|  |  |
| --- | --- |
| Logo_FPT_University_doc | **MINISTRY OF EDUCATION AND TRAINING** |

**FPT UNIVERSITY**

**CAPSTONE PROJECT DOCUMENT**

**Bookstore Management System**

|  |  |
| --- | --- |
|  | |
| Group Member | SE03975 - Olasehinde Ezekiel Olaoluwa |
| SE03444 - Kanda Joshua David |
| SE03520 - Shunom Peter Bala |
| Supervisor | Bui Đinh Chien |
| Ext. Supervisor |  |
| Project Code | BSMS |

- Hanoi, 05/2017 -

[A. INTRODUCTION 4](#_Toc488611826)

[1. INTRODUCTION 4](#_Toc488611827)

[1.1. Purpose 4](#_Toc488611828)

[1.2. Acronyms and Definitions 4](#_Toc488611829)

[2. ABSTRACT 5](#_Toc488611830)

[3. LITERATURE REVIEW 5](#_Toc488611831)

[4. PROPOSAL 6](#_Toc488611832)

[4.1. The idea 6](#_Toc488611833)

[4.2. Objectives 6](#_Toc488611834)

[4.3. Brief description about system 7](#_Toc488611835)

[4.4. System features 7](#_Toc488611836)

[5. BENEFIT FROM PROJECT 8](#_Toc488611837)

[5.1. For Organization 8](#_Toc488611838)

[5.2. For our group 8](#_Toc488611839)

[6. POTENTIAL RISKS 8](#_Toc488611840)

[6.1. Human Resource Risk 8](#_Toc488611841)

[6.2. Time Risk 9](#_Toc488611842)

[6.3. Communication Risk 9](#_Toc488611843)

[6.4. Technical Risk 9](#_Toc488611844)

[7. REFERENCES 9](#_Toc488611845)

[B. Project Management Plan 10](#_Toc488611846)

[1. PROBLEM DEFINITION 10](#_Toc488611847)

[1.1. Name of this Project 10](#_Toc488611848)

[2. PROJECT OVERVIEW 10](#_Toc488611849)

[2.1. Project Description 10](#_Toc488611850)

[2.2. Scope 11](#_Toc488611851)

[2.3. Standard Objectives 13](#_Toc488611852)

[2.4. Milestone and Deliverables 14](#_Toc488611853)

[3. PROJECT ORGANIZATION 14](#_Toc488611854)

[3.1. Software Process Model 14](#_Toc488611855)

[3.2. Project lifecycle 15](#_Toc488611856)

[3.3. Roles and Responsibilities 17](#_Toc488611857)

[4. TOOLS AND INFRASTRUCTURES 17](#_Toc488611858)

[4.1. Hardware 17](#_Toc488611859)

[4.2. Software 18](#_Toc488611860)

[5. SCHEDULE 18](#_Toc488611861)

[5.1. Detailed Schedule 18](#_Toc488611862)

[5.2. Meeting Schedule 20](#_Toc488611863)

[6. RISK MANAGEMENT 21](#_Toc488611864)

[7. CODING CONVENTION 22](#_Toc488611865)

[C. Software Requirement Specification 23](#_Toc488611866)

[1. USER REQUIREMENTS SPECIFICATION 23](#_Toc488611867)

[1.1 Introduction 23](#_Toc488611868)

[1.2 Overall Description 24](#_Toc488611869)

[2 SOFTWARE REQUIREMENTS SPECIFICATION 28](#_Toc488611870)

[2.1 System Purpose 28](#_Toc488611871)

[2.2 Functional Specification 30](#_Toc488611872)

[2.3 Non-functional Specification 87](#_Toc488611873)

[D. Software Architecture and Design 89](#_Toc488611874)

[1 PURPOSE 89](#_Toc488611875)

[2 ARCHITECTURE DESIGN 89](#_Toc488611876)

[2.2 Choice of Architecture Design 89](#_Toc488611877)

[2.3 Component Diagram 91](#_Toc488611878)

[3 DETAILED DESIGN 92](#_Toc488611879)

[3.2 Common Design 92](#_Toc488611880)

[3.3 Authentication Management 92](#_Toc488611881)

[3.4 Category Management 110](#_Toc488611882)

[3.5 Genre Management 120](#_Toc488611883)

[3.6 Staff Management 129](#_Toc488611884)

[3.7 Book Management 144](#_Toc488611885)

[3.8 Wishlist Management 178](#_Toc488611886)

[3.9 Author Management 185](#_Toc488611887)

[3.10 Profile Management 198](#_Toc488611888)

[3.11 System Search 207](#_Toc488611889)

[4 Web Service API 222](#_Toc488611890)

[4.2 Web Service and Method 222](#_Toc488611891)

[4.3 Web Method Description 224](#_Toc488611892)

[5 DATABASE DESIGN 266](#_Toc488611893)

[5.2 Entity Relationship Diagram 266](#_Toc488611894)

[5.3 Database Diagram 267](#_Toc488611895)

[5.4 Table Descriptions 268](#_Toc488611896)

# Introduction

# INTRODUCTION

## Purpose

The main aim of writing this document is to provide reviewers with insight and adequate information with regards to BSMS. Reviewers will fully gain insight of what our overall system is all about, including some detailed information about our system. This document contain brief overview of what our system is all about

## Acronyms and Definitions

|  |  |  |
| --- | --- | --- |
| Acronym & Abbreviation | Definition | Note |
| UI | User Interface |  |
| GUI | Graphical User Interface |  |
| BSMS | Bookstore Management System |  |
| **CRUD** | Create, Read, Update, Delete |  |
|  |  |  |
|  |  |  |
|  |  |  |

Table 1-1: Definitions and Acronyms

# ABSTRACT

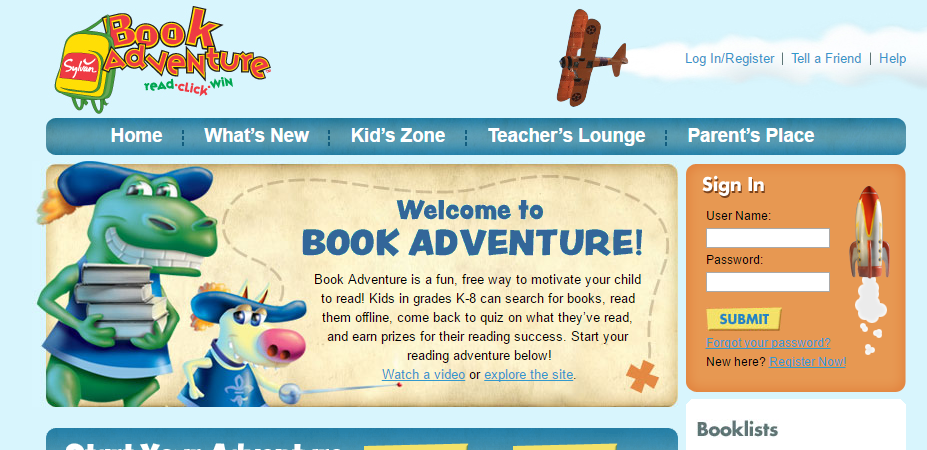
I have always assumed a system that could really help us managing information, so I thought many bookstores that manages their information using the manual book entry records,

# LITERATURE REVIEW

Existing system which have previously implement BSMS features have really done a lot of work and in our own implementation we will add more additional features, and dispose some of the feature that are less necessary in version 1.0.

During our team research we found this particular asp.net web-form website, that seems to imitate bookstore management system, <http://www.bookadventure.com/Home.aspx>, and this website is lacking a lot functionality and it is not dedicated to a particular bookstore, instead it just focus on books and kids.

We have come up with ideas to make this web application better than what it’s currently seems to be.



# PROPOSAL

## The idea

The idea of this project all comes from management problem faced by many managers and staff, we have decided to solve that problem as we build a content/information management system with full focus on Books so we’ve named our application Bookstore Management System (BSMS), with our BSMS web application work will became more easer for both Managers, Staffs and even Customers, whereas customer can keep track of books that the bookstore have and those they don’t have, bookstore can also provide some free books which are readable online.

## Objectives

The goal of building this system is to convince bookstores and customers to use the BSMS, the objectives include:

|  |  |  |
| --- | --- | --- |
| User | | What to do |
| Customer | | Read Book Online / Download |
| Receive Email Notifications |
| Wish list Management |
| Search |
|  | | |
| Staff | Book Management | |
| Producer Management | |
| Author Management | |
| Authentication Management | |
| ` | | |
| Admin | Oversees the Overall System | |

## Brief description about system

Bookstore Management System has been one of the trending type of web based applications this days. This is a complete customize version of the solution that have been use by many bookstore up till date, dissecting our application you will notice that Staff users can Add book, Update Book Information, as well as Delete, same functionality is allow of the staff for Author, Category and Many more, whereas the Admin of the system oversees the overall system and customer is allow to keep track of their favorite books.

## System features

The feature of our system includes the following:

* System Management
* Search
* Advance Search
* Authentication Management
* Login
* Logout
* Forget Password
* Register Customer / Staff
* Staff Management
* Create, Read, Update, Delete (CRUD)
* Category Management
* Create, Read, Update, Delete (CRUD)
* Book Management
* CRUD
* Wish List Management
* Add to Wish List
* Remove From Wish List
* Profile Management
* Change Password
* Edit Profile
* Author Management
* CRUD
* View Books By Author
* Producer Management
* CRUD
* View Book By Producer

# BENEFIT FROM PROJECT

## For Organization

Organizations like Bookstore companies will have a lot to gain from our system because we provide them with a management system that can help them reduce staff working in their organization thus reducing man power as they can keep track of their books, and also provides customer with information.

## For our group

This is a chance for our group to increase our knowledge; there are a lot of things we could learn from this project. Brainstorming is number one thing you we will surely learn from this, including Software Architecture, Coding Convention, REST web services, MVC design pattern, Unit testing and many more.

# POTENTIAL RISKS

Risk management is an important factor that hiders success of any project, being aware that at-least one the listed risk below might occur, our team is fully ready to tackle such events

## Human Resource Risk

Disciplining team members have always serves as one of the biggest problems; team member failure to show up at the meeting at the scheduled time is unavoidable as well as missing deadline which may incur project delay and in the worst case scenario total project failure.

## Time Risk

Team member could have possibly underestimate the project, which 3 month might be too small for 3 people, in-which defining the SRS document might be larger than the team ability as more man power might be required.

## Communication Risk

Communication risk is one of the most common hindrances that could occur in project development phase. All our team member lives in different location, long distance to be precise (as long as 30km). With this communication might be done through Email or any social network around like Facebook or Whatsapp.

## Technical Risk

Team member might not be good with some of the model we are using; a simple example might be Technology. Team member might not be familiar with technology we will be using. They might not be good with Project Management, or some required thing we need to do throughout the project.

# REFERENCES

[1] <http://www.ds.libol.fpt.edu.vn>

[2] <http://www.searchsoftwarequality.techtarget.com/definition/unit-testing>

[3] <http://www.softwaretestingfundamentals.com/system-testing>

[4] <http://www.docs.microsoft.com/dotnet>

# Project Management Plan

# PROBLEM DEFINITION

## Name of this Project

The major aim of this project is to build a (BSMS) bookstore management system in-order to make management of book more easier by bringing it to the digital world, Book maintenances, information management and keeping track of data will be make much more easy with this concept.

# PROJECT OVERVIEW

## Project Description

Bookstore management system (BSMS) is one of a kind which is widely known for its popularity, this system is mostly used in any advance bookstore today. BSMS can help manage staff of a bookstore including books and it provides full access to end users whom we call the Customers, they can keep track of books, and even find book they are currently in production. Information of books like Author, Producers, Synopsis and many more will before more easily for customer to find. Also staff of our system can access content/Information management as well as the system admin.

## Scope

The scope of our project will strictly follow the following stages:

Developing:

* Software Requirement Specification
* Software Architecture and detailed design
* Implementation + Unit Testing
* Test case and Combination Test

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Function** | **Sub-Function** | **Description** |
| **1** | Authentication Management | Login | Logging into the system |
| Logout | Signing out of BSMS |
| Register Customer | Guest registration to be become a customer |
| Register Staff | Registration of Staff users |
| Forget Password | Process of recovering lost password or reset |
| **2** | Category Management | Create Category | Creating new category in our system |
| Edit Category | Editing/Updating existing category |
| View Category | List all existing categories |
| Delete Category | Remove existing category from the system |
| View Books | View all book under a particular category |
| **3** | Genre Management | Create Genre | Creating new genre in our system |
| Edit Genre | Editing/Updating existing genre |
| View Genre | List all existing genre |
| Delete Genre | Remove existing genre from the system |
| View Genre | View all book under a particular genre |
| **4** | Staff Management | Create Staff | Create new staff |
| View Staff | View all staffs |
| Update Staff | Update staff information |
| Delete Staff | Delete staff from the system |
| **5** | Book Management | Add Book | Adding new book to the system |
| Edit Book | Edit existing books information |
| View Book | N/A |
| Delete Book | Delete book completely from the system |
| Read Online/Download | Read free books online or download the softcopy file |
| **6** | Wish list Management | Add to wish-list | Adding new books to wish-list directory |
| Remove from wish-list | Removing books from wish list directory |
| **7** | Profile Management | Edit Profile | User can use this feature to edit his/her own private information on the system |
| Change Password | Changing of password |
| **8** | Author management | Add Author | Add new author information |
| View Author | View author information |
| Update Author | Edit author information |
| Delete Author | Completely remove author from the system |
| View Book | View book by a particular author |
| **9** | Producer Management | Add Producer | Add a new producer |
| View Producer | Check producer information/profile |
| Update Producer | Edit producer information |
| Delete Producer | Remove producer information from the system |
| View Book | View book produced by a particular producer |
| **10** | System Search | Simple Search | Search book by name or synopsis. |
| Advance Search | Search the system for information. Book, Author, Producer, …… etc. |

## Standard Objectives

|  |  |  |  |
| --- | --- | --- | --- |
| **Metrics** | **Unit** | **Committed** | **Actual** |
| **Start Date** | dd-mm-yyyy | 14-May-2017 | N/A |
| **End Date** | dd-mm-yyyy | 14-Aug -2017 | N/A |
| **Duration** | Elapsed days | 102 days | N/A |
| **Team Size** | Number | 4 | 3 |

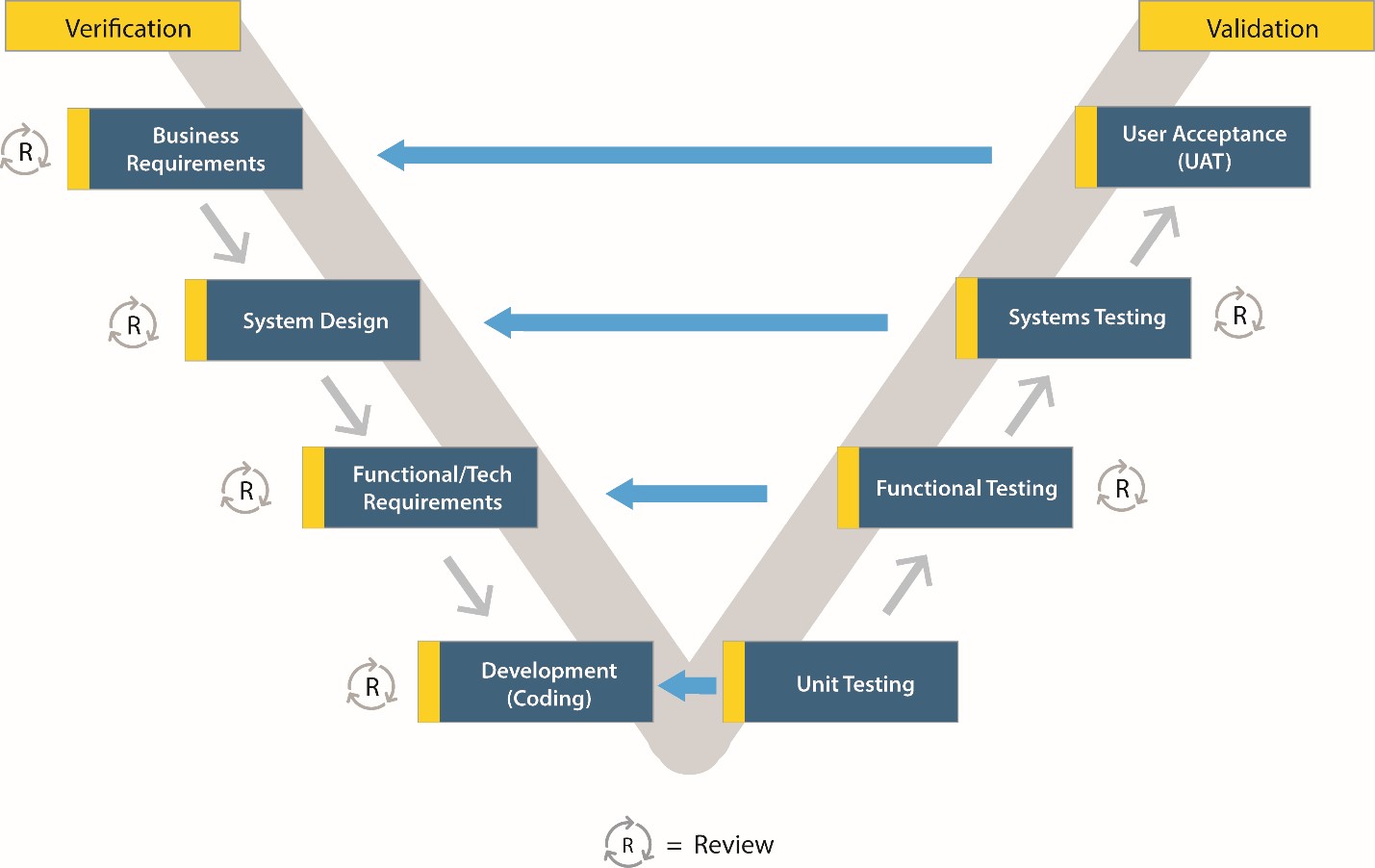
## Milestone and Deliverables

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***No*** | ***Deliverable*** | ***Committed*** | ***Actual*** | ***Deviation*** |
| **1** | Report #1 – Introduction | 21-May-2017 | 21-May-2017 | 0 days |
| **2** | Report #2 – Project Management Plan | 4-June-2017 | 4-June-2017 | 0 days |
| **3** | Report #3 – SRS | 10-June-2017 | 10-June-2017 | 0 days |
| **4** | Report #4 – Detailed Design V1.0 | 25-June-2017 | N/A | 0 days |
| **6** | Report #5 – Test Report V1.0 | N/A | N/A | 0 days |
| **7** | Report #6 – Manual | 16-Jul-2017 | N/A | 0 days |

# PROJECT ORGANIZATION

## Software Process Model

Our web application development phase will be implemented using the V software development process model. Our application will go through continuous updating following the V model format until final success of our project.



