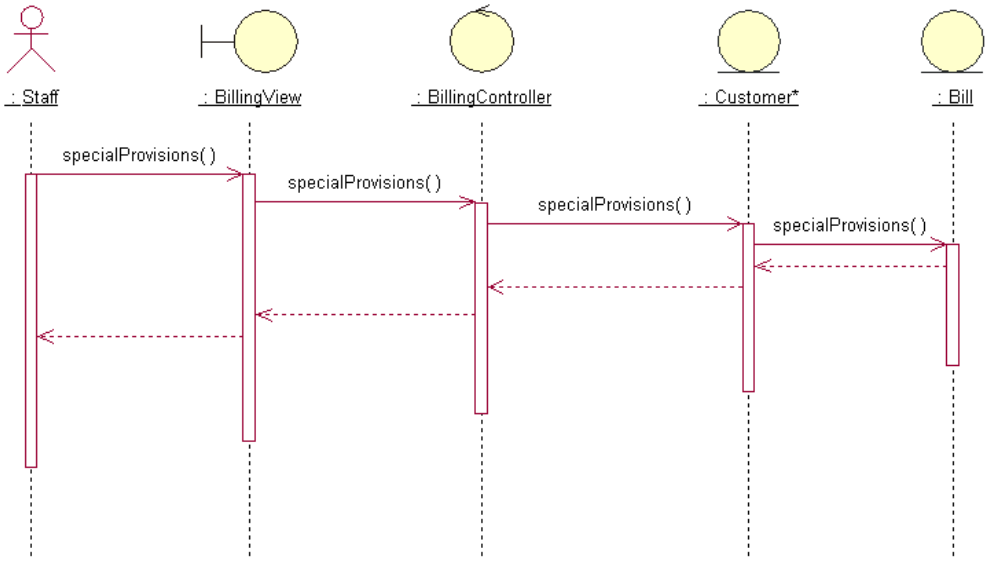
Browse Billing



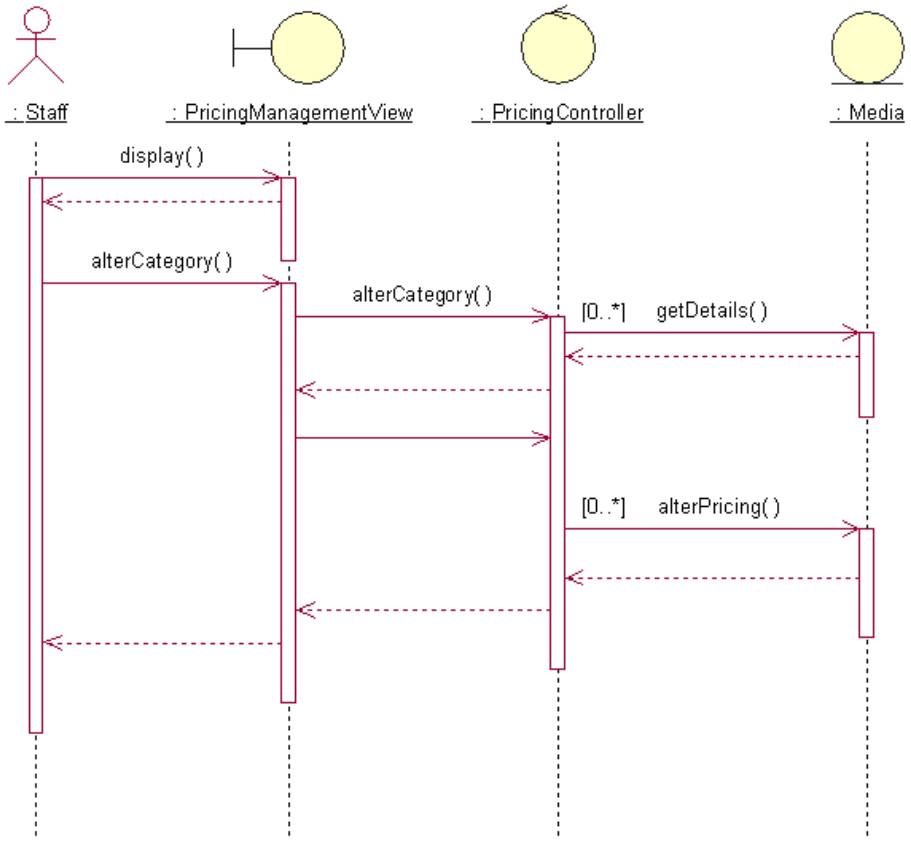
Mark Bill Paid



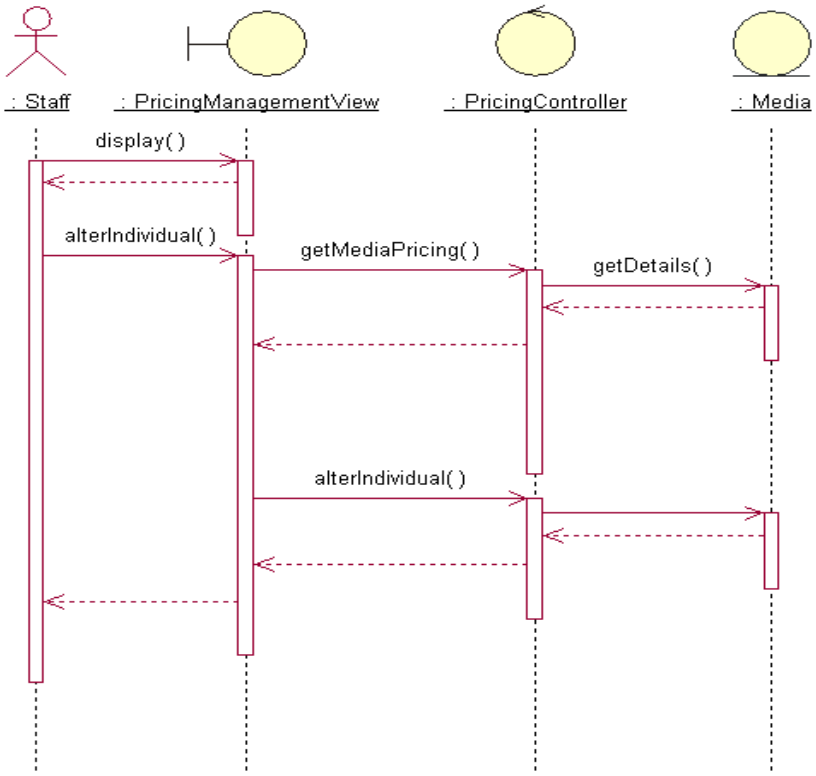
Apply Special Billing Provisions



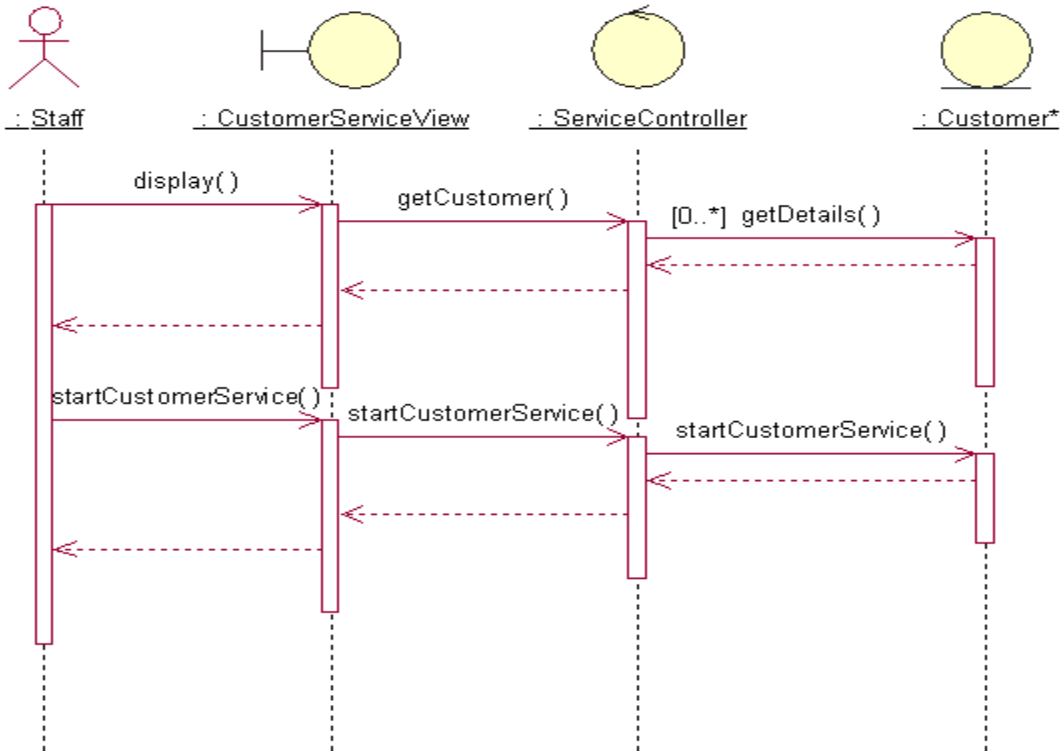
Alter Category Pricing



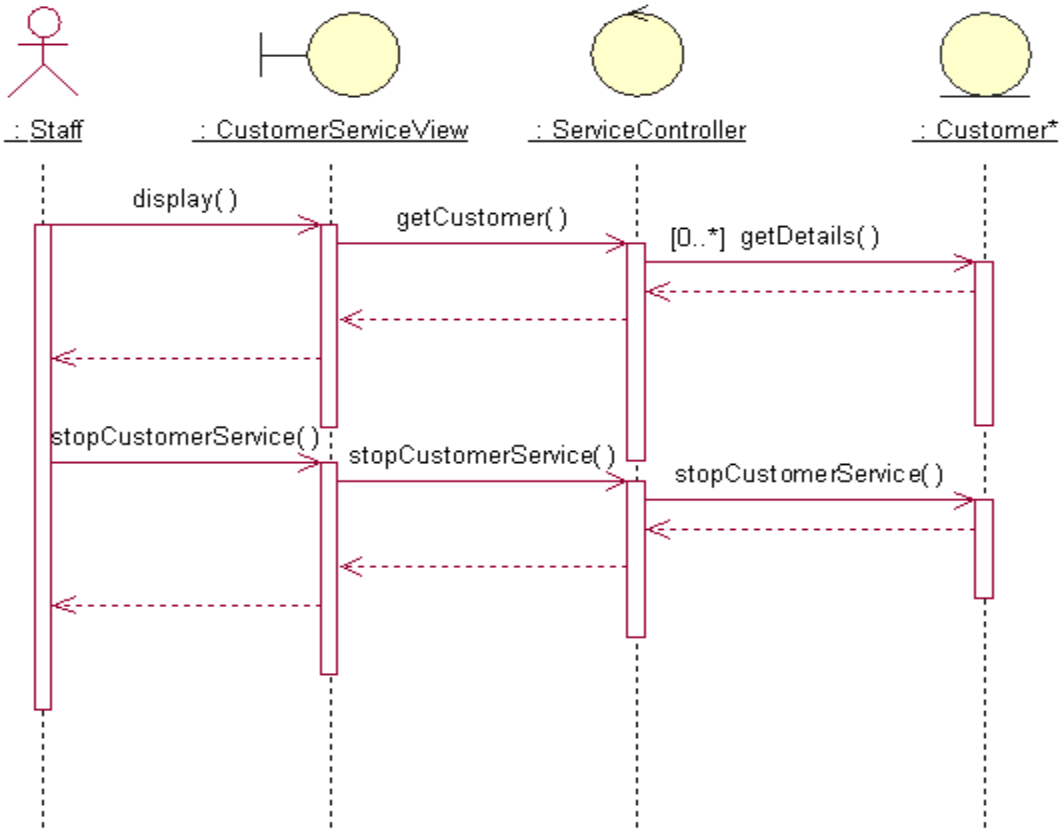