Why use the V-Model:

* Simple and easy to use.
* Testing activities like planning, test designing happens well before coding. This saves a lot of time. Hence higher chance of success over the waterfall model.
* Proactive defect tracking – that is defects are found at early stage.
* Avoids the downward flow of the defects.
* Works well for small projects where requirements are easily understood.

## Project lifecycle

|  |  |
| --- | --- |
| **1st Iteration** | |
| **Task** | **Description** |
| **Project Definition** | This happen as the beginning of everything, as all the team members collide to come with a project, and define what are the required features of our system. |
| **System Requirement Definition** | This module is where we will define in detail, the user requirement and how to implement them. |
| **Analysis and Design** |  |
| **Implementation and Manual** |  |
|  |  |

|  |  |
| --- | --- |
| **2nd Iteration** | |
| **Task** | **Description** |
| **System Requirement Definition** | This module is where we will define in detail, the user requirement and how to implement them. |
| **Analysis and Design** |  |
| **Implementation and Manual** |  |

## Roles and Responsibilities

|  |  |  |
| --- | --- | --- |
| **Team Member** | **Project Management** | |
| **Olasehinde Ezekiel Olaoluwa** | Role | Responsibility |
| Team Leader,  Developer  Tester  Database Engineer | Manages the team, and develop the system, Project manager of our system, The team leader goes will all function Testing, Development, Database Engineer, Design |
| **Shunmon Peter Bala** | Designer,  Developer | User interface designer, and support in implementation process |
| **Joshua David Kanda** | Database Engineer  Tester | Database Engineer, based on the database, he will have to manage anything relating to database engineering |

# TOOLS AND INFRASTRUCTURES

## Hardware

* Personal Computers for development with minimum configuration of :
* Core i3 4th generation 2.0 GHZ of Processor Speed
* 4GB of Ram
* At least 250 GB of hard disk space

## Software

* Operating System
* Windows 10
* Framework
* .Net Framework 4.0
* Angular JavaScript framework 4
* Front-End Framework Bootstrap 3/4
* Development Language
* JavaScript
* MSSQL
* C# (#=”Sharp”)
* Razor (CSHTML)

# SCHEDULE

## Detailed Schedule

|  |  |
| --- | --- |
| **N/O** | **Duration** |
| **Iteration 1** | 10 Weeks |
| **Iteration 2** | 4 Weeks |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Task** | **Description** | **Output** | **Deliverable** | **Resource needed** | **Dependency** | **Risk** |
| **SRS** | Defining system requirement specification | SRS document – Report 3 | 20-May-2017 | All Group Members | None | N/A |
| **Database Design** | Designing Database Tables and attributes | SQL Queries | 22-May-2017 | 1 Member | SRS | Database requirements tends to change during implementation |
| **Software Design Description** | N/A | Report #4  Software Architecture and Design document | 15-June-2017 | 3 Members | SRS and  Database Design |  |
| **UI Design** | Interface design for BSMS | Report #4 | 20-June-2017 | 2 Member | Report #4 | Design might change in SAD report 4 |
| **Implementation and Unit testing** | Coding section | Implementation according to SRS | 21-June-2017 | 2 Member | SRS | SRS document is subjected to future change |
| **System Testing** | Testing the system to be sure if its officially working correctly | Test Report #6 | 15-July-2017 | All Team Members | SRS | SRS document is subjected to future change |
| **Deployment** | Publishing of our web application to the internet | N/A | 17-July-2017 | 1-2 Member | Implementation |  |
| **User Manual** | Creating a manual for end-users to understand the application | User manual Document | 20-July-2017 | 1 member | Implementation |  |

## Meeting Schedule

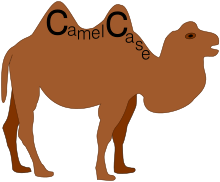
|  |  |  |  |
| --- | --- | --- | --- |
| **Subject** | **Capstone Project** | **Date** | **Every Saturday** |
| **Facilitator** | Fpt University | **Time** | 10:30 |
| **Location** | Fpt University campus high tech park, Hoa Lac | **Scribe** | Joshua David Kanda |
| **Attendees** | Supervisor | | |
| Olasehinde Ezekiel Olaoluwa | | |
| Shunom Peter Bala | | |
| Joshua David Kanda | | |
| **Absentee** | Olalekan Savage | | |

# RISK MANAGEMENT

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **NO** | **Risk Description** | **Probability** | **Effect** | **Strategy** |
| **People Risk** | | | | |
| **1** | Misunderstanding between team member can occur | High | Intermediate | Fix misunderstanding between team member to make the project flow continue |
| **2** | Team member have different schedule | Low | Basic | Get all team members schedule and find suitable time to assign task |
| **3** | Team members cannot meet task deadline | High | Advance | Push deadline forward and make plan that allow team member to work overtime inorder to finish before the next deadline |
| **Technical Risk** | | | | |
| **4** | Team members might not be versatile with the technology we are using in our system | High | Intermediate | Change plan to use technology that team members are familiar with. |
| **5** | Team member do not understand the scope of the problem or do not fully understands the Requirement | High | Serious | Explain project scope and requirement to team member in more detail |

# CODING CONVENTION

Coding convention in our web application will strictly follows the camel case throughout, from the web services to the backend to the unit testing will strictly follow this rule.



Camel case (stylized as camelCase or CamelCase; also known as camel caps or more formally as medial capitals) is the practice of writing compound words or phrases such that each word or abbreviation in the middle of the phrase begins with a capital letter, with no intervening spaces or punctuation. E.g. someone whose name is “Nguyen Huy” in camelCase we will write it like this “nguyenHuy”, this coding convention will be followed throughout all of our system.

# Software Requirement Specification

# USER REQUIREMENTS SPECIFICATION

## Introduction

### Purposes

The main aim of writing this requirement document is to provide reviewer with adequate information with regards to BSMS. Reviewers will get insight of the overall system and detailed information about our system.

This document includes full description of the product such as the Stack holders, functional requirement, non-functional requirement, UI design and some other system descriptor.

This record is the product prerequisite determination (SRS) of our BSMS system, which comprise of two main sections: The User Requirement and The System Requirement Specification.

### Definition and Acronyms

|  |  |
| --- | --- |
| Term | Description |
| UI | User Interface |
| GUI | Graphical UI |
| BSMS | Bookstore Management System |
| SRS | Software Requirement Specification |
| UC | Use-Case |
| N/A | Not Available |
| DB | Database |
| CRUD | Create, Read, Update and Delete |
| Admin | Administrator |
| E-U | End-Users |

**Table 3-1: Definition and Acronyms**

### Scope

This document explains all in detail the overall requirement of BSMS, how you interact with the system and how the system respond to it.

The document applies to many stages of developing the system:

* One, the product supervisor need to use this document to evaluate the End-Product.
* It is as well very important for the developers, testers, UI designer and E-U
* Developers need to understand user requirement and the requirement specification in order to build the application correctly.

* Testers as well need to understand the functionalities of the system in-order to be able to know which part of the system is not working correctly.

* Since UI designers need create a user interface for the system, and one of the utter-most things to consider when doing this is to understand the different type of users in the system thus they will be able to create interactive UI for each set of users.

## Overall Description

Figure 1.21 show overall use case diagram of BSMS, our system provide full information management feature including Add Books, Users Account, etc. as shown in the figure below each stake holders of the system and level of access (who can do what, what can a particular user type do?).



**Figure 1.21**

### Product Features

The list below offers a brief outline of our system features and functionalities. The list is comprised of two major sections, the System Core features and Additional feature of the system. The core feature are those part of our system that are required for it work correctly, while the additional feature of the system are for feature purposes and they will be implement over time.

|  |  |
| --- | --- |
| **Core Features** | |
| **Feature** | **Description** |
| **Authentication** | Our system authentication features include the Login, Register, Logout, Forget Password and Change Password section.  This allows Users to gain full access to the system. |
| **Category and Genre Management** | Category and Genre is one of the most important part our system, as it allows user to be able group books into difference preferences. Administrator Access of this Feature includes the Create, Update, Delete and Read (CRUD) |
| **Staff Management** | Staff management of BSMS is a level of access allow by the Admin alone, actions like Create, Update, Delete, View Profile.  Stakeholders like staff can also update their own profile and view their profile. |
| **Book Management** | One of the core feature of the system is the book management as the name implies, Staff is allowed to add books, delete books and update book information, including price of book, and softcopy of the book, while the Admin of system is require to validate all those information, then approve or reject the book.  Approved books can seen in search while rejected once remains hidden in the system, with only admin and staff being able to access them. Customer features include the search, read online or download feature if book is free, if not then the system will only provide information. |
| **Management** | The Staff is allowed to make use of this functionality. Which include creation of example (The Hobbit, Lord of the Ring), after creation he/she can then add book under this series either existing books or new once |
| **Watch List Management** | This is the core functionally of the Customer, in which only the customer is allow to use this feature. With ability to keep track of read list, including Delete and Add Watch list |
| **Profile Management** | This feature is available to all stockholders apart from the guest; they can edit their profile information. |
| **Author Management** | Before books are being create, author of such books are required to have exist in the system thus the Staff is required to Add books authors into the system, including their background, Update and delete should also be available for staffs |
| **Email Notification** | Email Notification should be provide, as users can choose to receive update on subscription, lovers of books to receive update new book arrivals. |
| **Producer Management** | Producers in our system stands as those who produced a particular book, action like create, update edit and delete is totally allow on this functionality, and only the staff of our system is allow to access this functionally. |
| **Search Functionality** | This serves as the biggest part of BSMS as both all user types are allowed to access this. Users can search product name, search anything relating to book, producer, ISBN, Synopsis and many others. |

# SOFTWARE REQUIREMENTS SPECIFICATION

## System Purpose

### User Characteristics

There are specifically 4 type of user that interact with system, looking at the system from inner approach there are mainly 3 type of users which is the Admin, The Staff and Customers, an additional one will be the guest. Each and every one of these users has their own different level of access and they all have their own different requirements.

|  |  |
| --- | --- |
| **User** | **Characteristics** |
| **Admin** | This user overseas the overall system, he’s main functionality is to approve new books and create new Staff. This is the most important part of BSMS, Category, Genre are all pushed by the admin. |
| **Staff** | Preceded by the Admin, staffs also have high level of access in the system as they are required perform CRUD functions on Books, |
| **Customer** | The customer in our system is only given limited access as they can either read books online or add book to their watch list. |
| **Guest** | The Guest has the lowest level of access as they can only view books. |

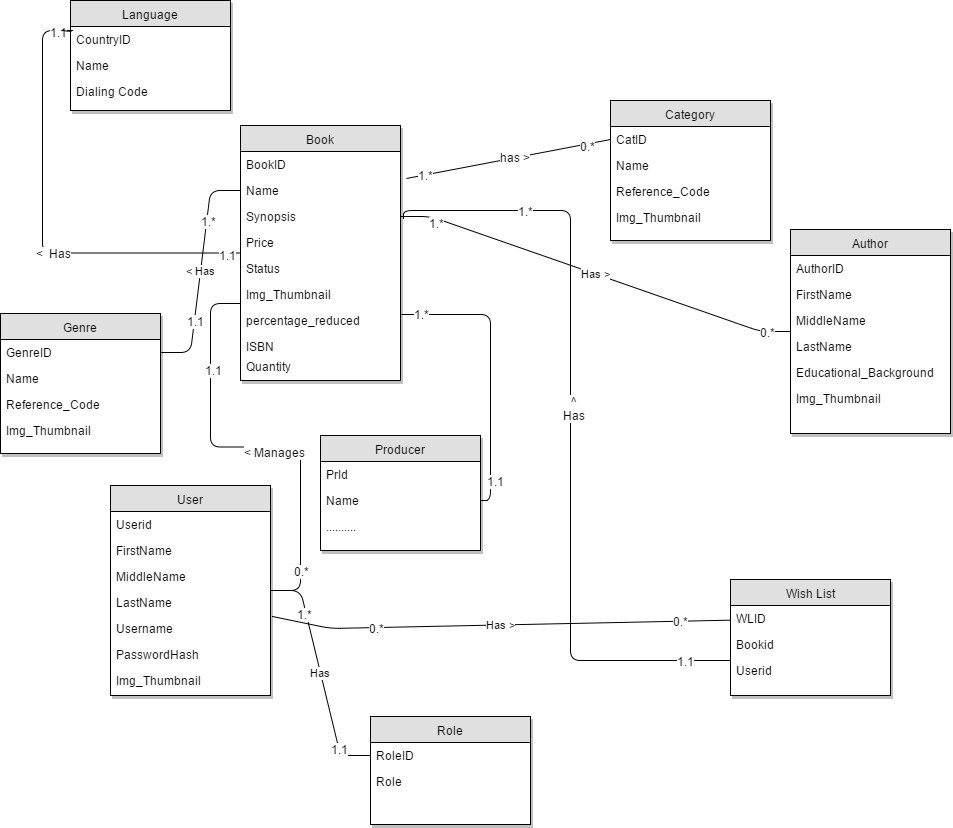
### User Documentation

Our target audiences in this case are bookstores and lovers of books who will like to keep track of new book in a particular bookstore and also it can serve as information management system to bookstores. For full user documentation on accessibility, kindly consult User's Manual Documents.

### Assumption

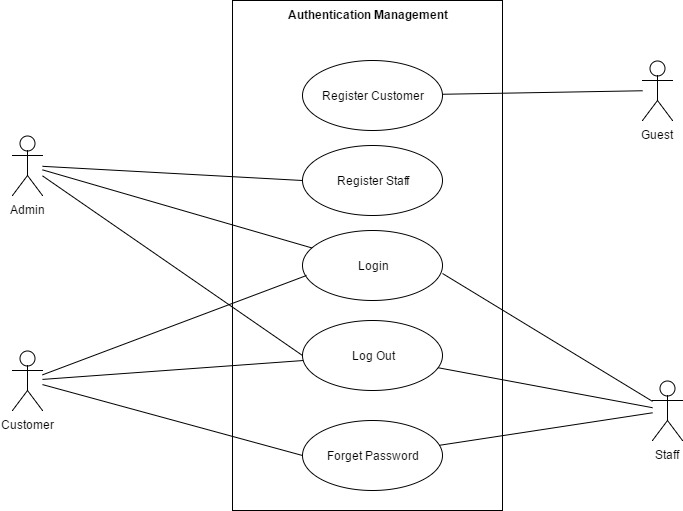
It is assumed that the system will work as planed on both mobile devices and pc’s as we will be using the mobile first approach in design using the bootstrap UI designer and Viewport.

### Entity-Relationship Diagram

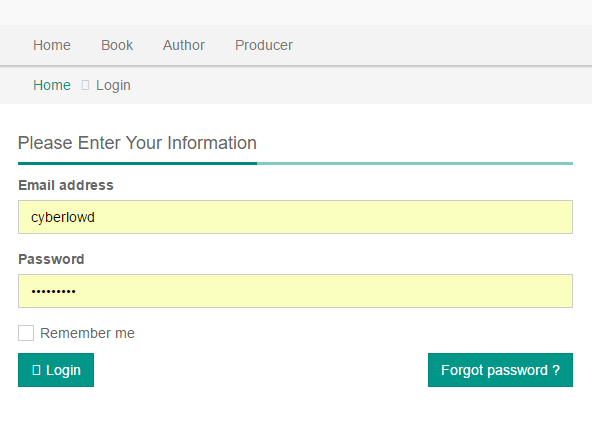


## Functional Specification

### Authentication Management



#### Login

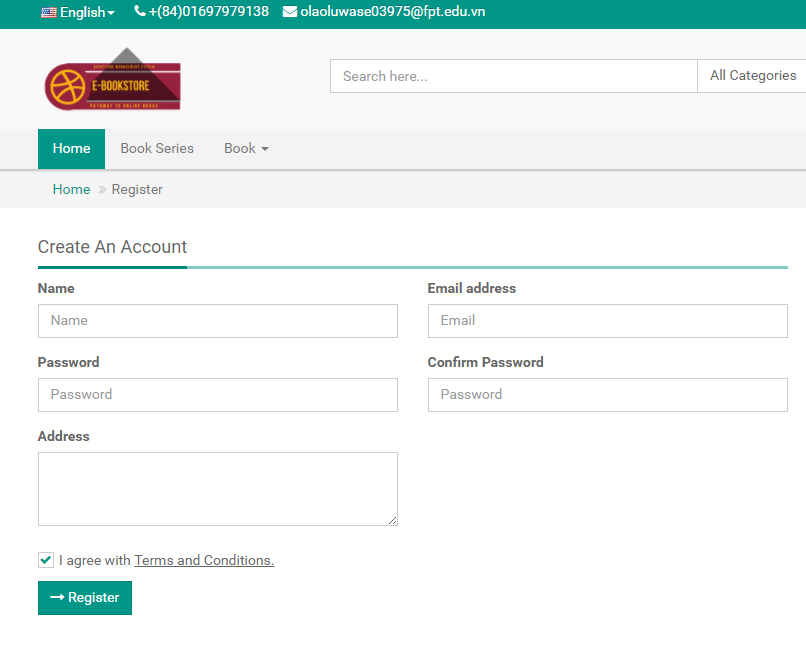


|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | AM01 | | **Version** | 1.0 |
| **Use Name** | | | Login | | | |
| **Actor** | | | Customer, Staff, Admin | | | |
| **Description** | | | This functionality specifically allows end-user to connect to the system, providing access to the overall in-built features. Users need to login to access their individual given level of access.  Login is the process by which an individual gains access to a computer system by identifying and authenticating themselves. | | | |
| **Precondition** | | | 1. End user must own a computer of any type and also E-U must have internet connection 2. E-U must access our website home page and then click on the login icon/button/link to navigate to the login page. 3. E-U must own a valid login detail | | | |
| **Trigger** | | | Login Button clicked | | | |
| **Post-Condition** | | | Homepage | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Customer  Admin  Staff | | Type URL into location field of internet browser and then press enter | | | |
| 2 | BSMS | | Display Login screen with the following fields:  - User Name  - Password  - Login button | | | |
| 3 | Customer  Admin  Staff | | Enter user name & password into User Name & Password fields on the Login screen, and then click on the Login button. | | | |
| 4 | BSMS | | Validate the entered user name & password and then display Home screen | | | |
|  |  | |  | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | Alternative flow, should incase an error occur in the program, either username or password is incorrect. | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 4 | | BSMS | | Display Login page with message "Your user name or password is incorrect." | | |
|  | |  | |  | | |
| **AT2** | | Username or Password field are is empty | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 4.1 | | BSMS | | Display login page with error message “Username and Password fields are required” | | |

**Business Rules**

|  |  |
| --- | --- |
| **#** | **Rule Description** |
| LGBR01 | Length of username must be greater than 4 |
| LGBR02 | Password must contain at-least a number |
| LGBR03 | Password length must be greater than 5 |
| LGBR04 | Username or Password cannot be empty |

#### Register Customer

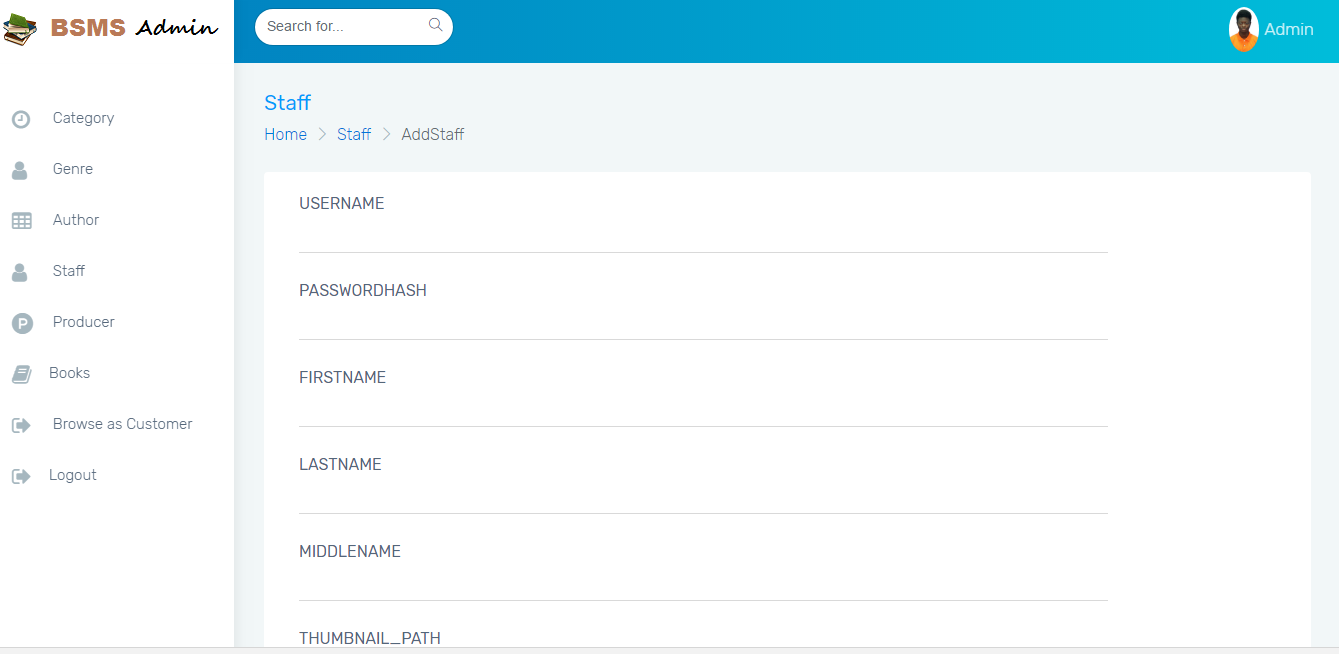
**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | AM02 | | **Version** | 1.0 |
| **Use Name** | | | Register Customer | | | |
| **Actor** | | | Guest | | | |
| **Description** | | | This functionality allow guest of BSMS to become a customer, thus they will be allow to the login functionalities BSMS, an also giving access to use all the functionalies of the system that match there prospective role | | | |
| **Precondition** | | | 1. End user must own a computer of any type and also E-U must have internet connection 2. E-U must access our website home page and then click on the login icon/button/link and then click on the Register to navigate to the Registration page. | | | |
| **Trigger** | | | Register button clicked | | | |
| **Post-Condition** | | | Registration Confirmation Page | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Guest | | Type URL into location field of internet browser and then press enter and navigate to the Register Page from the home page | | | |
| 2 | BSMS | | Display Registration screen with the following fields:  - User Name  - Password  - Email  - ……..  - …….  - radio-button EULA term  - Register button | | | |
| 3 | Guest | | 1. Fills in correct registration data in the (\*) fields 2. Check the Agree with Term of Service. 3. Click the Register button | | | |
| 4 | BSMS | | Validate registration information, if correct is correct entered in a normal case. Registration successful Information message is shown to Guest-User. | | | |
|  |  | |  | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | Alternative flow, should incase any internal error occur, ‘can’t connect to database’ or web service request failed | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Display Error page with message "Internal System Error, please contact with administrator" | | |
|  | |  | |  | | |
| **AT2** | | Alternative flow, should incase an error occur in the program, if any of the (\*) fields is empty or contains incorrect input. | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 4.1 | | BSMS | | Display Registration page with message “(\*) fields are required and must contain valid values” | | |
|  | |  | |  | | |

**Business Rules**

|  |  |
| --- | --- |
| **#** | **Rule Description** |
| RGBR01 | Length of username must be greater than 4 |
| RGBR02 | Password must contain at-least a number |
| RGBR03 | Password length must be greater than 5 |
| RGBR04 | Username or Password cannot be empty |
| RGBR05 | Email Field must be valid |
| RGBR06 | Email cannot be reused, “Email Already in use” |

#### Register Staff



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | AM03 | | **Version** | 1.0 |
| **Use Name** | | | Register Staff | | | |
| **Actor** | | | Admin | | | |
| **Description** | | | This functionality allow Admin to create a staff in the system, the functionality is only allow of the admin, as a staff is not allowed to register him or herself, after the below case is executed in a normal case staff will be allowed to login to the system | | | |
| **Precondition** | | | 1. Admin must own a computer of any type 2. Admin must be logged on to the system 3. Admin should navigate to Add Staff by clicking in the navigation bar “Staff” -> “Add Staff” | | | |
| **Trigger** | | |  | | | |
| **Post-Condition** | | | Before clicking on the register button Admin must make sure that (\*) field are not null and are valid. | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Admin | | Type URL into location field of internet browser and then press enter and navigate to the Add Staff from the navigation bar | | | |
| 2 | BSMS | | Display Registration screen with the following fields:  - User Name  - Email  - ……..  - …….  -Add Staff button | | | |
| 3 | Admin | | 1. Fills in correct registration data in the (\*) fields 2. Click the Add Staff button | | | |
| 4 | BSMS | | Validate registration information, if correct is correct entered in a normal case. Staff Addition success Information message is shown to Admin. | | | |
|  |  | |  | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | Alternative flow, should incase an error occur in the program, if any of the (\*) fields is empty or contains incorrect input. | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Display Registration page with message “(\*) fields are required and must contain valid values” | | |

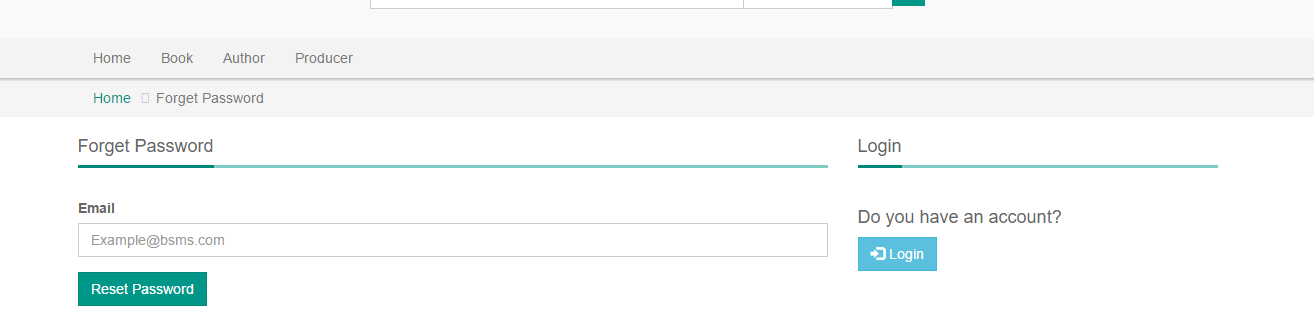
**Business Rules**

|  |  |
| --- | --- |
| **#** | **Rule Description** |
| RGBR05 | Email Field must be valid |
| RGBR06 | Email cannot be reused, “Email Already in use” |

#### Logout

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | AM04 | **Version** | 1.0 |
| **Use Name** | | Logout | | |
| **Actor** | | Customer, Staff, Admin | | |
| **Description** | | This functionality specifically allow all type of logged users to sign out of the system, On click of logout icon or text from any page E-U will be redirected to Login page | | |
| **Precondition** | | 1. Customer, Staff or Admin should have been previously logged in to the system | | |
| **Trigger** | | Logout button clicked from the navigation bar from any page | | |
| **Post-Condition** | | On Logging out BSMS will redirect users to the home page and allow them to browse like a normal guest. | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1 | Customer  Admin  Staff | Click on the Logout hyperlink in the any page | | |
| 2 | BSMS | Redirect to Home Page | | |

#### Forget Password

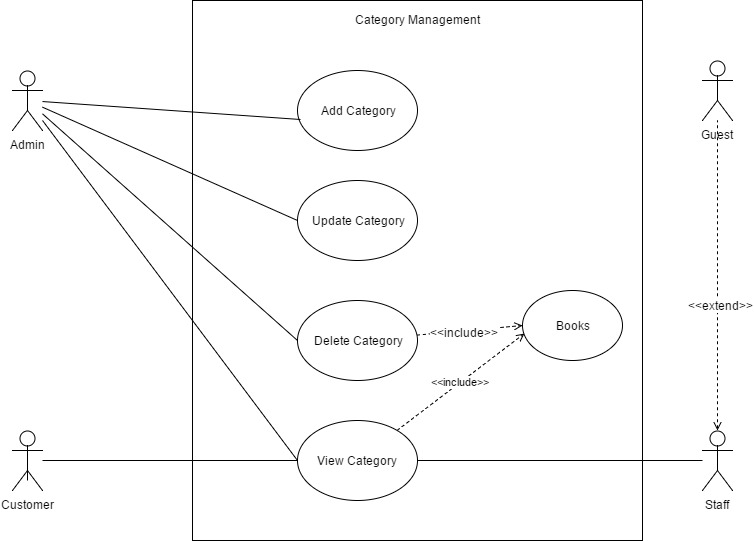


|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | AM05 | | **Version** | 1.0 |
| **Use Name** | | | Forget Password | | | |
| **Actor** | | | Staff, Customer | | | |
| **Description** | | | The forget password is used to retriever users passwords should incase they lose their previous password, a lot of user have varities of password they use in registering account, BSMS allow users who forget their password the functionality to reset their passwords | | | |
| **Precondition** | | | 1. To use this feature user must have a valid BSMS account, 2. Staff or customers must remember and have full access to their own email. 3. Staff or Customer clicked on the forget password on the login page 4. E-U filled in correct existing email in the system | | | |
| **Trigger** | | | Forget Password hyperlink clicked from the login page | | | |
| **Post-Condition** | | | N/A | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Customer  Staff | | Type URL into location field of internet browser and then press enter and navigate to the Login page in the navigation bar, click on forget password. | | | |
| 2 | BSMS | | Display Forget Password page with the following field:  - Email  -Add Staff button | | | |
| 3 | Customer  Staff | | 1. Fills in correct registration data in the (\*) fields 2. Click the Add Staff button | | | |
| 4 | BSMS | | Validate email, if email is in correct format and exist in our system email notification will send in-order to move on to the next process | | | |
|  |  | |  | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | Alternative flow, should incase an error occur in the program, if Email fields is empty or contains incorrect input. | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Display Registration page with message “(\*) fields are required and must contain valid values” | | |
|  | |  | |  | | |

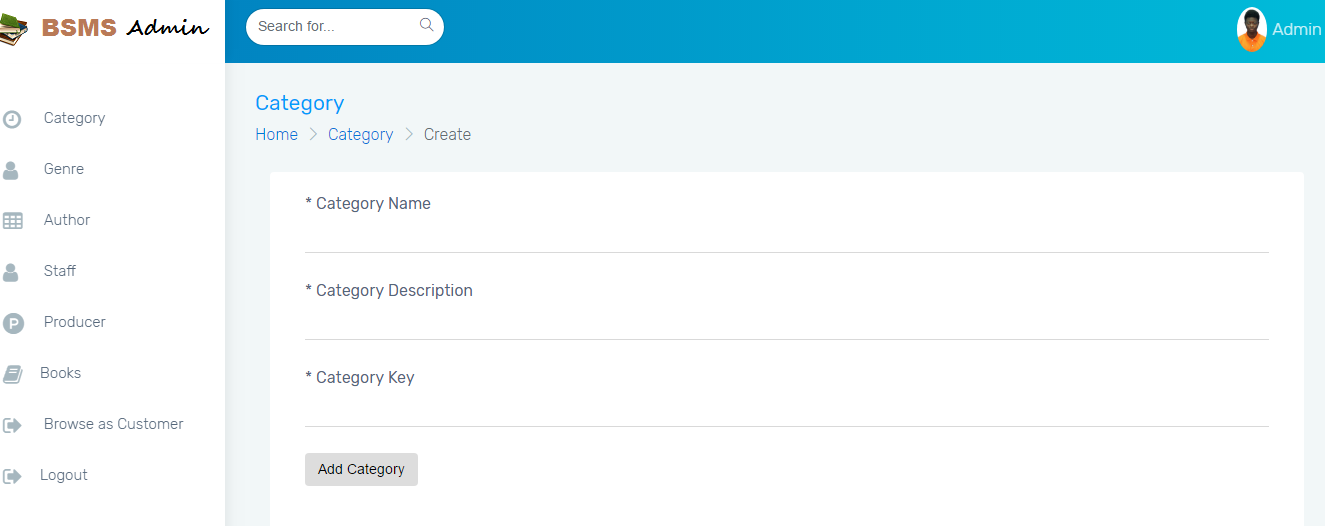
**Business Rules**

|  |  |
| --- | --- |
| **#** | **Rule Description** |
| FGBR01 | Email Address must be valid |
| FGBR02 | Email Address must have previously exist in BSMS |