Alter Individual Pricing



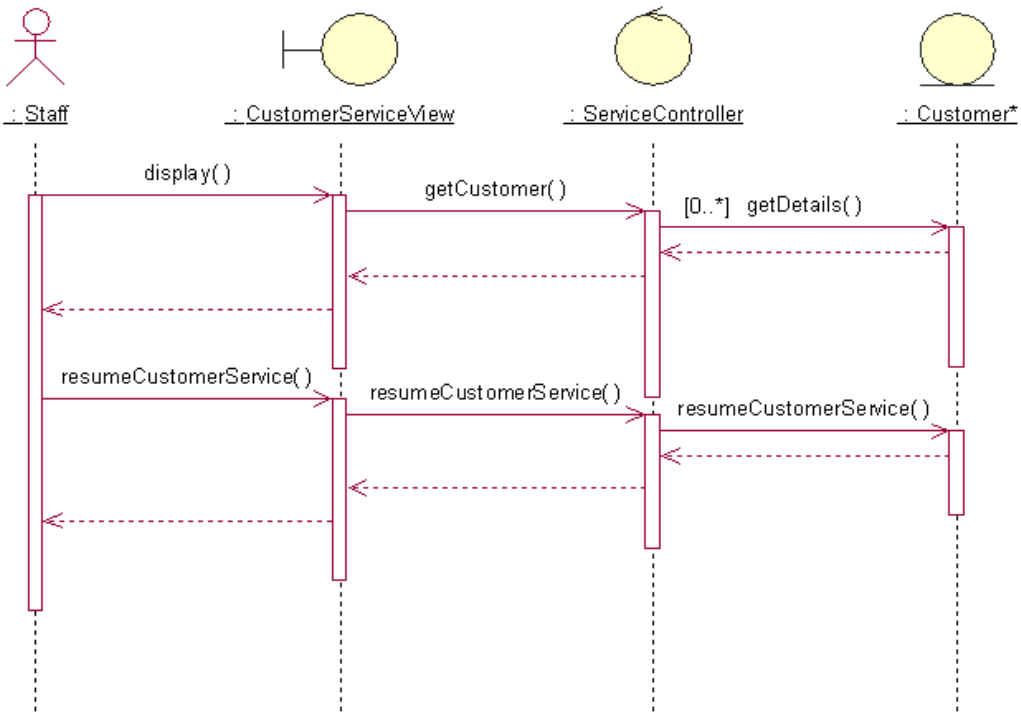
Start Service (for individual customer)



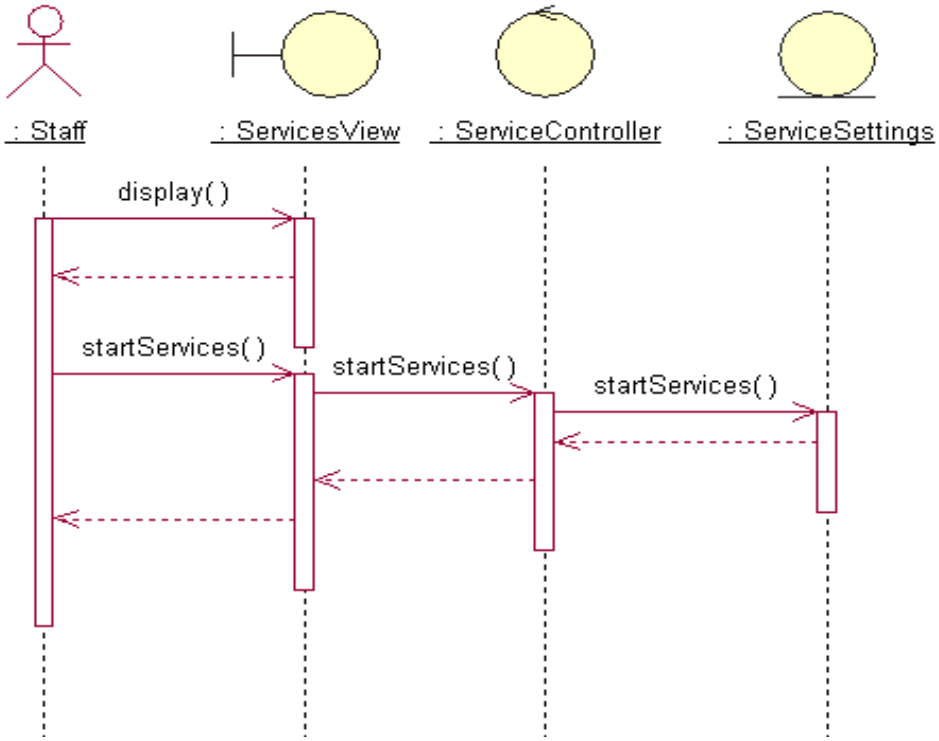
Stop Service (for individual customer)



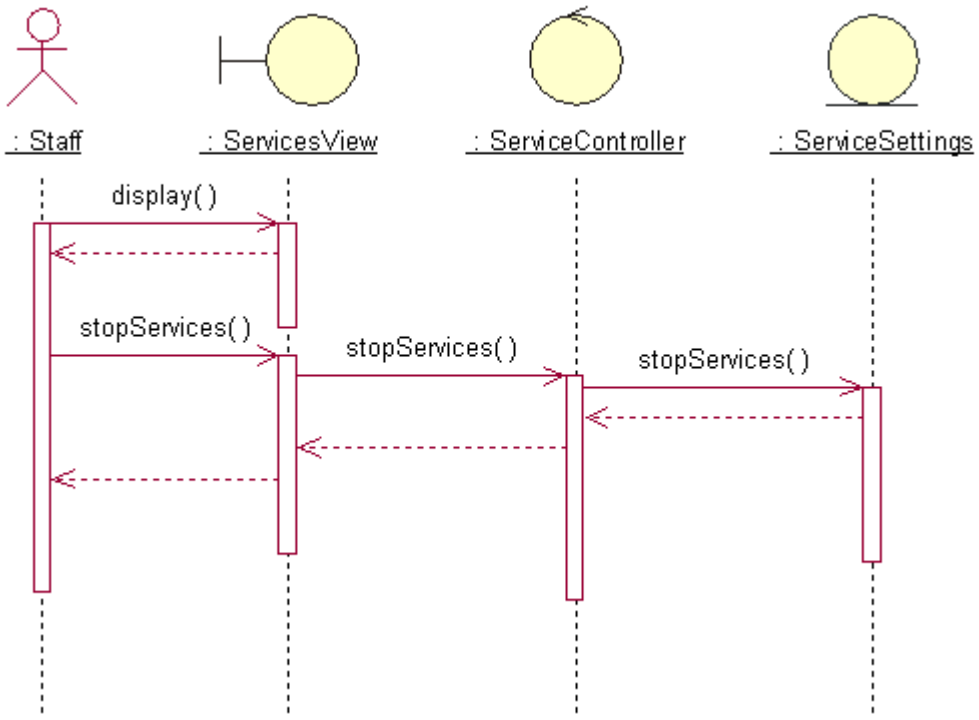
Resume Service (for individual customer)