### Category Management

** Figure 2.3.2**

#### Add Category

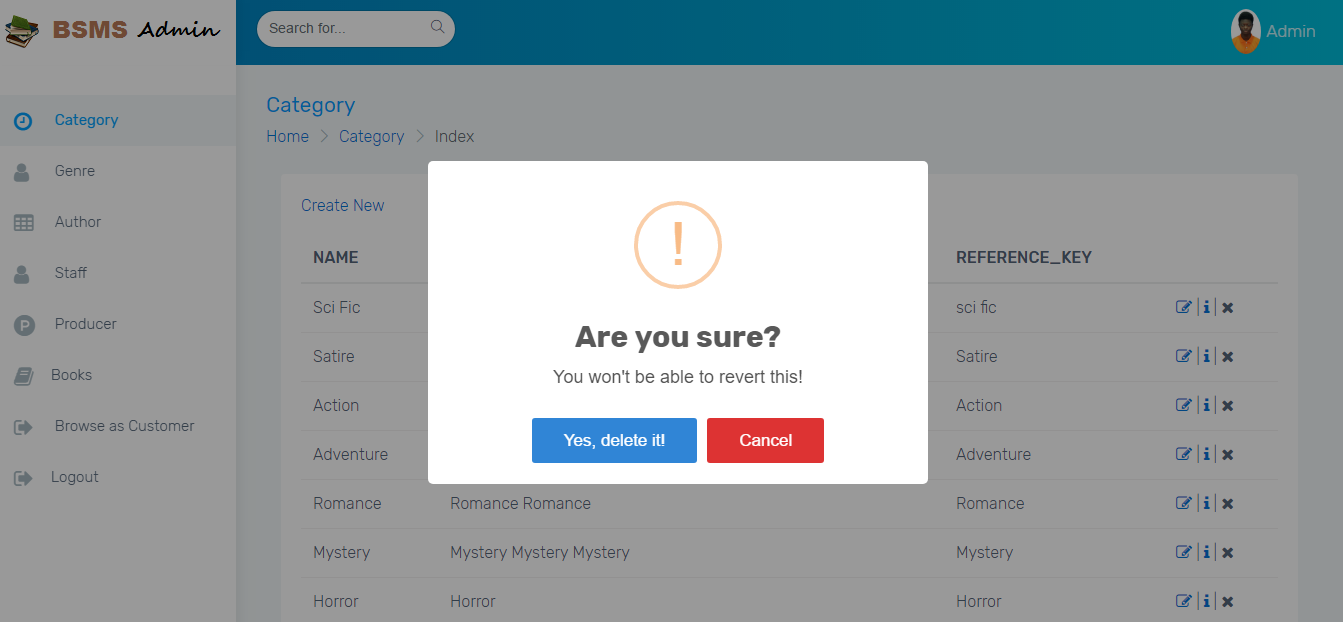


|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | CM01 | | **Version** | 1.0 |
| **Use Name** | | | Add Category | | | |
| **Actor** | | | Admin | | | |
| **Description** | | | In order to have category in our system, the admin might need to add one or more category so that we can categorize our books.  Adding Category in to the system is only allowed of the admin, which after the staff might add new book under the category. | | | |
| **Precondition** | | | An admin needs to be logged in to the system with valid account information | | | |
| **Trigger** | | | Add Category button click | | | |
| **Post-Condition** | | | Category List Page | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Admin | | Hovers on the category from the navigation bar and then clicked on the add category button. | | | |
| 2 | BSMS | | Display Add Category page with the following field:  - Name  -Description  -Upload Thumbnail  -Add Category Button | | | |
| 3 | Admin | | 1. Fills in correct registration data in the (\*) fields 2. Click the Add Category button | | | |
| 4 | BSMS | | Validate category information, if category fields are correct in a normal case. Category addition success Information message is shown to Admin. | | | |
|  |  | |  | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | Alternative flow, should incase an error occur in the program, if any of the (\*) fields is empty or contains incorrect input. | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Display Add category page with message “(\*) fields are required and must contain valid values” | | |
|  | |  | |  | | |

**Business Rules**

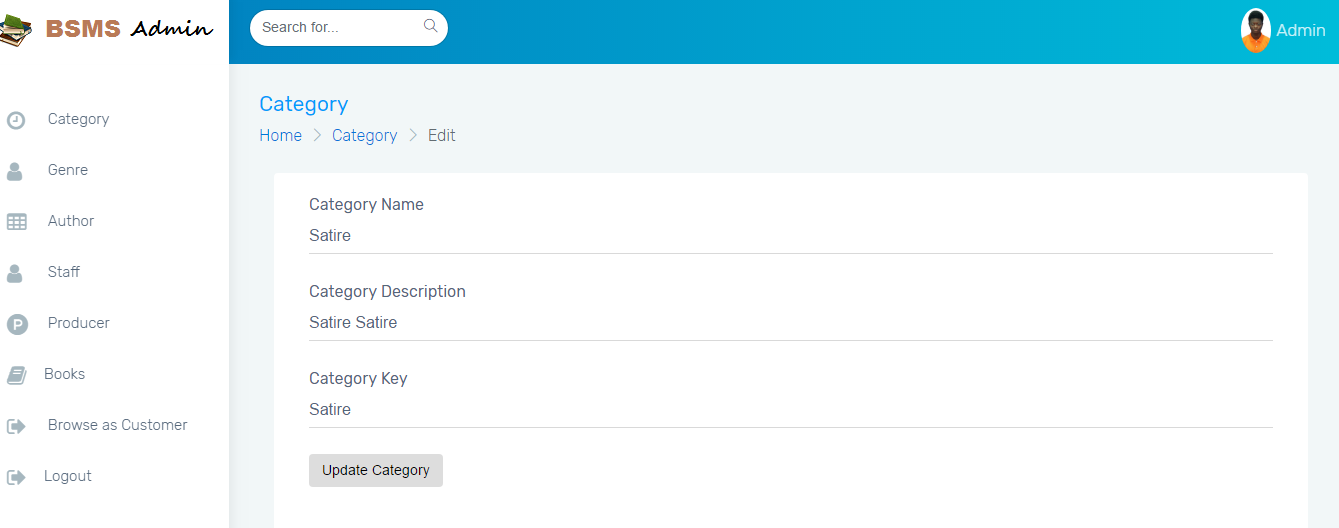
|  |  |
| --- | --- |
| **#** | **Rule Description** |
| ACBR01 | Category name must not be null and length must be greater than or equals 3 |
| ACBR02 | Category Image is not optional. |

#### Delete Category



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | CM02 | | **Version** | 1.0 |
| **Use Name** | | | Delete Category | | | |
| **Actor** | | | Admin | | | |
| **Description** | | | This functionality allow previously created category to be deleted from the system when they are no more needed, deleting of categories is a functionality allowed of the Admin user alone. | | | |
| **Precondition** | | | An admin needs to be logged in to the system with valid account information | | | |
| **Trigger** | | | Click the delete button on any category | | | |
| **Post-Condition** | | | N/A | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Admin | | Click on the Delete category button in the category List page | | | |
| 2 | BSMS | | Show delete confirm message | | | |
| 3 | Admin | | Admin click “Ok” in the confirm message box | | | |
| 4 | BSMS | | Delete Category from the system. | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | **The Admin click cancel on the confirm message box shown on the screen** | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Dispose the action | | |
|  | |  | |  | | |

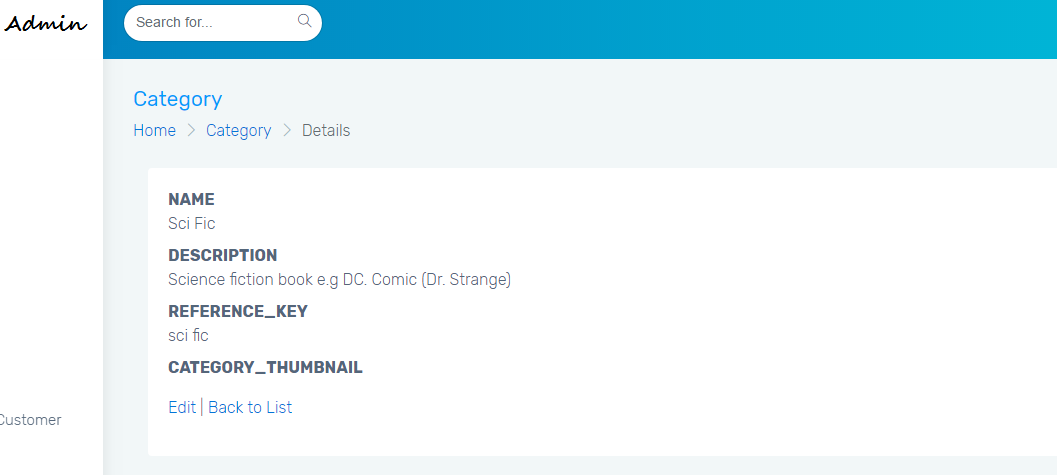
#### Update Category



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | CM03 | | **Version** | 1.0 |
| **Use Name** | | | Update Category | | | |
| **Actor** | | | Admin | | | |
| **Description** | | | This functionality allow previously created category to be edited if there exist any wrong information, editing information of category including the category name and other information e.g. changing category name from “Science Fiction” to “Sci Fi” | | | |
| **Precondition** | | | An admin needs to be logged in to the system with valid account information | | | |
| **Trigger** | | | Update Button | | | |
| **Post-Condition** | | | Category List page | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Admin | | Click on edit button on top any category in the category list page. | | | |
| 2 | BSMS | | Display update page with pre-populated data values that belongs to the category update button was clicked on. | | | |
| 3 | Admin | | Admin change desired fields then click the update button | | | |
| 4 | BSMS | | Display a confirm message to be sure, the Admin really want to update this category “Are you sure you want to update this category information” | | | |
| 5 | Admin | | Click ok on the confirmation message | | | |
| 6 | BSMS | | Update Category Information as update by the admin. | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | **The Admin click cancel on the confirm message box shown on the screen** | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Dispose the action | | |
| **AT2** | | **The Admin Click cancel before clicking on the Update button on the page** | | | | |
|  | | BSMS | | Redirect back to Category list page | | |

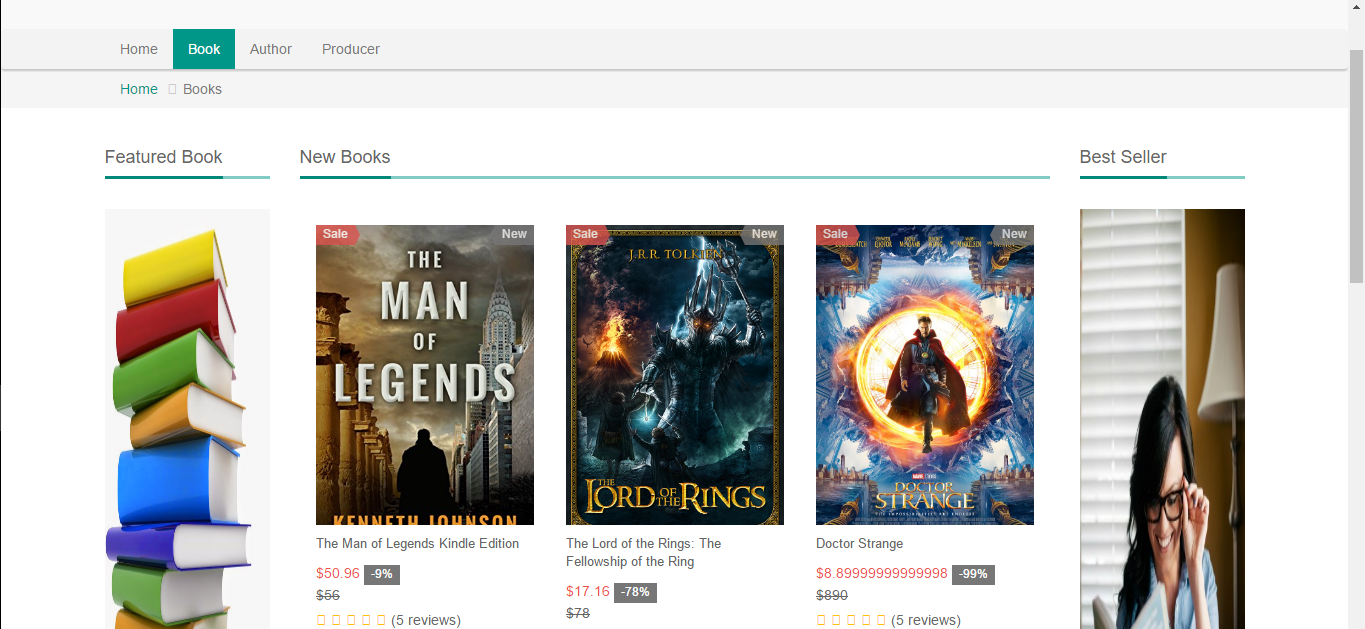
|  |  |
| --- | --- |
| **#** | **Rule Description** |
| UCBR01 | Category name must not be null and length must be greater than or equals 3 |
| UCBR02 | Category Image is not optional. |

#### View Category



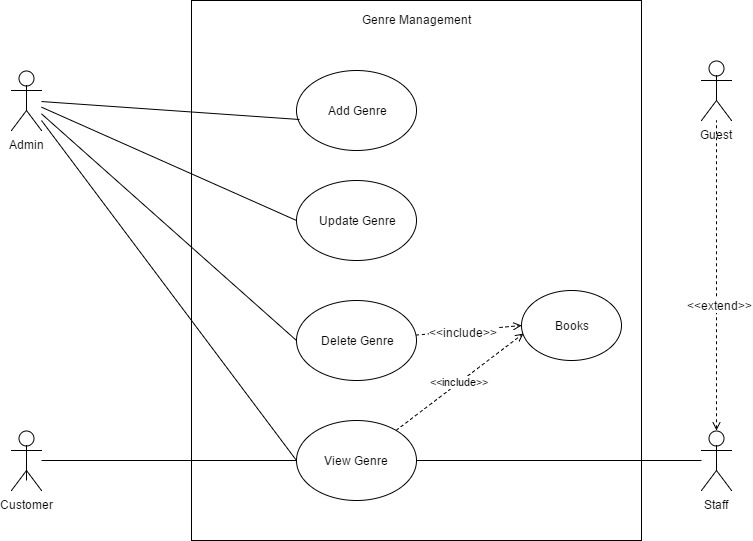
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | CM04 | **Version** | 1.0 |
| **Use Name** | | View Category | | |
| **Actor** | | Admin, Staff, Customers, Guest | | |
| **Description** | | This allows users of any access level to view categories so that they can view books under this category.  . | | |
| **Precondition** | | An admin needs to be logged in to the system with valid account information | | |
| **Trigger** | | Click category button in the navigation bar from any page | | |
| **Post-Condition** | | Category List page | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1 | Admin  Customer  Staff  Guest | Click on Category from the navigation bar | | |
| 2 | BSMS | Display list of categories with buttons under each one of them “View glyph icon ” or “View” | | |