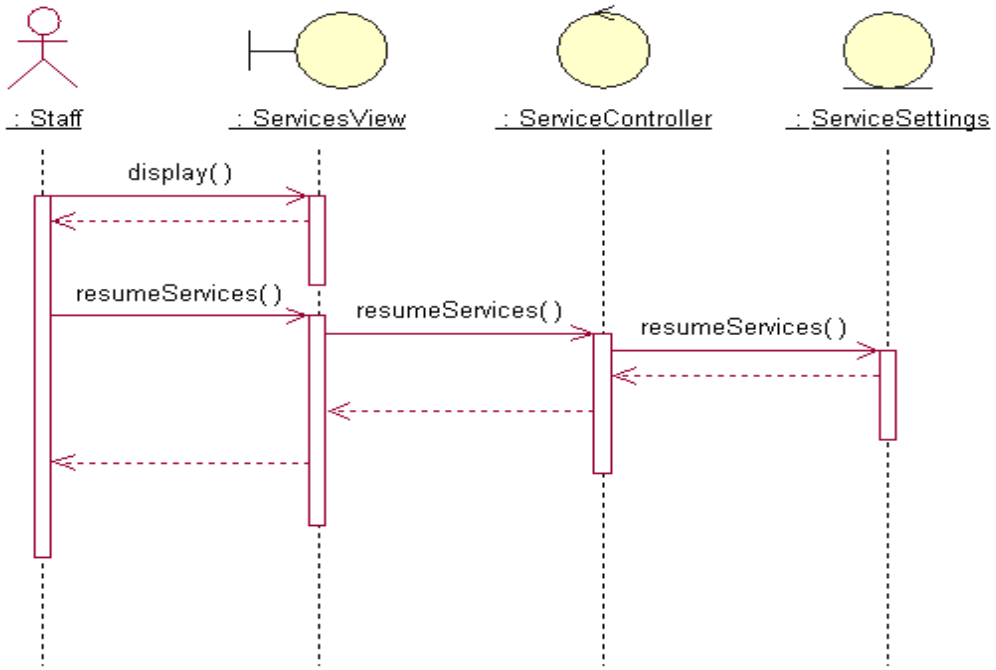
Start All Services



Stop All Services

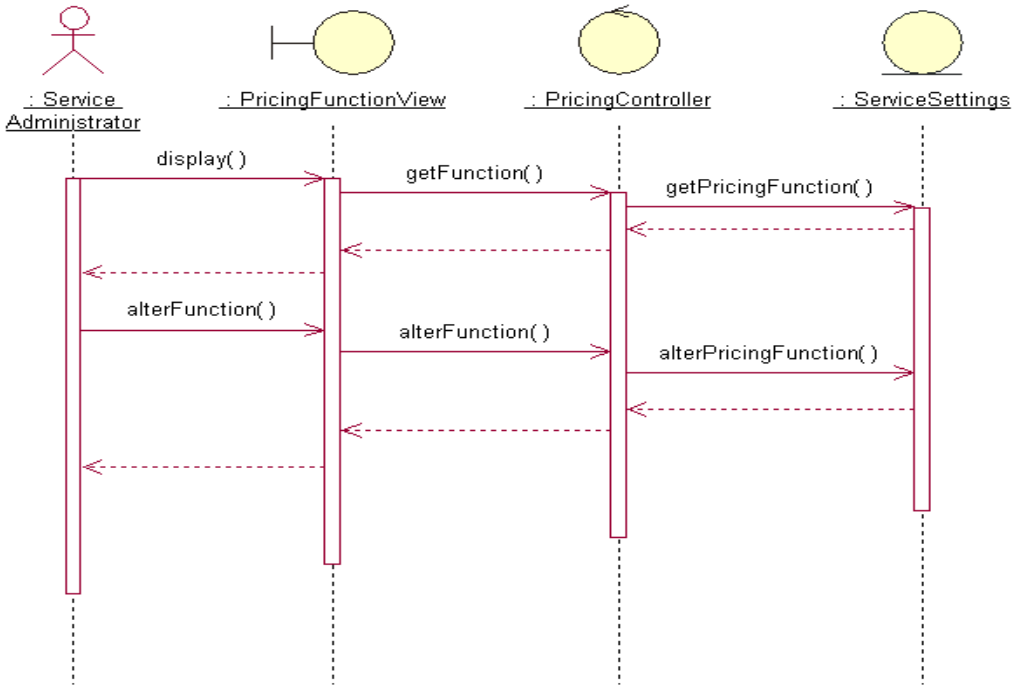


Resume All Services



The following is the sequence diagram corresponding to the use case associated with the service admin actor.

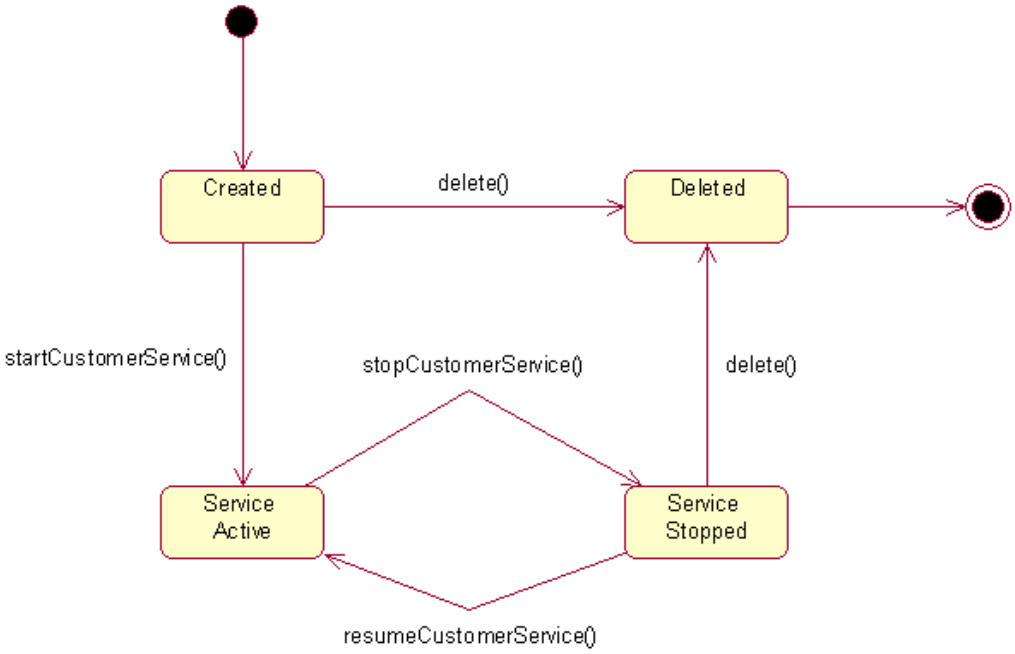
Alter Pricing Function



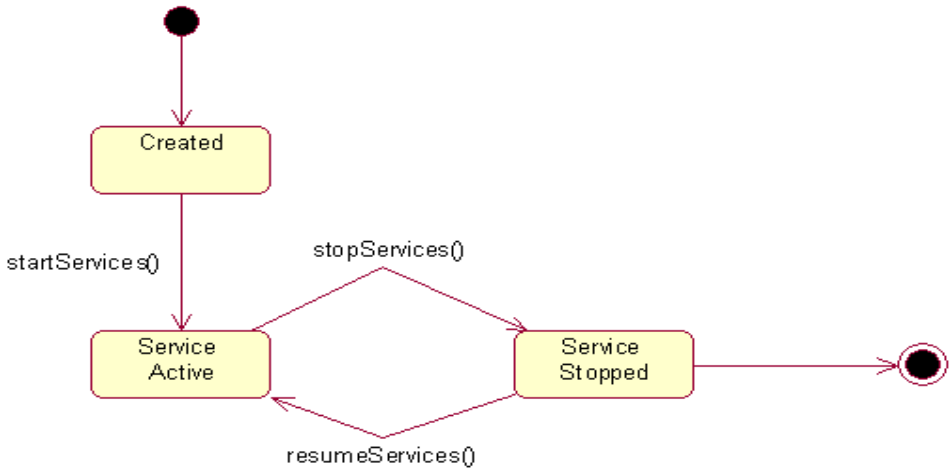
State Diagrams

The following are the state diagrams for the objects in the system which has meaningful states.