#### View Books



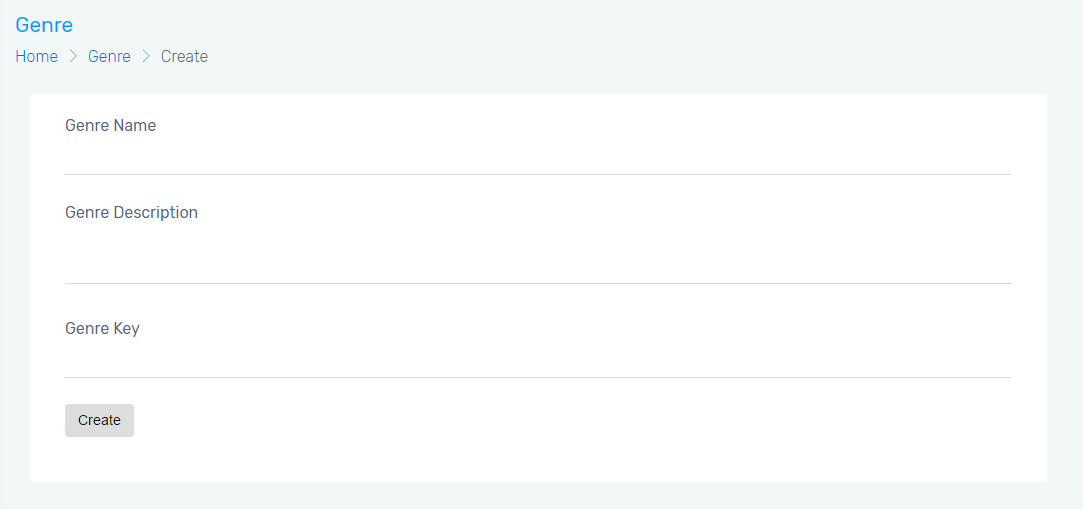
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | CM05 | **Version** | 1.0 |
| **Use Name** | | View Books | | |
| **Actor** | | Admin, Staff, Guest, Customer | | |
| **Description** | | This functionality allow previously created category to be edited if there exist any wrong information. | | |
| **Precondition** | | An admin needs to be logged in to the system with valid account information | | |
| **Trigger** | | Click on Glyph icon view or “View” button | | |
| **Post-Condition** | | Book List page | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1 | Admin, Staff, Guest, Customer | Click on Glyph icon view or “View” button | | |
| 2 | BSMS | Display Book list page with all book within the selected category | | |

### Genre Management

****

**Figure 2.3.2**

#### Add Genre



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | GM01 | | **Version** | 1.0 |
| **Use Name** | | | Add Genre | | | |
| **Actor** | | | Admin | | | |
| **Description** | | | In order to have Genre in our system, the admin might need to add one or more genre so that each categorize can use the genre, inorder to add a book into BSMS, the admin need to add an equivalent genre that match that of the new book. | | | |
| **Precondition** | | | An admin needs to be logged in to the system with valid account information | | | |
| **Trigger** | | | Add Genre button click | | | |
| **Post-Condition** | | | Genre List Page | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Admin | | Hovers on the genre from the navigation bar and then clicked on the add genre button. | | | |
| 2 | BSMS | | Display Add Genre page with the following field:  - Name  -Description  -Upload Thumbnail  -Add Genre Button | | | |
| 3 | Admin | | 1. Fills in correct registration data in the (\*) fields 2. Click the Add Genre button | | | |
| 4 | BSMS | | Validate genre information, if genre fields are correct in a normal case. Genre addition success Information message is shown to Admin. | | | |
|  |  | |  | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | Alternative flow, should incase an error occur in the program, if any of the (\*) fields is empty or contains incorrect input. | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Display Add Genre page with message “(\*) fields are required and must contain valid values” | | |
|  | |  | |  | | |

**Business Rules**

|  |  |
| --- | --- |
| **#** | **Rule Description** |
| AGBR01 | Genre name must not be null and length must be greater than or equals 3 |
| AGBR02 | Genre Image is not optional. |

#### Delete Genre

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | GM02 | | **Version** | 1.0 |
| **Use Name** | | | Delete Genre | | | |
| **Actor** | | | Admin | | | |
| **Description** | | | This functionality allow previously created genre to be deleted from the system when they are no more needed, assuming the case in which the admin add duplicate book genre and wish to delete the irrelevant one, then this functionality will play a key role in completing this action. | | | |
| **Precondition** | | | An admin needs to be logged in to the system with valid account information | | | |
| **Trigger** | | | Click the delete button on any genre | | | |
| **Post-Condition** | | | N/A | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Admin | | Click on the Delete genre button in the genre List page | | | |
| 2 | BSMS | | Show delete confirm message | | | |
| 3 | Admin | | Admin click “Ok” in the confirm message box | | | |
| 4 | BSMS | | Delete genre from the system. | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | **The Admin click cancel on the confirm message box shown on the screen** | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Dispose the action | | |
|  | |  | |  | | |

#### Update Genre

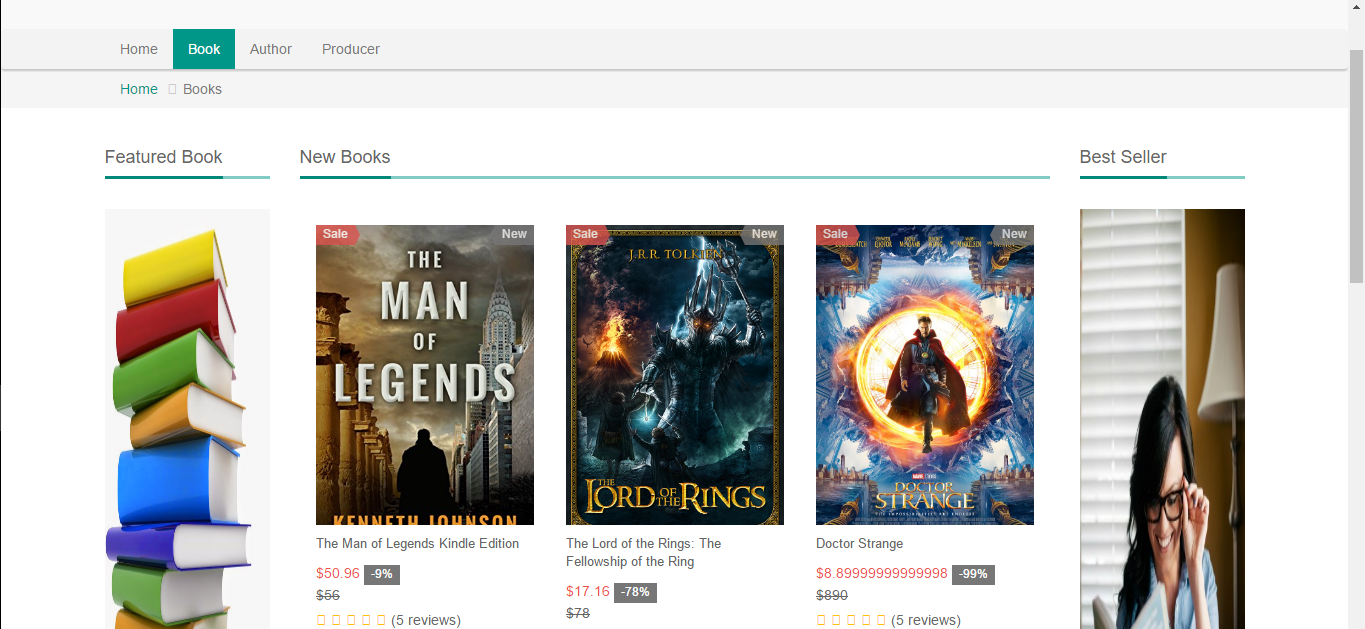
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | GM03 | | **Version** | 1.0 |
| **Use Name** | | | Update Genre | | | |
| **Actor** | | | Admin | | | |
| **Description** | | | This functionality allow previously created genre to be edited if there exist any wrong information, editing information of genre including the genre name and other information e.g. changing genre name from “Above 18” to “18+”, this functionality will play a key role if the admin make mistake or wish to make a major change in the system. | | | |
| **Precondition** | | | An admin needs to be logged in to the system with valid account information | | | |
| **Trigger** | | | Update Button | | | |
| **Post-Condition** | | | Category List page | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Admin | | Click on edit button on top any genre in the category list page. | | | |
| 2 | BSMS | | Display update page with pre-populated data values that belongs to the genre update button was clicked on. | | | |
| 3 | Admin | | Admin change desired fields then click the update button | | | |
| 4 | BSMS | | Display a confirm message to be sure, the Admin really want to update this genre “Are you sure you want to update this genre information” | | | |
| 5 | Admin | | Click ok on the confirmation message | | | |
| 6 | BSMS | | Update genre Information as updated by the admin. | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | **The Admin click cancel on the confirm message box shown on the screen** | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Dispose the action | | |
| **AT2** | | **The Admin Click cancel before clicking on the Update button on the page** | | | | |
|  | | BSMS | | Redirect back to genre list page | | |

|  |  |
| --- | --- |
| **#** | **Rule Description** |
| UGBR01 | Genre name must not be null and length must be greater than or equals 3 |
| UGBR02 | Genre Image is not optional. |

#### View Genre

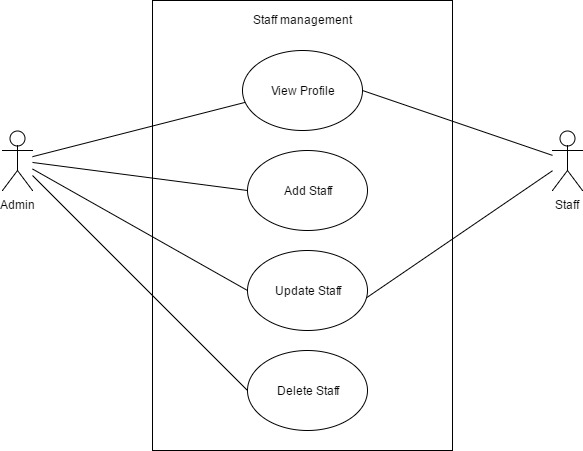
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | GM04 | **Version** | 1.0 |
| **Use Name** | | View Genre | | |
| **Actor** | | Admin, Staff, Customers, Guest | | |
| **Description** | | This allows users of any access level to view genre so that they can view books under this genre, Viewing genre means viewing all information that are embedded in | | |
| **Precondition** | | An admin needs to be logged in to the system with valid account information | | |
| **Trigger** | | Click genre button in the navigation bar from any page | | |
| **Post-Condition** | | Genre List page | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1 | Admin  Customer  Staff  Guest | Click on Genre from the navigation bar | | |
| 2 | BSMS | Display list of genre with buttons under each one of them “View glyph icon ” or “View” | | |

#### View Books



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | GM05 | **Version** | 1.0 |
| **Use Name** | | View Books | | |
| **Actor** | | Admin, Staff, Guest, Customer | | |
| **Description** | | This functionality allow previously created genre to be edited if there exist any wrong information. | | |
| **Precondition** | | E-U of any type navigates to the genre page, either when logged in or not. | | |
| **Trigger** | | Click on Glyph icon view or “View” button | | |
| **Post-Condition** | | Book List page | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1 | Admin | Click on Glyph icon view or “View” button | | |
| 2 | BSMS | Display Book list page with all book within the selected genre | | |

### Staff Management



**Figure 2.3.4**

#### View Profile

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | SM01 | **Version** | 1.0 |
| **Use Name** | | View Profile | | |
| **Actor** | | Admin, Staff | | |
| **Description** | | This functionality allow staff to view his own profile and also allow the admin to view profile or any staff, in-case if he need to find out some information about his staffs | | |
| **Precondition** | | 1. Staff click on his thumbnail image or name on the navigation bar, 2. Admin can view list of staffs in the system and click on the button View Profile | | |
| **Trigger** | | View Profile hyperlink, image icon in the navigation bar clicked or name | | |
| **Post-Condition** | | Profile Page | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1 | Staff | Click on his/her image thumbnail or name in the navigation bar from any page | | |
| 2 | BSMS | Redirect staff to his/her own profile page | | |
| 3 | Admin | 1. Go to list of staff on his/her dashboard 2. click view profile button on any staff his wish to see | | |
|  | BSMS | Display staff profile page to admin | | |

#### Add Staff

For this functional requirement, kindly refer to **AM03** in the Authentication Management.

#### Update Staff

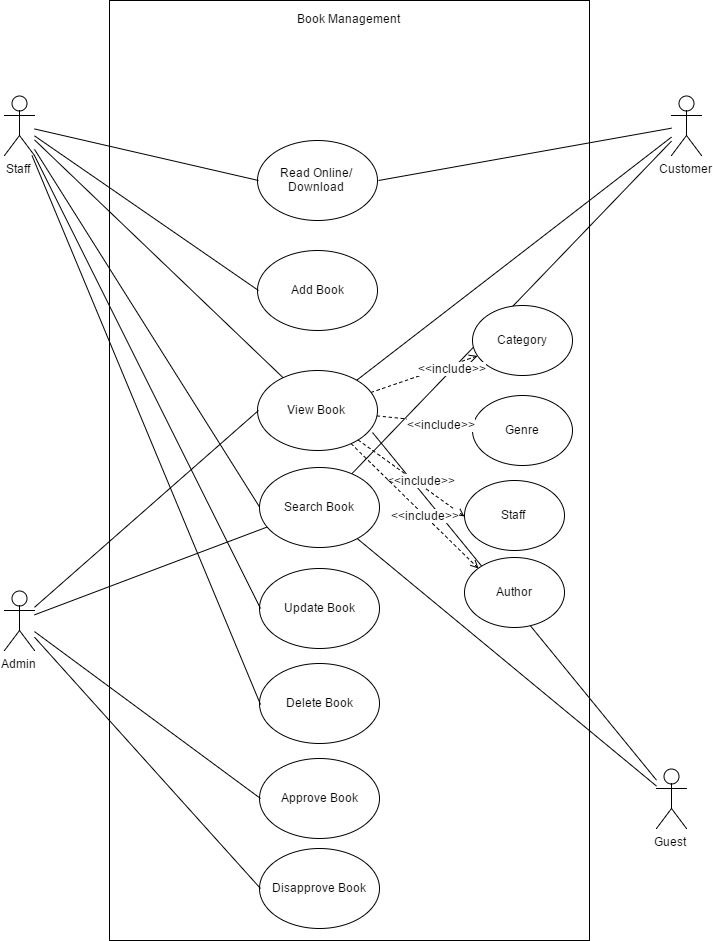
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | SM02 | | **Version** | 1.0 |
| **Use Name** | | | Update Staff | | | |
| **Actor** | | | Admin | | | |
| **Description** | | | This functionality allow previously created Staff to be edited if there exist any wrong information. | | | |
| **Precondition** | | | An admin needs to be logged in to the system with valid account information | | | |
| **Trigger** | | | Update Button | | | |
| **Post-Condition** | | | Category List page | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Admin | | Click on edit button on top any Staff in the category list page. | | | |
| 2 | BSMS | | Display update page with pre-populated data values that belongs to the Staff update button was clicked on. | | | |
| 3 | Admin | | Admin change desired fields then click the update button | | | |
| 4 | BSMS | | Display a confirm message to be sure, the Admin really want to update this Staff “Are you sure you want to update this Staff information” | | | |
| 5 | Admin | | Click ok on the confirmation message | | | |
| 6 | BSMS | | Update Staff Information as updated by the admin. | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | **The Admin click cancel on the confirm message box shown on the screen** | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Dispose the action | | |
| **AT2** | | **The Admin Click cancel before clicking on the Update button on the page** | | | | |
|  | | BSMS | | Redirect back to Staff list page | | |

|  |  |
| --- | --- |
| **#** | **Rule Description** |
| UGBR01 | Staff name must not be null and length must be greater than or equals 3 |
| UGBR02 | Staff Image is not optional. |

#### Delete Staff

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | SM03 | | **Version** | 1.0 |
| **Use Name** | | | Delete Staff | | | |
| **Actor** | | | Admin | | | |
| **Description** | | | This functionality allow previously created Staff to be deleted from the system when they are no more needed. | | | |
| **Precondition** | | | An admin needs to be logged in to the system with valid account information | | | |
| **Trigger** | | | Click the delete button on any Staff | | | |
| **Post-Condition** | | | N/A | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Admin | | Click on the Delete Staff button in the Staff List page | | | |
| 2 | BSMS | | Show delete confirm message | | | |
| 3 | Admin | | Admin click “Ok” in the confirm message box | | | |
| 4 | BSMS | | Delete taff from the system. | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | **The Admin click cancel on the confirm message box shown on the screen** | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Dispose the action | | |
|  | |  | |  | | |

### Book Management



**Figure 2.3.5**

#### Read Book Online/Download

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | BM01 | **Version** | 1.0 |
| **Use Name** | | Read Book Online/Download | | |
| **Actor** | | Staff, Customer | | |
| **Description** | | This feature main target is for customer to be able to View free books online or download | | |
| **Precondition** | | 1. Staff or Customer is logged in to the system 2. Selected book is tagged free | | |
| **Trigger** | | Click on “View Glyph Icon” displayed on books tagged “Free” or download Glyph Icon | | |
| **Post-Condition** | | pdf or document format of book is displayed on a new browser page | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1. | Staff  Customer | Customer click on “Download/Read Online Glyph Icon” on any book tagged free.  Staff click on preview book , on any book in the Book List page | | |
| 2 | BSMS | 1. Send file to browser for download if softcopy format is in .docx file002E 2. Preview File online if book softcopy is in pdf format | | |
|  |  |  | | |

#### Add Book

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | BM02 | | **Version** | 1.0 |
| **Use Name** | | | Add Book | | | |
| **Actor** | | | Admin | | | |
| **Description** | | | This functionality allow Admin to create a book in the system | | | |
| **Precondition** | | | 1. Admin must own a computer of any type 2. Admin must be logged on to the system 3. Admin should navigate to Add Book by clicking in the navigation bar “Book” -> “Add Book” | | | |
| **Trigger** | | |  | | | |
| **Post-Condition** | | | Before clicking on the register button Admin must make sure that (\*) field are not null and are valid. | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Admin | | Type URL into location field of internet browser and then press enter and navigate to the Book navigation bar then click on Add Book from the dropdown list | | | |
| 2 | BSMS | | Display Registration screen with the following fields:  - Category  - Genre  - Name  - Author  - Producer  - …….  - Add Book button | | | |
| 3 | Admin | | 1. Fills in correct registration data in the (\*) fields 2. Click the Add Book button | | | |
| 4 | BSMS | | Validate registration information, if correct book information is entered in a normal case. Book Addition success Information message is shown to Admin. | | | |
|  |  | |  | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | Alternative flow, should incase an error occur in the program, if any of the (\*) fields is empty or contains incorrect input. | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Display Registration page with message “(\*) fields are required and must contain valid values” | | |

**Business Rules**

|  |  |
| --- | --- |
| **#** | **Rule Description** |
| ABR01 | (\*) field must cannot be empty |
| ABR02 | Book name should be greater than or equals length of 4 |
| ABR03 | Book Name cannot be null |
| ABR04 | Price must be in decimal |
| ABR06 | Synopsis length must be more than 100 words |

#### View Book

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | BM03 | **Version** | 1.0 |
| **Use Name** | | View Book | | |
| **Actor** | | Admin, Staff, Guest, Customer | | |
| **Description** | | This functionality allow all users of any type to view a book, Viewing a book means you will see all the book details like Synopsis, Producer, ISBN number including book author and many other information | | |
| **Precondition** | | On home page or Book list page, click the Detail button on any book of your own choice to view its information, to do this logging in is not required as guest users are also allow to access this functionality. | | |
| **Trigger** | | “View Detail” button clicked on any book in either the home page or book list page | | |
| **Post-Condition** | | N/A | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1 | Admin  Staff  Customer  Guest | 1. Navigate the list book page or the home page 2. Find any book they want to view. 3. Then click the “View Detail” at the lower side of the book | | |
| 2 | BSMS | Redirect user to the Book Detail page | | |

#### Search Book

For this functional requirement, kindly refer to use case **SR01** in the Search Management.

#### Update Book

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | BM04 | | **Version** | 1.0 |
| **Use Name** | | | Update Book | | | |
| **Actor** | | | Staff | | | |
| **Description** | | | This functionality allows previously created Book to be edited if there exist any wrong information. | | | |
| **Precondition** | | | A Staff needs to be logged in to the system with valid account information | | | |
| **Trigger** | | | Update Button | | | |
| **Post-Condition** | | | Book List Page | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Staff | | Click on edit button on top any Book in the book list page. | | | |
| 2 | BSMS | | Display update page with pre-populated data values that belongs to the Book update button was clicked on. | | | |
| 3 | Staff | | Staff change desired fields then click the update button | | | |
| 4 | BSMS | | Display a confirm message to be sure, the Staff really want to update this Book “Are you sure you want to update this Book information” | | | |
| 5 | Staff | | Click ok on the confirmation message | | | |
| 6 | BSMS | | Update Book Information as updated by the staff. | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | **The Staff click cancel on the confirm message box shown on the screen** | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Dispose the action | | |
| **AT2** | | **The Staff Click cancel before clicking on the Update button on the page** | | | | |
|  | | BSMS | | Redirect back to Book list page | | |

**Business Rules**

|  |  |
| --- | --- |
| **#** | **Rule Description** |
| ABR01 | (\*) field must cannot be empty |
| ABR02 | Book name should be greater than or equals length of 4 |
| ABR03 | Book Name cannot be null |
| ABR04 | Price must be in decimal |
| ABR06 | Synopsis length must be more than 100 words |

#### Delete Book

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | BM05 | | **Version** | 1.0 |
| **Use Name** | | | Delete Book | | | |
| **Actor** | | | Staff | | | |
| **Description** | | | This functionality allow previously created Book to be deleted from the system when they are no more needed. | | | |
| **Precondition** | | | An Staff needs to be logged in to the system with valid account information | | | |
| **Trigger** | | | Click the delete button on any Book | | | |
| **Post-Condition** | | | Book list page | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Staff | | Click on the Delete Book button in the Book List page | | | |
| 2 | BSMS | | Show delete confirm message | | | |
| 3 | Staff | | Staff click “Ok” in the confirm message box | | | |
| 4 | BSMS | | Delete Book from the system. | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | **The Staff click cancel on the confirm message box shown on the screen** | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Dispose the action | | |
|  | |  | |  | | |

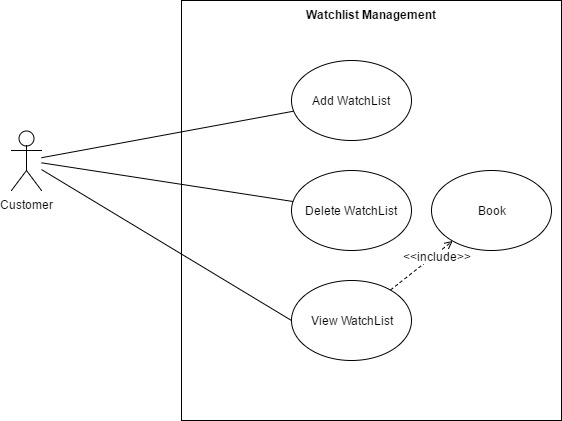
#### Approve Book

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | BM06 | **Version** | 1.0 |
| **Use Name** | | Approve Book | | |
| **Actor** | | Admin | | |
| **Description** | | After staff add a book in the system they remain hidden to all users apart from admin, admin can remit that by reviewing and approving the book, this way the book will be published and all user will be able to see the book as part of the list | | |
| **Precondition** | | Click on View Detail of book from “Pending approval page” | | |
| **Trigger** | | Approve button clicked | | |
| **Post-Condition** | | Pending Approval Book page | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1. | Admin | Admin go to the “Book” in the navigation bar and click on pending approval in the dropdown list.  Then Click the button View Detail | | |
| 2. | BSMS | Displays Book Detail | | |
| 3. | Admin | Review new book information and clicked approve | | |
| 4. | BSMS | Publish book to all user. | | |
|  |  |  | | |

#### Disapprove Book

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | BM07 | **Version** | 1.0 |
| **Use Name** | | Disapprove Book | | |
| **Actor** | | Admin | | |
| **Description** | | When staff input invalid book information into the system, they remain hidden from all users apart from the admin, the admin can then take actions like disapproving/rejecting such book so that it will never have to appear to other E-U apart from him.  Rejected book are more like deleted book, the difference between both methods is that deleted book leave the system completely as they can never return but rejected books still serve the opportunity of being approve later in the future. | | |
| **Precondition** | | Click on View Detail of book from “Pending approval page” | | |
| **Trigger** | | Disapprove button clicked | | |
| **Post-Condition** | | Pending Approval Book page | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1. | Admin | Admin go to the “Book” in the navigation bar and click on pending approval in the dropdown list.  Then Click the button View Detail | | |
| 2. | BSMS | Displays Book Detail | | |
| 3. | Admin | Review new book information and clicked disapprove or reject button | | |
| 4. | BSMS | Keeps book hidden forever unless Admin take another action on such book. | | |
|  |  |  | | |

### Watch List Management



**Figure 2.3.6**

#### Add Wish List

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | WM01 | **Version** | 1.0 |
| **Use Name** | | Add Wish List | | |
| **Actor** | | Customer | | |
| **Description** | | This feature allow users to add book to wish list, this functionally is more of personalizing some book you wish to read in the future or maybe customer wish to buy, it more like the favorite feature. | | |
| **Precondition** | | Customer have a valid login information and he/she is logged into the system | | |
| **Trigger** | | Click on the “Love Glyph Icon” in product list page or home page | | |
| **Post-Condition** | | N/A | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1. | Customer | Click on the Icon or button “Add to Wish List” show on any book | | |
| 2. | BSMS | Add book to customer wish list. | | |
|  |  |  | | |

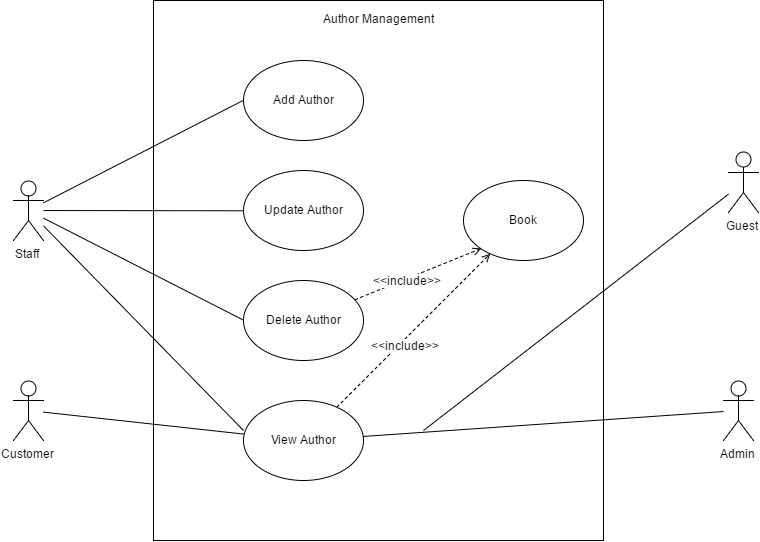
#### View Wish List

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | WM02 | **Version** | 1.0 |
| **Use Name** | | View Wish List | | |
| **Actor** | | Customer | | |
| **Description** | | This feature allow user to be able to view product in their wish list | | |
| **Precondition** | | Customer have a valid login information and he/she is logged into the system | | |
| **Trigger** | | N/A | | |
| **Post-Condition** | | Wish list page | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1. | Customer | Click on the wish list button or icon in the navigation bar | | |
| 2. | BSMS | Display wish list page, where Customer can find all their books | | |
|  |  |  | | |

#### Delete Wish List

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | WM03 | **Version** | 1.0 |
| **Use Name** | | Add Wish List | | |
| **Actor** | | Customer | | |
| **Description** | | This feature allow users to remove book from wish list items | | |
| **Precondition** | | Customer have a valid login information and he/she is logged into the system  Customer have previously add at least 1 product to his/her wish list  Customer have access the wish list page on the navigation bar | | |
| **Trigger** | | “Remove from Wish list“ button click | | |
| **Post-Condition** | | N/A | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1. | Customer | Click on the “Remove from wish list” button at the bottom of any book in the wish list page | | |
| 2. | BSMS | Remove book from customer wish list. | | |
|  |  |  | | |

### Author Management

****

**Figure 2.3.7**

#### Add Author

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | AUM01 | | **Version** | 1.0 |
| **Use Name** | | | Add Author | | | |
| **Actor** | | | Staff | | | |
| **Description** | | | In order to have Author in our system, the Staff might need to add one or more Author so that we can add author of our books. | | | |
| **Precondition** | | | An Staff needs to be logged in to the system with valid account information | | | |
| **Trigger** | | | Add Author button click | | | |
| **Post-Condition** | | | Author List Page | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Staff | | Hovers on the Author from the navigation bar and then clicked on the add Author button in the dropdown list. | | | |
| 2 | BSMS | | Display Add Author page with the following field:  - Name  -Description  -Upload Thumbnail  - …………..  - …………..  -Add Author Button | | | |
| 3 | Staff | | 1. Fills in correct registration data in the (\*) fields 2. Click the Add Author button | | | |
| 4 | BSMS | | Validate Author information, if Author fields are correct in a normal case. Author addition success Information message is shown to Staff. | | | |
|  |  | |  | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | Alternative flow, should incase an error occur in the program, if any of the (\*) fields is empty or contains incorrect input. | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Display Add Author page with message “(\*) fields are required and must contain valid values” | | |
|  | |  | |  | | |

**Business Rules**

|  |  |
| --- | --- |
| **#** | **Rule Description** |
| ACBR01 | Author name must not be null and length must be greater than or equals 3 |
| ACBR02 | Author Image is optional. |

#### Delete Author

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | AUM02 | | **Version** | 1.0 |
| **Use Name** | | | Delete Author | | | |
| **Actor** | | | Staff | | | |
| **Description** | | | This functionality allow previously created Author to be deleted from the system when they are no more needed. | | | |
| **Precondition** | | | An Staff needs to be logged in to the system with valid account information | | | |
| **Trigger** | | | Click the delete button on any Author | | | |
| **Post-Condition** | | | N/A | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Staff | | Click on the Delete Author button in the Author List page | | | |
| 2 | BSMS | | Show delete confirm message | | | |
| 3 | Staff | | Staff click “Ok” in the confirm message box | | | |
| 4 | BSMS | | Delete Author from the system. | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | **The Staff click cancel on the confirm message box shown on the screen** | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Dispose the action | | |
|  | |  | |  | | |

#### Update Author

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | AUM03 | | **Version** | 1.0 |
| **Use Name** | | | Update Author | | | |
| **Actor** | | | Staff | | | |
| **Description** | | | This functionality allow previously created Author to be edited if there exist any wrong information. | | | |
| **Precondition** | | | An Staff needs to be logged in to the system with valid account information | | | |
| **Trigger** | | | Update Button | | | |
| **Post-Condition** | | | Author List page | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Staff | | Click on edit button on top any Author in the Author list page. | | | |
| 2 | BSMS | | Display update page with pre-populated data values that belongs to the Author update button was clicked on. | | | |
| 3 | Staff | | Staff change desired fields then click the update button | | | |
| 4 | BSMS | | Display a confirm message to be sure, the Staff really want to update this Author “Are you sure you want to update this Author information” | | | |
| 5 | Staff | | Click ok on the confirmation message | | | |
| 6 | BSMS | | Update Author Information as update by the Staff. | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | **The Staff click cancel on the confirm message box shown on the screen** | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Dispose the action | | |
| **AT2** | | **The Staff Click cancel before clicking on the Update button on the page** | | | | |
|  | | BSMS | | Redirect back to Author list page | | |

|  |  |
| --- | --- |
| **#** | **Rule Description** |
| UCBR01 | Author name must not be null and length must be greater than or equals 3 |
| UCBR02 | Author Image is not optional. |

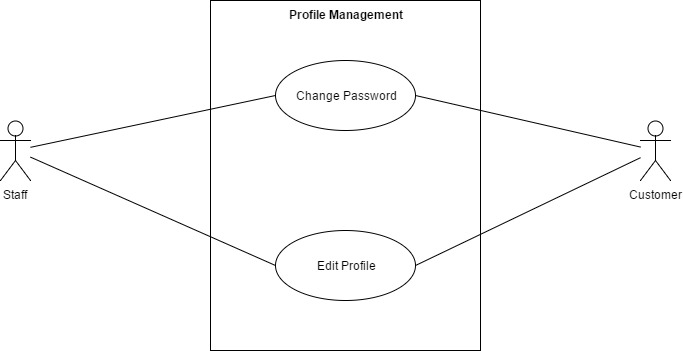
#### View Author

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | AUM04 | **Version** | 1.0 |
| **Use Name** | | View Author | | |
| **Actor** | | Staff, Admin, Customers, Guest | | |
| **Description** | | This allows users of any access level to view categories so that they can view books under this Author. | | |
| **Precondition** | | A Staff needs to be logged in to the system with valid account information | | |
| **Trigger** | | Click Author button in the navigation bar from any page | | |
| **Post-Condition** | | Author List page | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1 | Admin  Customer  Staff  Guest | Click on Author from the navigation bar | | |
| 2 | BSMS | Display list of categories with buttons under each one of them “View glyph icon ” or “View” | | |

#### View Books

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | AUM05 | **Version** | 1.0 |
| **Use Name** | | View Books | | |
| **Actor** | | Admin, Staff, Guest, Customer | | |
| **Description** | | This functionality allow previously created Author to be edited if there exist any wrong information. | | |
| **Precondition** | | An Staff needs to be logged in to the system with valid account information | | |
| **Trigger** | | Click on Glyph icon view or “View” button | | |
| **Post-Condition** | | Book List page | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1 | Admin, Staff, Guest, Customer | Click on Glyph icon view or “View” button | | |
| 2 | BSMS | Display Book list page with all book within the selected Author | | |