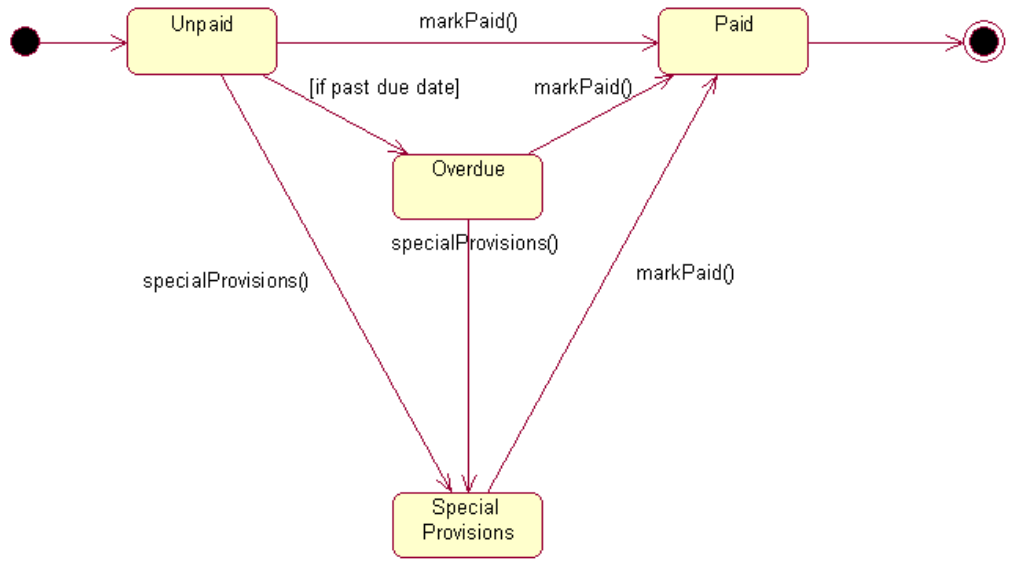
Customer



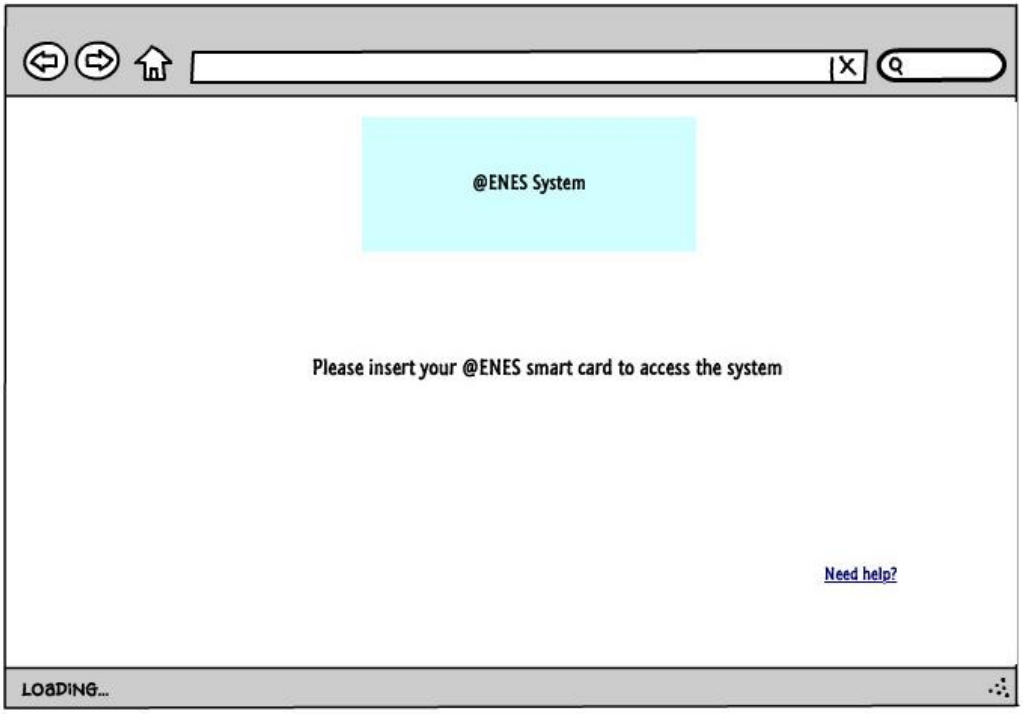
Service Settings



Bill

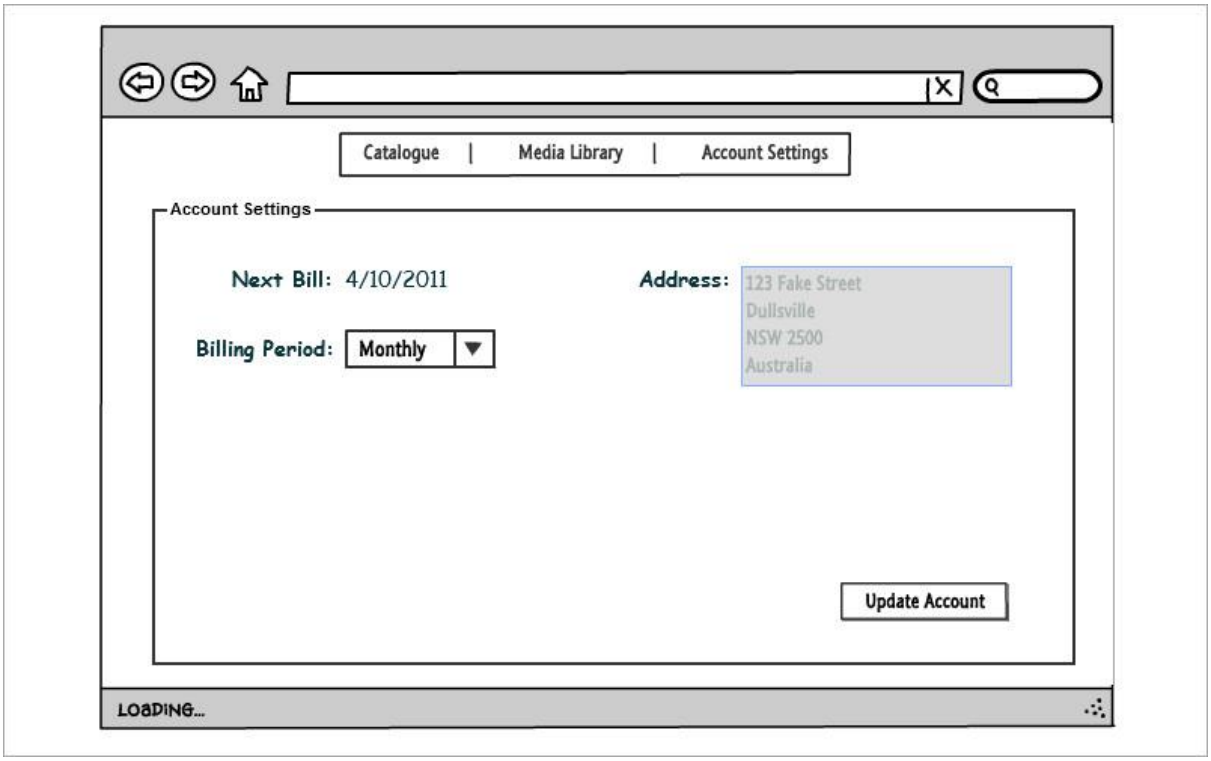

Unpaid
markPaid()
Paid
[if past due date]
markPaid()
Overdue
specialProvisions()
specialProvisions()
markPaid()
Special Provisions

User Interface Design

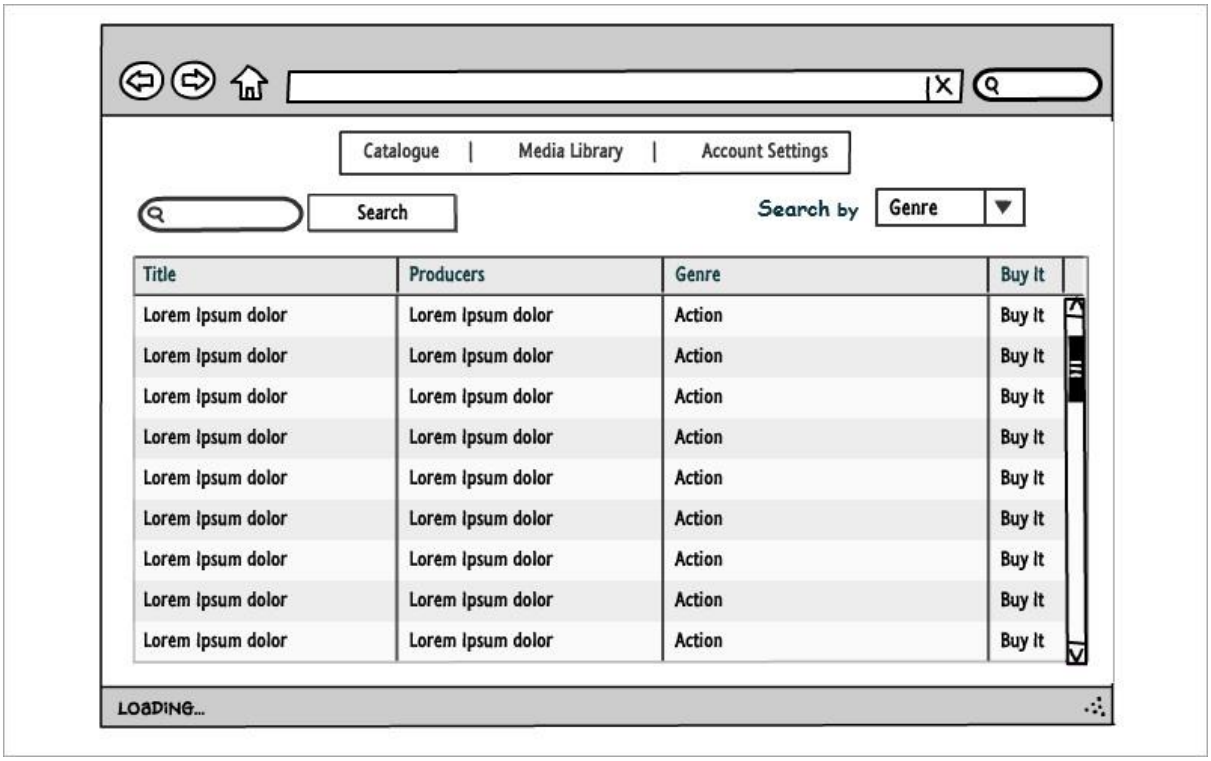


The login screen requests customers to insert their smart card

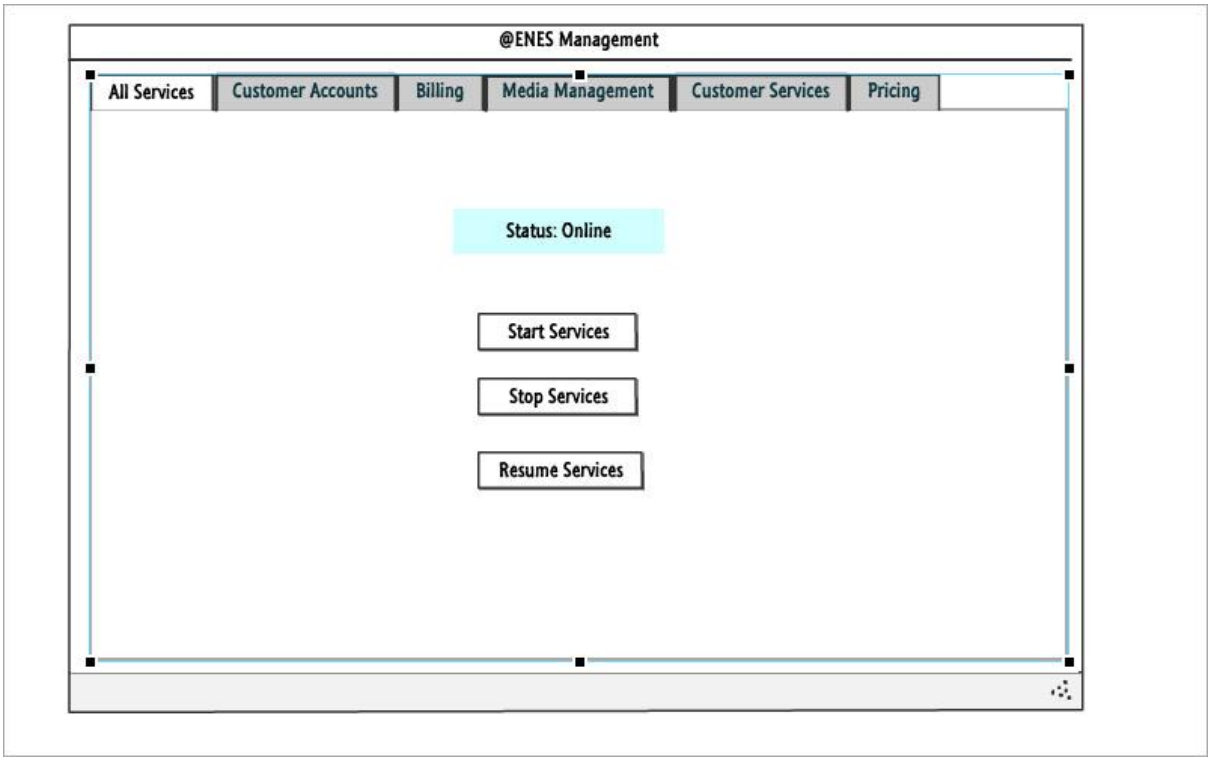
The Media Playback page has the media player with navigation elements



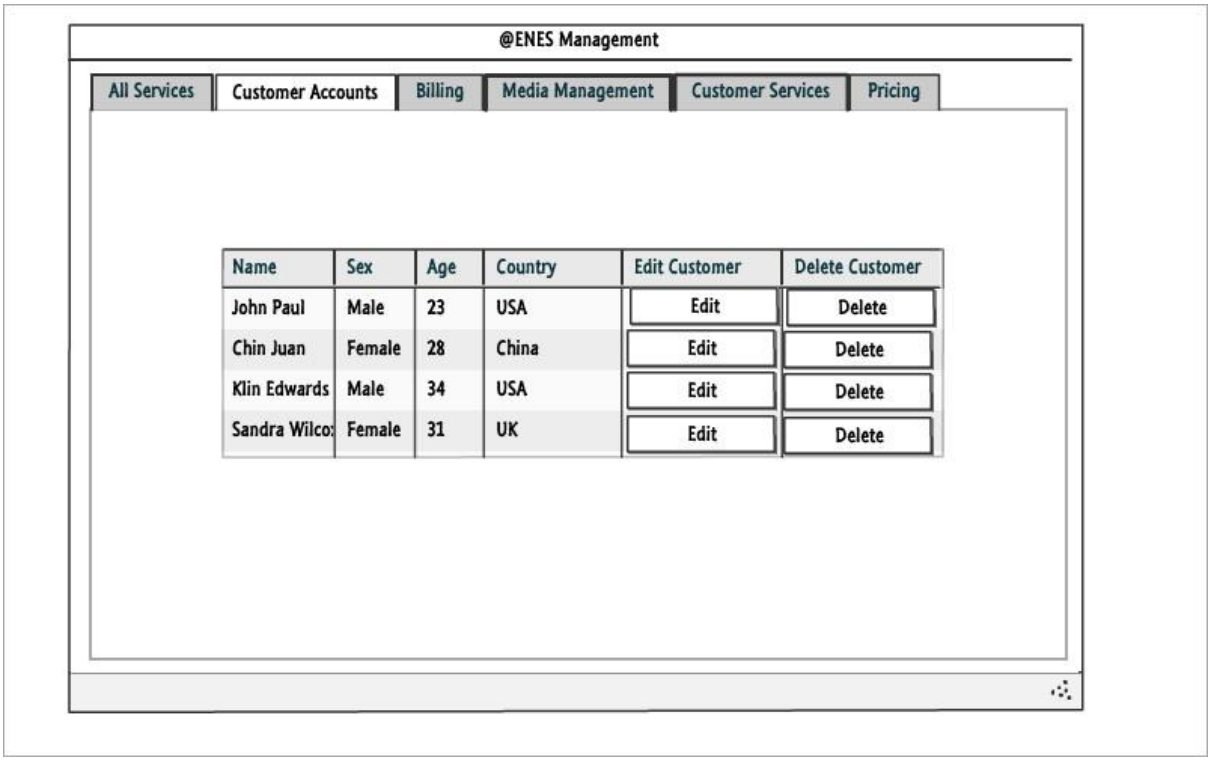
The Account Settings page allows customers to update their personal settings



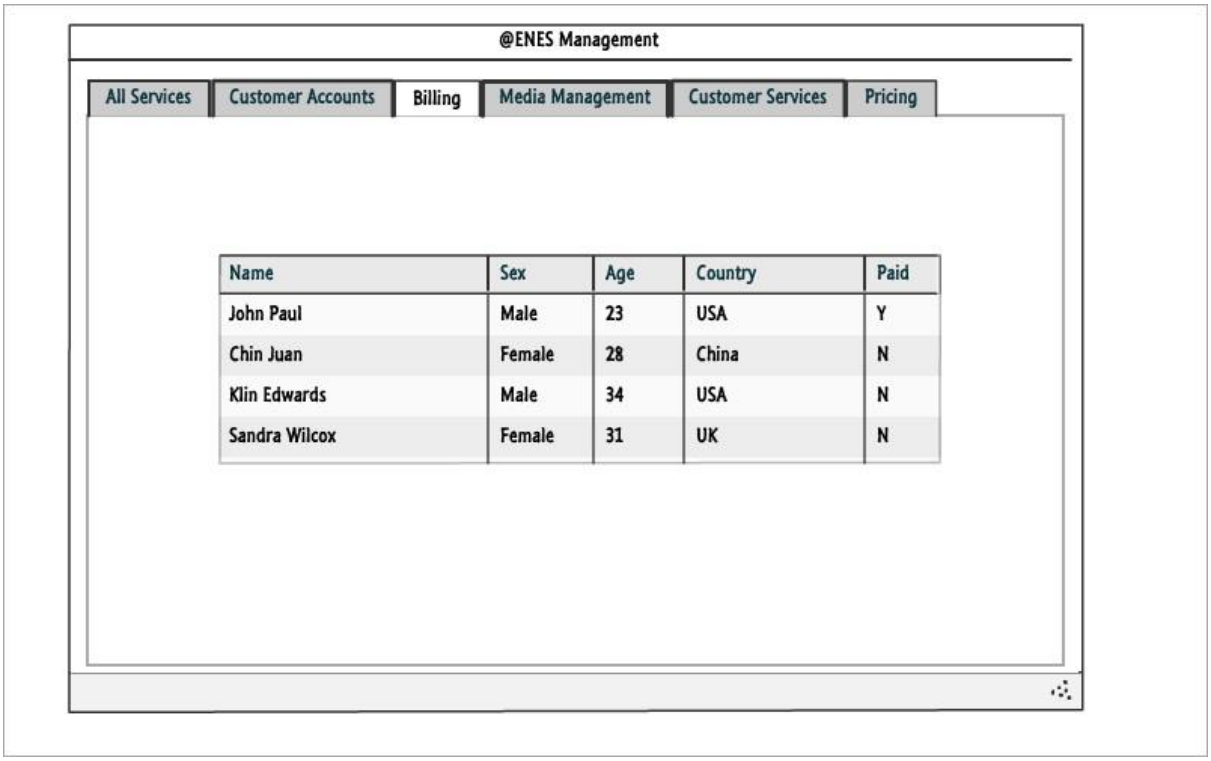
The Catalogue View allows users to search or browse media from the library