### Profile Management



**Figure 2.3.8**

#### Change password

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | PM01 | | **Version** | 1.0 |
| **Use Name** | | | Change password | | | |
| **Actor** | | | Staff, Customer | | | |
| **Description** | | | This system feature allows users to be able to change their own password. | | | |
| **Precondition** | | | User must be logged into the system | | | |
| **Trigger** | | | Settings button clicked from the navigation bar in the login page | | | |
| **Post-Condition** | | | Login Page | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1. | Staff  Customer | | Click on the setting hyperlink or icon from any page in the system | | | |
| 2. | BSMS | | Display pop up with change password option | | | |
| 3. | Staff  Customer | | Click on Change password option | | | |
| 4. | BSMS | | Displays Change Password page with the following password fields:   * Previous Password * New Password * Confirm Password * Change Password Button | | | |
| 5. | Staff  Customer | | Input Correct information into the passwords fields | | | |
| 6. | BSMS | | Change logged user password and redirect to login page to relogin with their new user detail. | | | |
|  |  | |  | | | |
| **Alternative flows** | | | | | | |
| **AT1** | |  | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | Customer  Staff | | Confirm password field does not match | | |
| 2.2 | | Customer  Staff | | User current password is not correct | | |

**Business Rules**

|  |  |
| --- | --- |
| **#** | **Rule Description** |
| PMBR01 | Password fields must contain at-least a number |
| PMBR02 | Password text must be longer than 5 length |

#### Edit Profile

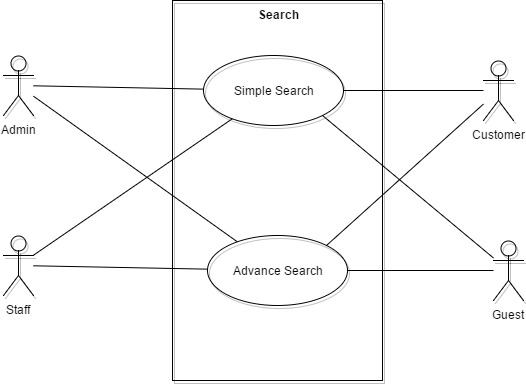
#### 

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | PM01 | | **Version** | 1.0 |
| **Use Name** | | | Edit Profile | | | |
| **Actor** | | | Staff , Customer | | | |
| **Description** | | | this feature allows users to be able to edit their own information | | | |
| **Precondition** | | | User is logged in to the system with correct account information | | | |
| **Trigger** | | | Edit Profile button Clicked | | | |
| **Post-Condition** | | | Profile Detail Page | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1. | Staff  Customer | | Click on the setting hyperlink or icon from any page in the system | | | |
| 2. | BSMS | | Display a pop up with the following hyper link and Icons:   * Change Password * View Profile * Edit Profile | | | |
| 3. | Staff  Customer | | Click on the Edit profile hyperlink or icon | | | |
| 4. | BSMS | | Display Edit Profile page with pre-populated values | | | |
| 5. | Staff  Customer | | Input all *(\*)* Fields correctly and click on the Edit Profile Button | | | |
| 6. | BSMS | | Update user information and show a popup message to inform user that profile have been updated. | | | |
|  |  | |  | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | ***(\*)* field are empty or are invalid** | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Display Edit profile page with error message saying “(\*) fields are required and must be valid” | | |

**Business Rules**

|  |  |
| --- | --- |
| **#** | **Rule Description** |
| PMBR01 | First name, last name, middle name cannot be null and length must be greater than 2 |

### System Search

 **Figure 2.3.9**

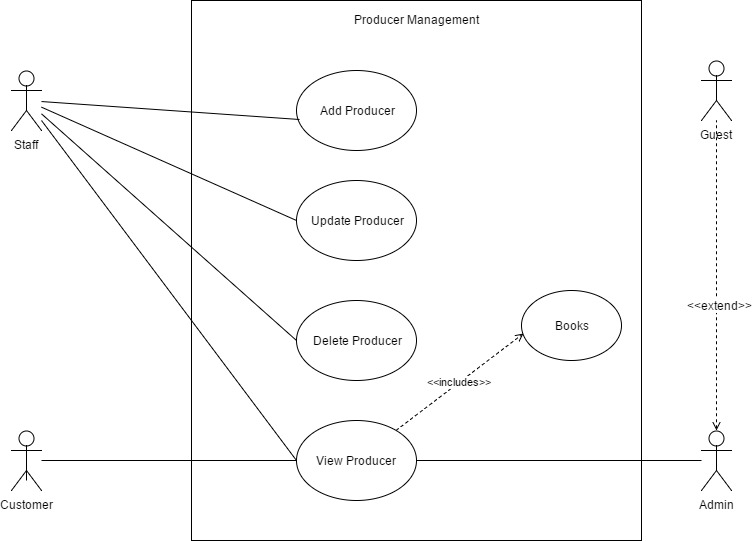
#### Simple Search

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | **SR01** | **Version** | 1.0 |
| **Use Name** | | Simple Search | | |
| **Actor** | | Admin, Customer, Staff, Guest | | |
| **Description** | | The allow user to search books through the search box on the navigation bar from any page | | |
| **Precondition** | | Users of any type is browsing through our web application | | |
| **Trigger** | | N/A | | |
| **Post-Condition** | | Search Result Page | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1. | Admin  Customer  Staff  Guest | 1. Type a book name in the search functionality e.g. (“Lord of Ring”) user typed “Lord” or “Ring”. 2. Select book category or leave it in all category 3. click the search icon | | |
| 2. | BSMS | Display search result page with book that match the given criteria | | |
|  |  |  | | |

#### Advance Search

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | **SR02** | **Version** | 1.0 |
| **Use Name** | | Advance Search | | |
| **Actor** | | Admin, Customer, Staff, Guest | | |
| **Description** | | The allow user to search books or any other item using information’s like ISBN, Author and many other criteria | | |
| **Precondition** | | Users of any type is browsing through our web application | | |
| **Trigger** | | N/A | | |
| **Post-Condition** | | Search Result Page | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1. | Admin  Customer  Staff  Guest | User clicked the advance search hyperlink in the navigation bar from any page | | |
| 2. | BSMS | Displays advance search page with the following fields or more.   * Name * Producer * Author * Category * Genre * Search type (“Author”, “Book”, “Category” …. etc) * Search Button | | |
| 3. | Admin  Customer  Staff  Guest | User input their search criteria correctly. | | |
| 4. | BSMS | Display search result page | | |
|  |  |  | | |

### Producer Management



**Figure 2.3.10**

#### Add Producer

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | PRM01 | | **Version** | 1.0 |
| **Use Name** | | | Add Producer | | | |
| **Actor** | | | Staff | | | |
| **Description** | | | In order to have Producer in our system, the admin might need to add one or more Producer so that each book can have more than or atleast one producer. | | | |
| **Precondition** | | | An admin needs to be logged in to the system with valid account information | | | |
| **Trigger** | | | Add Producer button click | | | |
| **Post-Condition** | | | Producer List Page | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Staff | | Hovers on the Producer from the navigation bar and then clicked on the add Producer button. | | | |
| 2 | BSMS | | Display Add Producer page with the following field:  - Name  -Description  -Upload Thumbnail  -Add Producer Button | | | |
| 3 | Staff | | 1. Fills in correct registration data in the (\*) fields 2. Click the Add Producer button | | | |
| 4 | BSMS | | Validate Producer information, if Producer fields are correct in a normal case. Producer addition success Information message is shown to Admin. | | | |
|  |  | |  | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | Alternative flow, should incase an error occur in the program, if any of the (\*) fields is empty or contains incorrect input. | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Display Add Producer page with message “(\*) fields are required and must contain valid values” | | |
|  | |  | |  | | |

**Business Rules**

|  |  |
| --- | --- |
| **#** | **Rule Description** |
| PRCBR01 | Producer name must not be null and length must be greater than or equals 3 |
| PRCBR02 | Producer Image is not optional. |

#### Delete Producer

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | PRM02 | | **Version** | 1.0 |
| **Use Name** | | | Delete Producer | | | |
| **Actor** | | | Staff | | | |
| **Description** | | | This functionality allow previously created Producer to be deleted from the system when they are no more needed. | | | |
| **Precondition** | | | A Staff needs to be logged in to the system with valid account information | | | |
| **Trigger** | | | Click the delete button on any Producer | | | |
| **Post-Condition** | | | N/A | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Staff | | Click on the Delete Producer button in the Producer List page | | | |
| 2 | BSMS | | Show delete confirm message | | | |
| 3 | Staff | | Staff click “Ok” in the confirm message box | | | |
| 4 | BSMS | | Delete Producer from the system. | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | **The Staff click cancel on the confirm message box shown on the screen** | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Dispose the action | | |
|  | |  | |  | | |

#### Update Producer

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case ID** | | | PRM03 | | **Version** | 1.0 |
| **Use Name** | | | Update Producer | | | |
| **Actor** | | | Staff | | | |
| **Description** | | | This functionality allow previously created Producer to be edited if there exist any wrong information. | | | |
| **Precondition** | | | A staff needs to be logged in to the system with valid account information | | | |
| **Trigger** | | | Update Button | | | |
| **Post-Condition** | | | Producer List page | | | |
| **Main flows** | | | | | | |
| **Step** | **Actor** | | **Action** | | | |
| 1 | Staff | | Click on edit button on top any Producer in the Producer list page. | | | |
| 2 | BSMS | | Display update page with pre-populated data values that belongs to the Producer update button was clicked on. | | | |
| 3 | Staff | | Staff change desired fields then click the update button | | | |
| 4 | BSMS | | Display a confirm message to be sure, the Staff really want to update this Producer “Are you sure you want to update this Producer information” | | | |
| 5 | Staff | | Click ok on the confirmation message | | | |
| 6 | BSMS | | Update Producer Information as update by the Staff. | | | |
| **Alternative flows** | | | | | | |
| **AT1** | | **The Staff click cancel on the confirm message box shown on the screen** | | | | |
| **Step** | | **Actor** | | **Action** | | |
| 2.1 | | BSMS | | Dispose the action | | |
| **AT2** | | **The Staff Click cancel before clicking on the Update button on the page** | | | | |
|  | | BSMS | | Redirect back to Producer list page | | |

|  |  |
| --- | --- |
| **#** | **Rule Description** |
| PUBR01 | Producer name must not be null and length must be greater than or equals 3 |
| PUBR02 | Producer Image is not optional. |

#### View Producer

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | PRM04 | **Version** | 1.0 |
| **Use Name** | | View Producer | | |
| **Actor** | | Admin, Staff, Customers, Guest | | |
| **Description** | | This allows users of any access level to view categories so that they can view books under this Producer. | | |
| **Precondition** | | An admin needs to be logged in to the system with valid account information | | |
| **Trigger** | | Click Producer button in the navigation bar from any page | | |
| **Post-Condition** | | Producer List page | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1 | Admin  Customer  Staff  Guest | Click on Producer from the navigation bar | | |
| 2 | BSMS | Display list of categories with buttons under each one of them “View glyph icon ” or “View” | | |

#### View Books

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case ID** | | PRM05 | **Version** | 1.0 |
| **Use Name** | | View Books | | |
| **Actor** | | Admin, Staff, Guest, Customer | | |
| **Description** | | This functionality allow previously created Producer to be edited if there exist any wrong information. | | |
| **Precondition** | | An admin needs to be logged in to the system with valid account information | | |
| **Trigger** | | Click on Glyph icon view or “View” button | | |
| **Post-Condition** | | Book List page | | |
| **Main flows** | | | | |
| **Step** | **Actor** | **Action** | | |
| 1 | Admin, Staff, Guest, Customer | Click on Glyph icon view or “View” button | | |
| 2 | BSMS | Display Book list page with all book within the selected Producer | | |

## Non-functional Specification

### Reliability

BSMS should always be available, incase if there is any causalities the Web app should come back online with short period of time.

Critical maintenance that requires the server to completely shut down should be done in the mid-night as only few users tend to use the system in the night.

### Performance Requirements

Code optimization should be done frequently in-order for the system to run faster as more and more user come into the system.

#### Availability

Our web application should be available 24/7, which means the application only goes inactive when the server goes down for critical maintenance, which should be done only times in a year as scheduled.

#### Response time

Response time per request should not be more than 1 minute per request made unless there is an internal error in the web application, or the data users is looking for cannot be processed on our server

### User Interface

It is required that the user interface of our web application to be highly interactive and friendly, so that even a dummy can use our application as a guest or customer.

This web application will also be design with the mobile approach in mind (The Mobile First User Interface approach.) should be applied in our system, in such a way that our system understands the type of device E-U is viewing from and automatically adapt to the size of E-U screen.

### Usability

System should be easy to in such a way that there is no need for any external training before E-U will be able to use the system

Friendly and modernize UI, Embossed text for clear view of the system.

English language as a world universal language will be implemented

### Security Requirements

BSMS Web Application should implement MD5 security features in order to protect the system from potential attacks.

BSMS application should not be venerable to SQL injection attack

### Hardware/Software Requirements

To access our web application you will need the following tools:

Browser of your choice: Preferably

For PC Users:

* Microsoft Edge

For Mobile Users:

* Microsoft Edge

# Software Architecture and Design

# PURPOSE

This document is proposed with intent of describing the technical design of our bookstore Management system, mainly to give development team an overall overview of the system’s architecture, thus bringing them to a level of enlightenment of how the system should work and how to implement it. This section contains the following

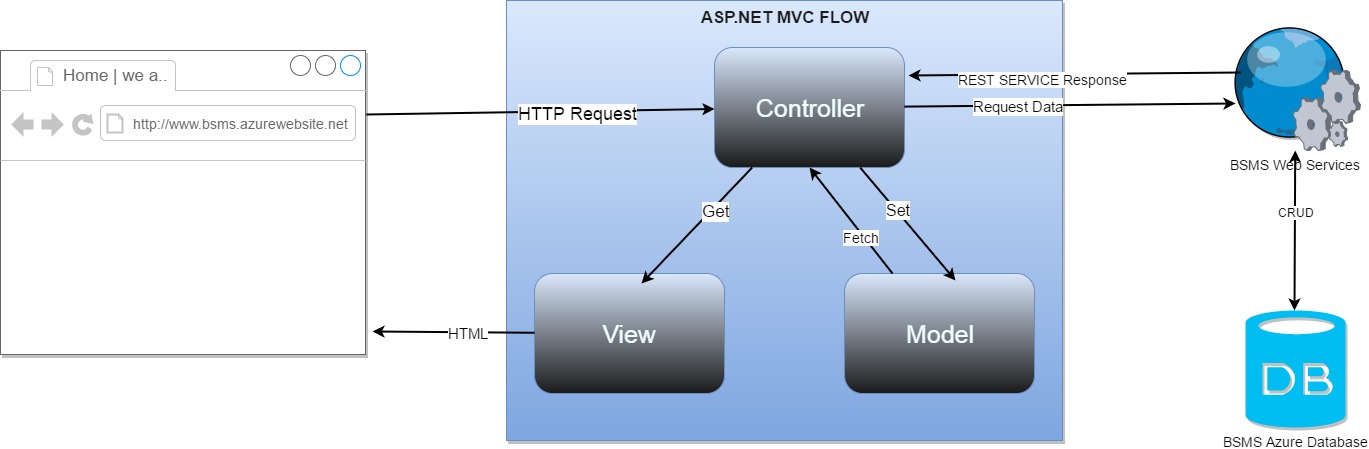
* System Architecture Design
* Database Design
* Detail Design

# ARCHITECTURE DESIGN

## Choice of Architecture Design

### Our System Architecture

BSMS web application uses the MVC (Model-View and Controller) architecture together with ASP.NET Web Services (XML,JSON), which has been one of the best software design pattern this days, This will allow our code to be more readable, more understandable and editable than the native coding pattern.



* **Models :** Model objects are the parts of the application that implement the logic for the application's data domain. Often, model objects retrieve and store model state in a database. For example, a Product object might retrieve information from a database, operate on it, and then write updated information back to a Products table in a SQL Server database.

In small applications, the model is often a conceptual separation instead of a physical one. For example, if the application only reads a dataset and sends it to the view, the application does not have a physical model layer and associated classes. In that case, the dataset takes on the role of a model object.

In Our system the model will be referencing to the BSMS web service to complete database related tasks.

* **Views :** Views are the components that display the application's user interface (UI). Typically, this UI is created from the model data. An example would be an edit view of a Products table that displays text boxes, drop-down lists, and check boxes based on the current state of a Product object..
* **Controllers :** This serves as the system handler, it handles user input and what the system do with them. When request is send from the client side the asp.net send a request to the controller, which after the controller will find the equivalent get method and them render the called view.