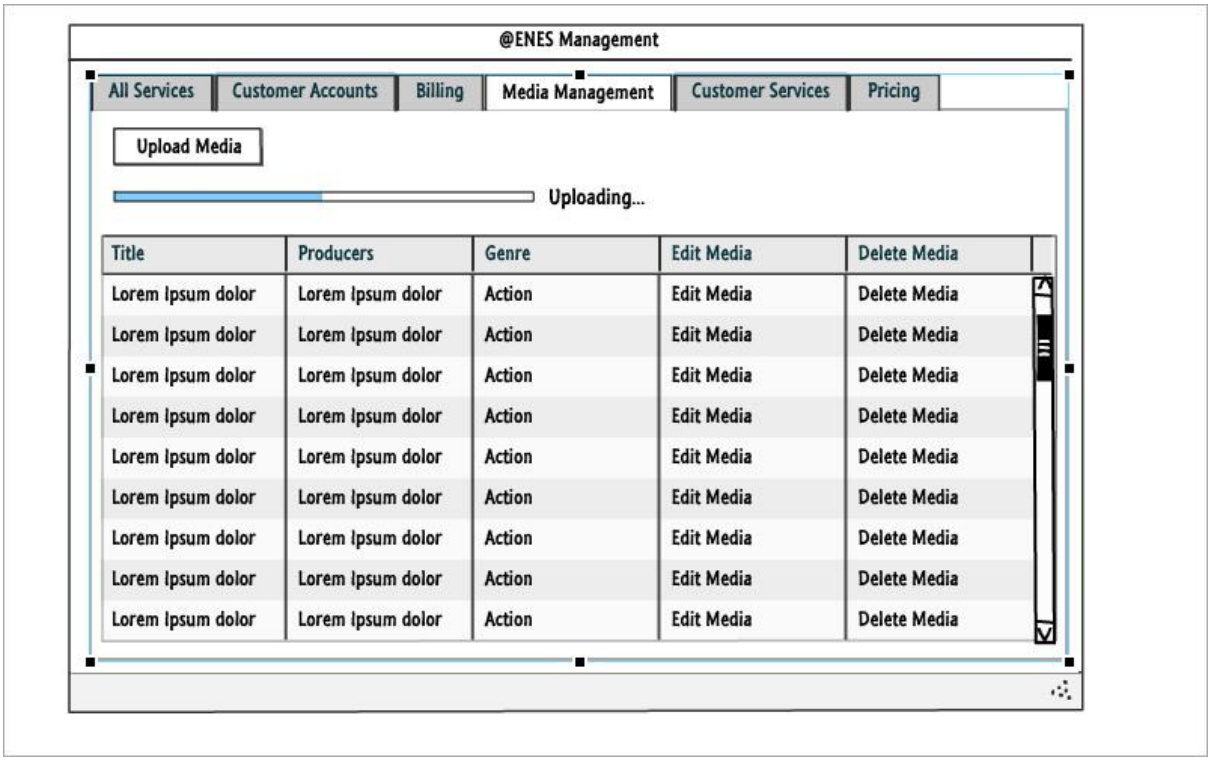
The @ENES Management application has an All Services tab to control the service overall



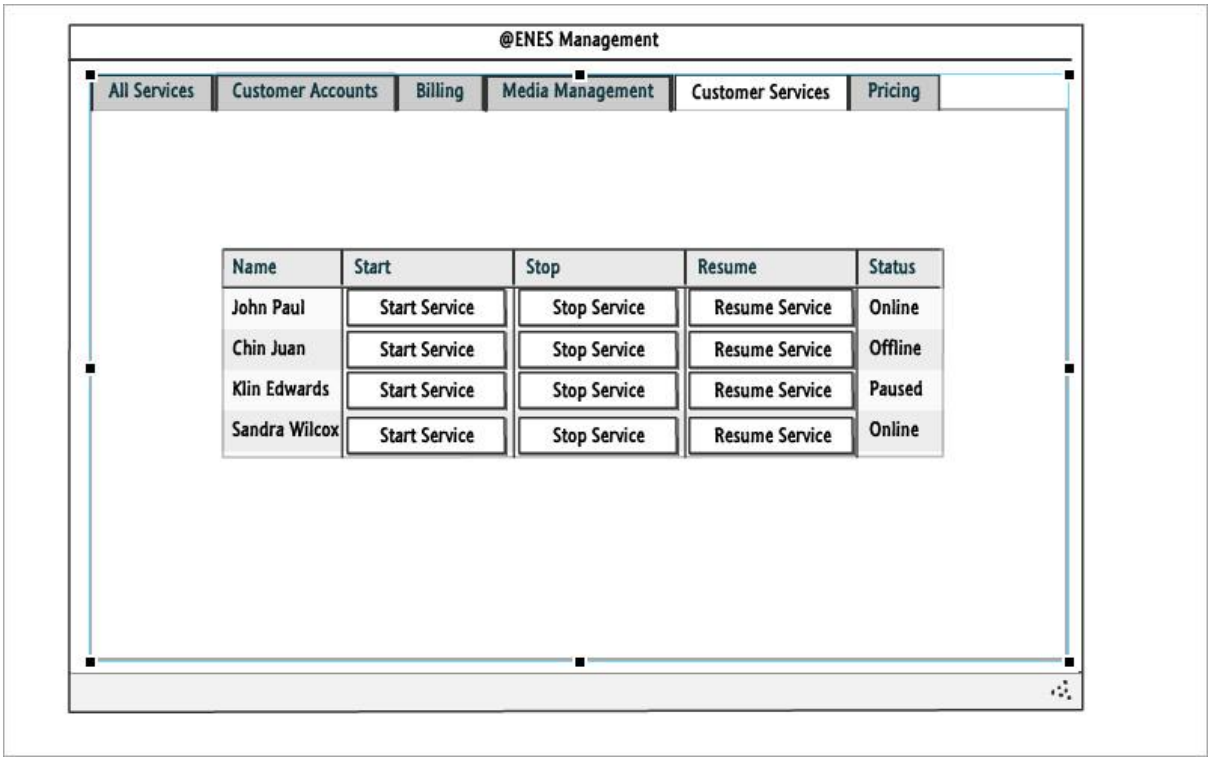
The Customer Accounts tab allows staff to individually select customers for details and administration



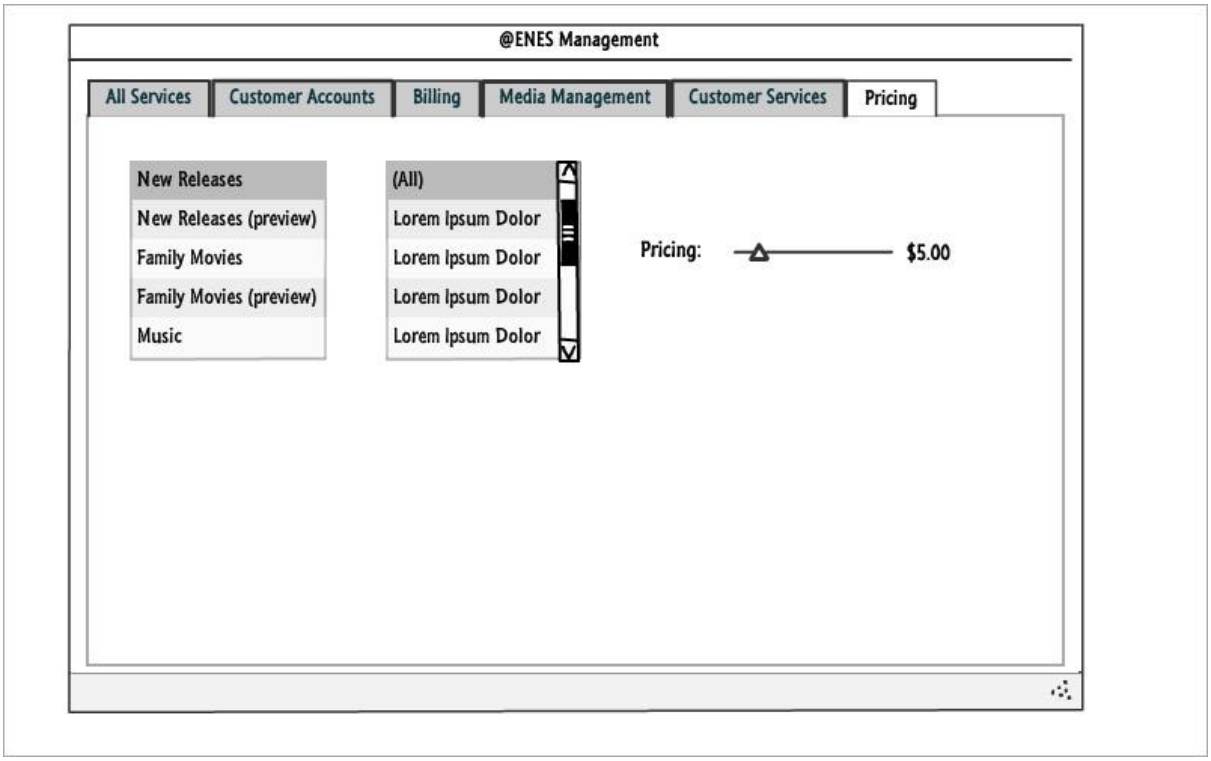
The Billing tab shows staff which customers have outstanding bills



The Media Management tab allows staff to browse and administrate existing media entries and upload new ones



The Customer Services tab allows staff to manipulate individual customers' services



The Pricing tab allows staff to change pricing tariffs. It also allows services administrators to change the pricing function.

Iterations

Milestone	Artifact	Prepared by	Completion Dates	
			Planned	Actual
Iteration 1 (Inception)			14/10/11	17/10/11
	Initial Project Plans	All	5/10/11	5/10/11
	Initial Iterations Plan	All	5/10/11	5/10/11
	Requirements Document	All	6/10/11	6/10/11
	Vision Document	Michael	12/10/11	12/10/11
	Problem Statement	Michael	12/10/11	12/10/11
	Position Statement	Michael	12/10/11	12/10/11
	Initial Use Case Models	All	13/10/11	13/10/11
	Risks List	Michael	14/10/11	17/10/11
Iteration 2 (Elaboration)			27/10/11	27/10/11
	Updated Project Plan	All	17/10/11	17/10/11
	Updated Requirements	Ian	17/10/11	17/10/11
	Updated Use Case Models	Tiaki	17/10/11	17/10/11
	Use Case Scenarios	Tiaki	19/10/11	24/10/11
	Architecture Document	Ian	19/10/11	20/10/11
	Persistent Data Diagrams (Schemas)	Ian/Michael	24/10/11	24/10/11
	User Interface Design	Ian	24/10/11	24/10/11
	Class Diagrams	All/Glenn	20/10/11	26/10/11
	Sequence Diagrams	Glenn	24/10/11	26/10/11
	State Diagrams	Glenn	26/10/11	27/10/11
	Construction Iteration Plan	Michael	26/10/11	26/10/11
Iteration 3 (Construction)				

Construction Iterations

The iterations to be used in the construction of this service follow the standard pattern of implementing functional requirements of progressively lower priority with each iteration, mainly that high level requirements will be implemented first. The priority for the iterations will also depend on the user they are for (i.e. the main users, the customers will have their functionality added first) and depending on what is needed before other functionality can be added (e.g. Access to the data will be added before any functionality using the data can be added).

Each iteration tends to develop the functionality for one (or in some cases more than one) of these sections the requirements were broken into, namely:

- Customer Interface/Usage
- Access and Storage
- Security
- Billing
- Pricing
- Rewards
- Staff Interface/Usage

The following summarises the work to be done in each construction iteration phase.

Iteration 1

This iteration will focus on adding:

- 'Access and Storage' functionality - so the system can store and use the data required for the other functionalities. It involves:
    - Creating the database entities
    - Creating the controllers to access those entities
    - Ensuring this data is saved and secure

This iteration is the most important as all other functionalities will require the use this data in some way, so will be building upon this iteration.

Iteration 2

This iteration focuses on adding the following functionality:

- High level requirements of 'Customer Interface/Usage' – namely the functionality for providing a web-based GUI application that allows customers to browse a catalogue for media and view it (and previews). This involves:
  - Constructing the web based interface
  - Adding appropriate views and the elements to those views
  - Creating the catalogue view
  - Adding search functionality to catalogue
  - Adding sort functionality to catalogue
  - Enabling media/previews to be transferred and played via the interface
- High level requirements of 'Staff Interface/Usage' – namely providing the functionality for the staff interface, allowing staff to add, edit and remove media recordings, customers and services. This involves:
  - Constructing the Java Application Interface
  - Adding functionality to add, edit and delete media recordings
  - Adding functionality to add, edit and delete customers
  - Adding functionality to start, stop and resume services.

This iteration will be the longest, as it is implementing most of the functionality of the system. The staff functionality is added in addition to the customer functionality, as it helps to manage the testing of the customer functionality (i.e. it allows the adding of media recordings so the catalogue can be tested).

Iteration 3

This iteration focuses on adding the following functionality:

- 'Pricing' – this is providing the functionality to change the prices of media in the system. This involves:
  - Adding the interface view to perform this action
  - Adding functionality to change the price of an individual record
  - Adding functionality to change the price of a group of records
  - Adding functionality to change the method these prices are calculated (i.e. the 'pricing function')
- 'Security' – the only high priority requirement left. It involves:
  - Providing a encrypted certificate relative to a customer and their smart-card
  - Adding functionality to interface with a smart-card to get this certificate and log a customer/staff member in

Iteration 4

This iteration will move to adding the functionality for the medium priority requirements of the system, namely:

- Medium level requirements of 'Customer Interface/Usage'- involves:
  - Creating the customer view to manage playlists
  - Adding functionality to create playlists
  - Adding functionality to schedule playback of media

Iteration 5

This iteration focuses on adding:

- 'Billing' functionality – namely the functionality to change the customer's billing details and view this information. This will involve:
  - Creating the customer view to change the billing information
  - Adding the functionality to change this information
- Functionality to the 'Staff Interface/Usage' – namely creating a staff view that shows all customer billing information

Iteration 6

The iteration moves down to low level priority requirements, namely those of:

- 'Rewards' functionality – this involves:
  - Adding to the system to ability to store reward points
  - Adding the functionality to manage these reward points (view them, spend them, accumulate them)
  - Add the option of having 'support partnerships' involved in this rewards system

Project Details

Group Work

The nature of this project meant the way the group worked together was quite different to how we worked for the second assignment. Much of the work we completed for this project we completed as a group rather than allocating tasks and working individually. This is especially true for most of the early work on the project. Often we would work together as a group during our group meeting or at other times to produce a rough version of one of the required documents to get a good start which all the group agreed upon. We would then allocate one or more group members to refine the document and eventually produce a final version. Later in the project there were more tasks which were appropriate to be allocated and worked on individually. These tasks were allocated as fairly as possible and according to group members strengths where possible.

Meetings

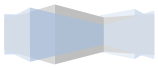
We held group meetings at least weekly for the duration of the project. Our First meeting was very informal and basically just involved all members agreeing to continue holding our group meetings on Wednesdays. From then on we held weekly formal meetings, along with an additional formal meeting in each of the last two weeks of the project.

The main focus of each of the formal meetings, apart from the first, were progress reports from each group member and the allocation of tasks to be completed before the next meeting. During the early meetings we also spent a large amount of time working together on initial project documentation. For full meeting minutes along with samples from group members work diaries see the Appendices.

Role Allocation

The following table show the role allocation that we used for this project. Due to the nature of the project all roles with the exception of project manager and system architect were shared amongst all group members.

	Glenn	Ian	Michael	Tiaki
Project manager			X	
Systems Architect		X		
Requirements Engineer	X	X	X	X
Analysts	X	X	X	X
Designers	X	X	X	X
Documentation	X	X	X	X

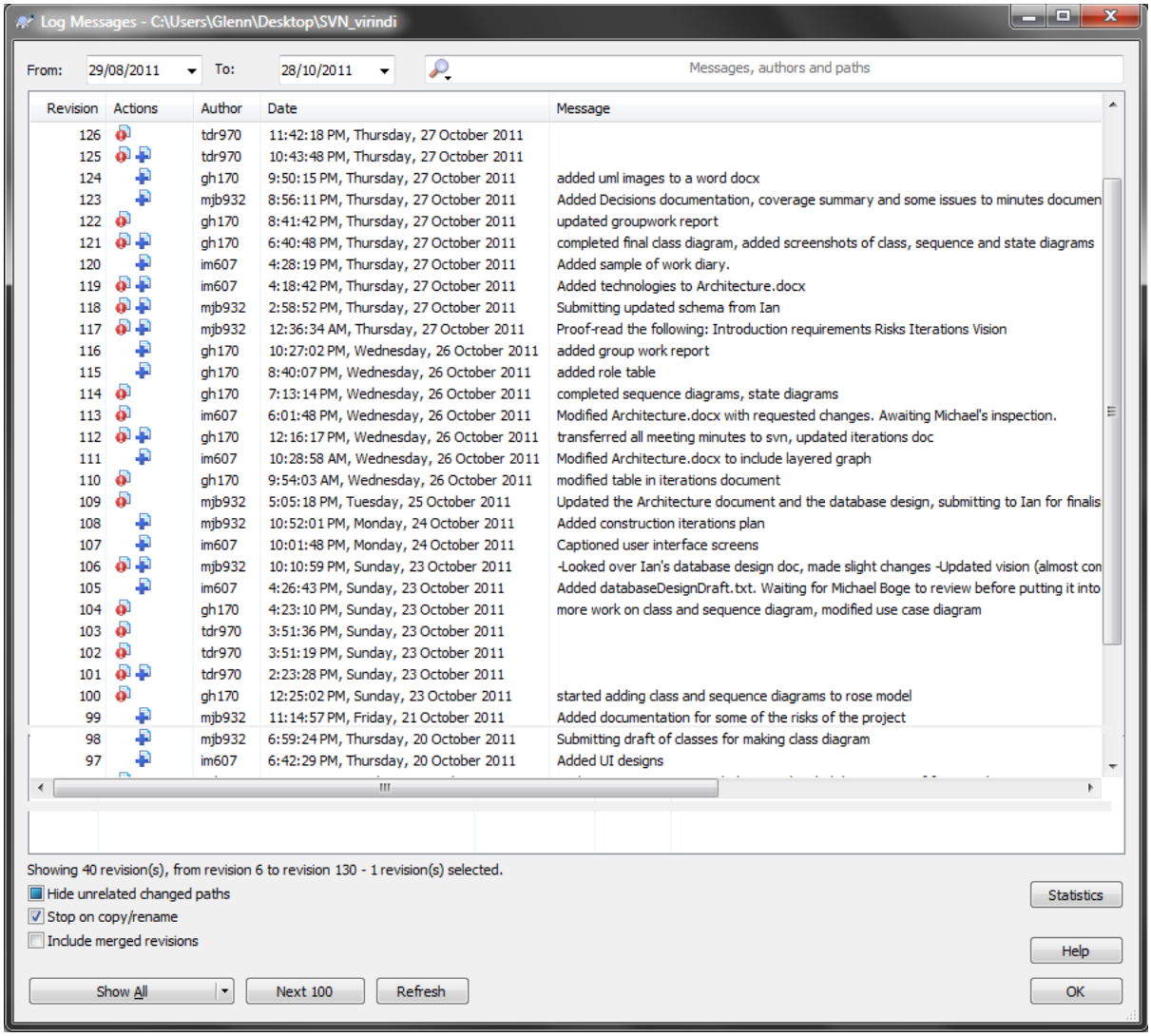


Please see the Iteration Plan document for the artefacts each member delivered during the project.

Version Control

We used Subversion (primarily through TortoiseSVN) through the duration of the project for our version control. This meant that we could make changes to the numerous documents we prepared during the project without the risk of permanently losing or disturbing any of our previous. It also meant that we could very our work with other members of the group quickly and easily no matter where we all were.

The following shows the log file from our SVN account during the elaboration phase of the project.



Specification Coverage

Our team believes that our solution covers all of the information in the specification. This is because our system was designed from the outset using this specification. During the inception phase we developed our required functionalities from these specifications, which we further expanded in the elaboration phase. The report sections above describe the solution and show how it fulfils all the functionalities that we found in the specification (see the Requirements section of this report details on these requirements).

Decisions

The project design was modified a number of times during the inception and elaboration phases discussed in this report. This was because of issues found issues found in the initial ideas or because of better ideas being thought of. These decisions and the changes to them occurred in the following sections of the project design:

Architecture

The decision to use a server-client based system as the main system architecture of the project was maintained throughout the entire project.

On the other hand, our software architecture was changed. Initially we used Model View Controller architecture because we felt that it best followed the BCE design pattern for the project.

Data Persistence (database)

Initially the database consisted of a few simple entities, mainly to hold the data for the media, customer, staff and system administrator, as well as entities to provide the functionality, namely the playlist, media library, billing and service entities.

After constructing the first schema the team decided to eliminate the system administrator and services entities, and to replace them with an attribute in the 'staff' entity to specify if they were regular staff or an administrator, and an attribute in customer showing their service status. This was done to eliminate the need to construct special controllers to access these entities.

Much later in the project, the problem was identified that we had no way of linking the playlist and media library to multiple media records, the decision was then made to fix this by adding the linking entities 'libraryLink' and 'playlistLink', which essentially act as a bridging entity to allow many to many relationships in the planned database.

Classes

Throughout the entire project our classes were designed to keep the BCE relationship in mind, and our classes reflect this as they always matched this pattern, throughout all design decisions and changes.

Initially our classes were relatively simplified, consisting of the boundary classes CustomerInterface and StaffInterface, which essentially wrapped all actions their respective user could perform. There were the controller classes which provided the functions to perform the specific action (such as billingController, which allowed billing details to be set and bills to be viewed) and entity classes representing each entity in the database (to provide a way to access/store this data).

The most major design change occurred here when we decided to separate our boundary interface classes into a number of view classes (i.e. classes that represented each view in the system the users could see and the actions they could perform there). This decision was made because it was felt that the separated classes would be simpler to construct and test, as well as helping us to link them with their appropriate controllers.

The controller classes themselves remained very similar through the design phases, as they had been made initially to encompass all actions to be performed, and had been broken up suitably.

The entity classes were changed as the database schema was modified (see data persistence for details).

Users

Initially we defined the system to have 3 users: customer, staff and system administrator, these being the key users listed by the specification. We considered the system administrator to be a separate user, because it was never mentioned that they could perform the staff's tasks, and it was specified that only they could edit the pricing function.

Later on in the project, we decided to keep the system administrator as a specific user, but made them essentially an extension on a staff member user. This decision was made because of the schema, it was decided that adding a specific entity for a service administrator was not a good option, so to instead include an attribute in the 'staff' entity to specify if they were regular staff or an administrator. This decisions was also made so that the system administrator could use the staff interface without needing a special access method (as they are contained in the staff table of the database).

Use Cases

The use cases for the system remained relatively consistent throughout all phases, as they had again been designed using the functional requirements directly, meaning no major additions/removals were needed as these cases covered all possible actions within the system.

The only notable changes needed for the use cases were that it was decided to integrate the giving and using of rewards points into the catalogue process, so that when a piece of media points are awarded as appropriate and an option is given to pay using reward points.

Interfaces

Our interfaces were designed early into the project, as we felt they were a major part of the system specified in the non-functional requirements (since these interfaces would determine what could be done with the system and how). Because of this decision the interface designs remained similar throughout the whole project.

The main decisions made here initially were, to enable easy navigation throughout these interfaces to perform different actions (provided by tabs at the top of each interface to get to an actions respective view) and to make simple to use by providing common elements found in similar systems (such as providing scrollable lists to easily browse through the catalogue).

All of these decisions and changes are reflected in each sections respective documentation/diagrams.

Appendices

Appendix 1: Meeting Minutes

Week: 10
Date: 4/10
Members Present: All

Informal Meeting: Discussed possible member roles and the planned activities for tomorrow's meeting. Plan to create an initial plan for the project. All members should have read through the spec.

Week: 10
Date: 5/10
Members Present: All

Apologies:
Progress reports:

- All members have read the spec

Main business for this meeting:

- Allocated roles:
  - Michael – leader
  - Ian – system architect
  - All - Requirements Engineer, Analysts, Designers, Documentation
- Devised our initial project and iteration plans. At this stage we plan to have the inception phase completed by 14/10 (end of wk 11)
- Created a list of documents to be prepared and allocated the early one to members
- Discussed the content of the vision, problem and position statements

Action items to be completed by next meeting:

- The group will work on the requirements list together in lab time tomorrow
- Micheal will prepare vision, problem and position statements based on the groups discussion
- Glenn will prepare meeting minutes
- Ian will type up requirements list and will group requirements
- Tiaki will ensure that our list of required documents is adequate

Week: 11
Date: 12/10
Members Present: All

Apologies:
Progress reports:

- Michael – has completed the vision, problem and position statements. Wants all members to look over.
- Ian – has typed up the list of requirements devised in lab last week and grouped them into appropriate categories
- Tiaki – has added to the list of documents, thinks there will be more to add later.
- Glenn – has prepared last week's meeting minutes.

Main business for this meeting:

- Discussed possible use cases
- Also discussed possible classes
- Decided we will have a meeting on Monday 17
- Checked iteration plan, all documents required for inception phase should be done by Monday. (only slightly behind schedule)

Action items to be completed by next meeting:

- We will all work on initial use case diagram tomorrow after lab task
- Michael will prepare a list of risks for the project
- Ian will update the requirements list
- Tiaki will refine the use case diagram after group work tomorrow (Glenn will help)
- Glenn will continue with minutes.

Week: 12
Date: 17/10
Members Present: All

Apologies:
Progress reports:

- Michael has prepared list of risks
- Ian has updated the list of requirements, adding a number of requirements that we left out
- Tiaki and Glenn have added a number of use cases to the use case diagram

Main business for this meeting:

- Discussed issues found with initial use cases, with some cases being forgotten or irrelevant, corrected these issues
- Inception phase now complete
- Updated the project and iteration plans
- Prepared a rough class diagram
- Will have next meeting in lab time on Thursday

Action items to be completed by next meeting:

- Glenn will refine class diagram prepare on rational rose
- Ian will prepare the architecture document
- Tiaki will do use case scenarios
- Michael will start on report

Week: 12
Date: 20/10
Members Present: All

Apologies:
Progress reports:

- Glenn has started the class diagram on rational rose, has not got all the attributes and operations but will complete while preparing sequence diagrams
- Ian has prepared the architecture document
- Tiaki has started on the use case scenarios, not finished yet but will keep working on them
- Michael has started on the final report, got introductory stuff done

Main business for this meeting:

- Some problems with previous class designs, needed to remove service and system admin entities, finished fixing this issue before Glenn continued
- Everything else is going ok, just need to keep working hard
- Will have another meeting on Monday 24th

Action items to be completed by next meeting:

- Michael will keep working on various parts of the final report
- Tiaki will keep working on use case scenarios
- Glenn will work on the sequence diagrams and completing the class diagram
- Ian will create a database schema and user interface design (Michael will help)

Week: 13
Date: 24/10
Members Present: All

Apologies:
Progress reports:

- Glenn has most of the sequence diagrams done, has made some changes to the class diagram due to some issues that have come up while doing sequence diagrams
- Tiaki is finished the use case scenarios
- Ian and Michael have prepared database and user interface designs
- The final report is going ok

Main business for this meeting:

- Discussed what need to be done to finish the project
- Discussed state diagrams and worked out which classes will require state diagrams
- Will have final meeting on Wednesday

Action items to be completed by next meeting:

- Glenn will complete the sequence diagrams and try to complete the state diagrams
- Michael will prepare a Construction iteration plan for the report
- Michael, Ian and Tiaki will all work on the report

Week: 13
Date: 26/10
Members Present: All

Apologies:
Progress reports:

- Glenn is finished sequence diagrams
- Michael has finished the construction plan
- Final report is going well

Main business for this meeting:

- More discussion about state diagrams
- Highlighted the remaining parts of the report that need to be done

Action items to be completed by next meeting:

- Glenn will do the state diagrams (Michael will help)
- Everyone will keep doing things for the final report

Appendix 2: Work Diary Samples

Tiaki's Work Diary

14/10

- 2hr – Group meeting during 222 lab - I was assigned the task of constructing the use case scenarios and also to looking over the current use cases.

16/ 10

- 2hr – Created the use case scenarios file and constructed templates for all of the current use cases.
- 1hr - Discovered problems with the use cases, (some in the wrong places being attached the the wrong cases, some missing) will bring up in next group meeting.

17/10

- 2hr – Group meeting during 222 lab - I was told to continue with the use cases. I brought up the problems discovered previously and discussed methods to solve them.
- 1hr - Made modifications to use case diagram to suit the fixes discussed.

18/10

- 2hr - Made modifications to use case diagram (sorted out our extends and includes) for most cases. Brought up some more design issues.

19/10

- 1hr - Made some more changes to use case scenarios to match changes made to the use case diagram.
- Use case diagram and scenarios are now complete, will start new task after next meeting.

Ian Mckay's Work Diary

Monday 17/10/2011:

- Had group meeting at 12PM (1 hour)
    - Discussed requirements generated last week.
    - Told to prepare the architecture document.

Tuesday 18/10/2011:

- Wrote up preliminary architecture evaluation for distributed systems and submitted to SVN to check for Michael's approval (2 hours)

Wednesday 19/10/2011:

- Talked to Michael, feedback on distributed systems section was good. Also discussed the other sections of the architecture document yet to be added (software architecture and technologies). (1 hour)
- Added sections discussed with Michael and again submitted for feedback (2 hours)

Thursday 20/10/2011:

- Starting on UI design, began constructing preliminary user interface design images. Committed these to SVN. (2 hours)
- Had group meeting in lab (2 hours)
    - Discovered issue with using controller method architecture so switched to layered architecture.
    - Fixed the documentation for this and constructed layered diagram. Architecture document now finished.
    - Told to continue working on UI design.

Friday 21/10/2011

- Finished adding UI design images for all views, committed to SVN (2 hours)

Saturday 22/10/2011

- Added images and captions to a UI design document, UI design section now complete (2 hours)