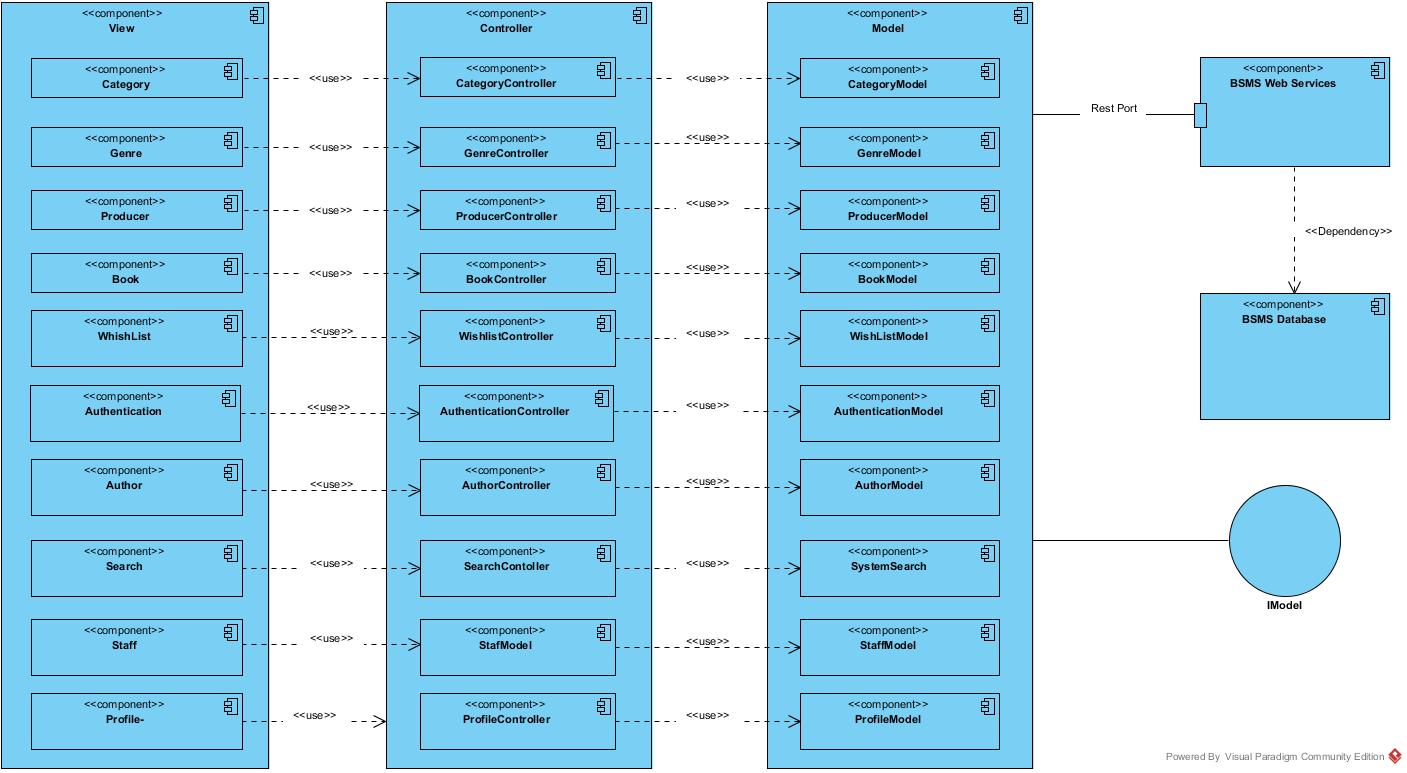
 Controllers are the components that handle user interaction, work with the model, and ultimately select a view to render that displays UI. In an MVC application, the view only displays information; the controller handles and responds to user input and interaction. For example, the controller handles query-string values, and passes these values to the model, which in turn might use these values to query the database**.**

* **Web Service**: This is the service that connects to the database and fetches data for the controller to make use of; CRUD methods will be implemented using this web service.

### FEATURES OF THE ASP.NET MVC FRAMEWORK

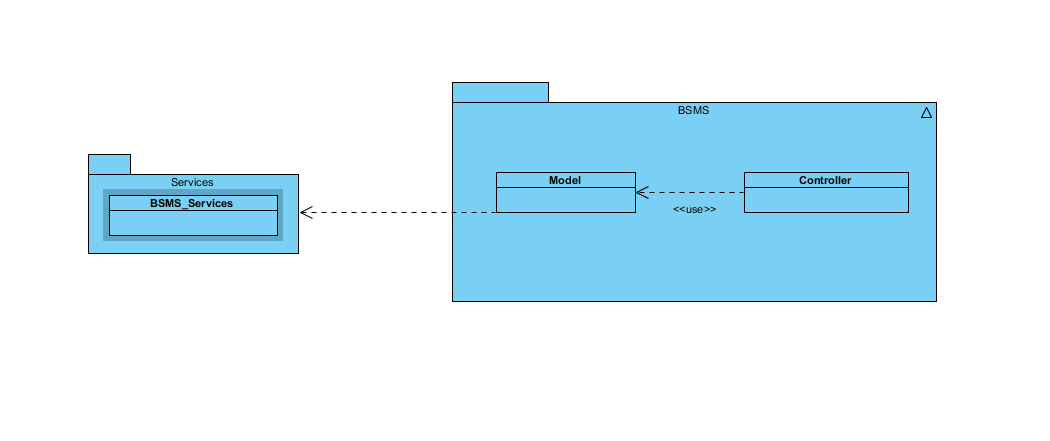
* Separation of application tasks (input logic, business logic, and UI logic), testability, and test-driven development (TDD). All core contracts in the MVC framework are interface-based and can be tested by using mock objects, which are simulated objects that imitate the behavior of actual objects in the application. You can unit-test the application without having to run the controllers in an ASP.NET process, which makes unit testing fast and flexible. You can use any unit-testing framework that is compatible with the .NET Framework.
* An extensible and pluggable framework. The components of the ASP.NET MVC framework are designed so that they can be easily replaced or customized. You can plug in your own view engine, URL routing policy, action-method parameter serialization, and other components. The ASP.NET MVC framework also supports the use of Dependency Injection (DI) and Inversion of Control (IOC) container models. DI enables you to inject objects into a class, instead of relying on the class to create the object itself. IOC specifies that if an object requires another object, the first objects should get the second object from an outside source such as a configuration file. This makes testing easier.
* Extensive support for ASP.NET routing, which is a powerful URL-mapping component that lets you build applications that have comprehensible and searchable URLs. URLs do not have to include file-name extensions, and are designed to support URL naming patterns that work well for search engine optimization (SEO) and representational state transfer (REST) addressing.
* Support for using the markup in existing ASP.NET page (.aspx files), user control (.ascx files), and master page (.master files) markup files as view templates. You can use existing ASP.NET features with the ASP.NET MVC framework, such as nested master pages, in-line expressions (<%= %>), declarative server controls, templates, data-binding, localization, and so on.
* Support for existing ASP.NET features. ASP.NET MVC lets you use features such as forms authentication and Windows authentication, URL authorization, membership and roles, output and data caching, session and profile state management, health monitoring, the configuration system, and the provider architecture.

## Component Diagram



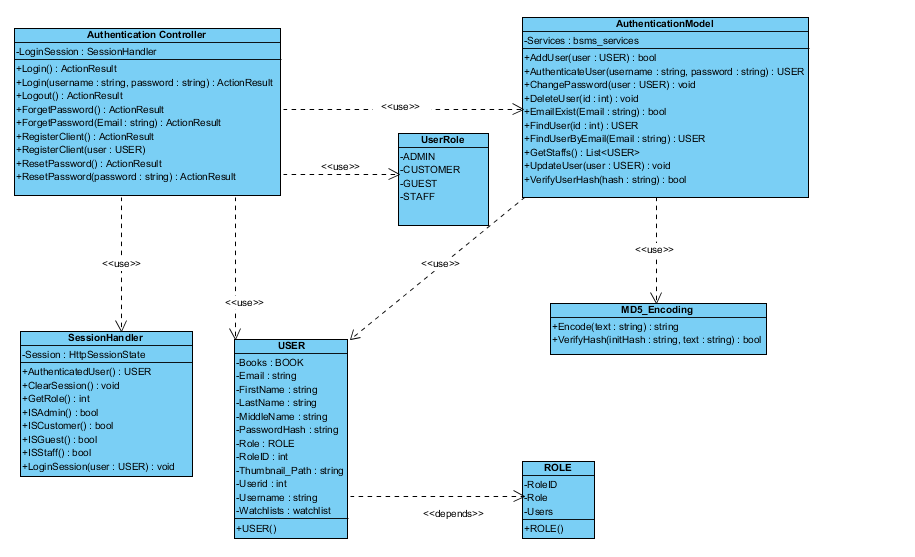
# DETAILED DESIGN

## Common Design



## Authentication Management

### Class Diagram



### Class Description

|  |  |  |
| --- | --- | --- |
| Class | Dependency | Description |
| User | Role | The user class holds all user data and information including all the usernames and passwords in the system, this is known as the BSMS user recognition methodology. |
| Role | N/A | This class is intended to hold unique set of roles for users, since the user uses this class as dependency, user has a role, which is how the system recognise the type of user coming into the system |
| AuthenticationController | N/A | Controller class which extends from the base class system.web.muvc controller, this class handles all http request including http post and get, delete method. |
| SessionHandler | N/A | This class acts as session assistant, in other words this class handles all session related activities |
| AuthenticationModel | USER | This is a class that handles all operations that involves Authentication including CRUD Operations, thus this class have direct connection to BSMS SOAP Webservices. |
| MD5\_Encoding | N/A | This class handles encoding, like encode, decode |
| UserRole | N/A | This class acts like enum class, as this keeps static variables of each types of users that exist in the system |

#### User

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | User | | | |
| **Description** | User data, this class holds all the user information including the password and username | | | |
| **Base Class** | Authentication | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | User.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Attributes** | Name | Type | Description | |
| Name | String | N/A | |
| Username | String | Users username is contained in this attribute | |
| PasswordHash | String | This contains 64 digit MD5 encrypted password | |
| Email | String |  | |
| RoleID | int | This hold the role of user in number format ex. (1:”Staff” , 2:”Admin”) | |
|  | Thumbnail | String | This hold the thumbnail path in the system | |
| **Methods** | Name | Input | Output | Description |
| Authentication | Username  Password | User | This function check if user is valid or not, returns null if not valid and user information if valid |
| LoggedOn | N/A | Boolean | This function is intended to check if any user is logged on to the system |

#### Role

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Role | | |
| **Description** | This contain all the user roles that exists in the system | | |
| **Base Class** | N/A | | |
| **Constructor** | Default | | |
| **Prototype** |  | | |
| **Source File** | Role.cs | | |
| **Namespace** | BSMS.Model | | |
| **Attributes** | Name | Type | Description |
| Roleid | int | This is a unique integer value that holds a unique id for each role that exist in the system |
|  | Role | String | Name of this role e.g “Admin” |

#### Authentication Controller

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | AuthenticationController | | | |
| **Description** | Controller class which extends from the base class system.web.muvc controller, this class handles all http request including http post and get, delete method. | | | |
| **Base Class** | System.Web.Mvc.Controller | | | |
| **Constructor** | N/A | | | |
| **Prototype** |  | | | |
| **Source File** | AuthenticationController.cs | | | |
| **Namespace** | BSMS.Controller | | | |
| **Methods** | Name | Input | Output | Description |
| Login | N/A | ActionResult | This method make a request to the Http Get in order to request for a page |
| Login | String, String | ActionResult | This method call makes request to Http Post and calls the Authentication Model class to verify user detail |
| Logout | N/A | ActionResult | On this method call E-U will be logged out of the system and redirect to the login page |
| ForgetPassword | N/A | ActionResult | Http Method call to request forget password page |
| ForgetPassword | String | ActionResult | Http Post method call to act on user input |
| RegisterUser | N/A | ActionResult | Request made to register user or customer page |
| RegisterUser | User | ActionResult | Process new user Http Post method |

#### Session Handler

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | SessionHandler | | | |
| **Description** | This class acts as session assistant, in other words this class handles all session related activities | | | |
| **Base Class** | N/A | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | SessionHandler.cs | | | |
| **Namespace** | BSMS.Session | | | |
| **Attributes** | Name | Type | Description | |
| Session | HttpSessionState | The Session is an instance of the HttpSessionState, which contains the application context Session | |
| **Methods** | Name | Input | Output | Description |
|  | AuthenticatedUser | N/A | USER | This method return the currently logged in user, and returns it’s object |
| ClearSession | N/A | N/A | This method is used to clear application context session |
| GetRole | N/A | Integer | This method will be used to get role of currently loggeduser |
| ISAdmin | N/A | Boolean | This method is mean’t to check if currently logged in user is Admin |
| ISCustomer | N/A | Boolean | This method is mean’t to check if currently logged in user is Customer |
| ISGuest | N/A | Boolean | This method is mean’t to check if there is no logged user in the system |
| ISStaff | N/A | Boolean | This method is mean’t to check if currently logged in user is Staff |
| LoginSession | USER | N/A | This method is mainly used to create a login session, which mean this will be used only when users’ needs to login. |

#### Authentication Model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | AuthenticationModel | | | |
| **Description** | This is a class that handles all operations that involves Authentication including CRUD Operations, thus this class have direct connection to BSMS SOAP Webservices. | | | |
| **Base Class** | N/A | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | Authentication.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Attributes** | Name | Type | Description | |
| Services | bsms\_services | This is a private instance of “BSMS.SERVICES” | |
| **Methods** | Name | Input | Output | Description |
| AddUser | USER | Boolean | This method is use to call the webservices method inorder to add a new user into the system |
| AuthenticateUser | String, String | USER | This method is proposed to check if login credential is correct or not, this return null if username or password is incorrect and User Object otherwise |
| ChangePassword | USER | N/A | This method is used to change user passwords on reset password request |
| DeleteUser | Int | N/A | This static method is used to delete user from BSMS |
| FindUser | Int | USER | This static method return User Object which matches the id of user inpt |
| FindUserByEmail | Int | USER | This method is used to find user by email, since the email address of each user is unique |
| GetStaffs | N/A | List<USER> | This method is used to get all avaliable staff in the system |
| UpdateStaff | USER | N/A | This is used to update staff informations |
| VerifyUserHash | String | Boolean | This method is mainly used to verify hash code of users |

#### MD5\_Encoding

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | MD5\_Encoding | | | |
| **Description** | This class handles encoding, like encode, decode | | | |
| **Base Class** | MD5\_Encoding | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | MD5\_Encoding.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Methods** | Name | Input | Output | Description |
| Encode | String | String | This function mainly encode input string to MD5 encryption |
| VerifyHash | String,String | Boolean | This function is intended to check if user password match its encypted format |

#### UserRole

|  |  |  |  |
| --- | --- | --- | --- |
| Class | UserRole | | |
| **Description** | This class acts like enum class, as this keeps static variables of each types of users that exist in the system | | |
| **Base Class** | UserRole | | |
| **Constructor** | Default | | |
| **Prototype** |  | | |
| **Source File** | UserRole.cs | | |
| **Namespace** | BSMS.Model | | |
| **Attribute** | Name | Type | Description |
| ADMIN | int | This contains role id of Admin users |
| CUSTOMER | int | This contains the role id of customer users. |
|  | GUEST | int | This contains role id of Guest users |
|  | STAFF | int | This contains role id of Staff users |

### Register Customer/Staff

BSMS guest who might have intention of becoming a customer, or Admin who wish to register a new staff in the system, both the guest and the admin will make use of this functionality in order to create an account, when the admin make use of this feature, automatically a Staff will be added into the system and an email notification containing password reset link for the newly added staff, in the other case when a guest uses this feature then they will be registered as customer, and given the ability to login to the system

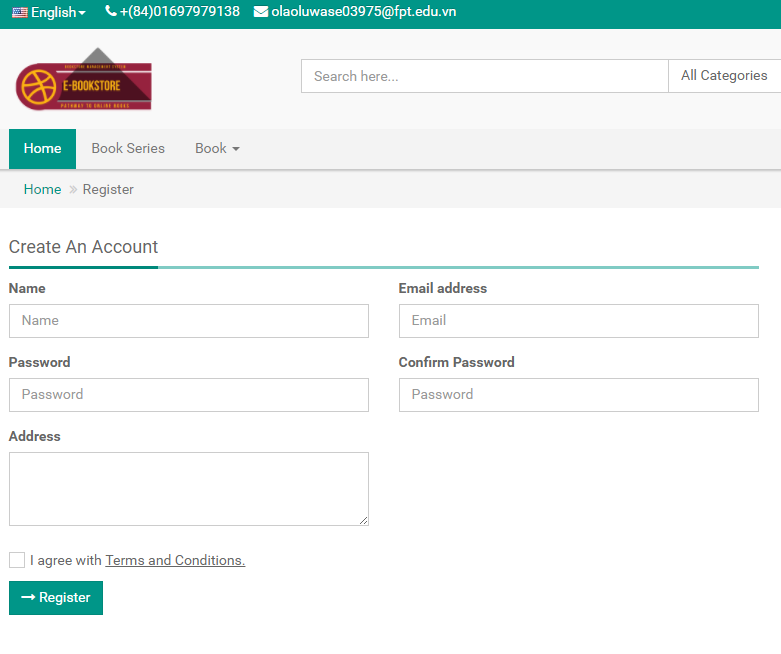
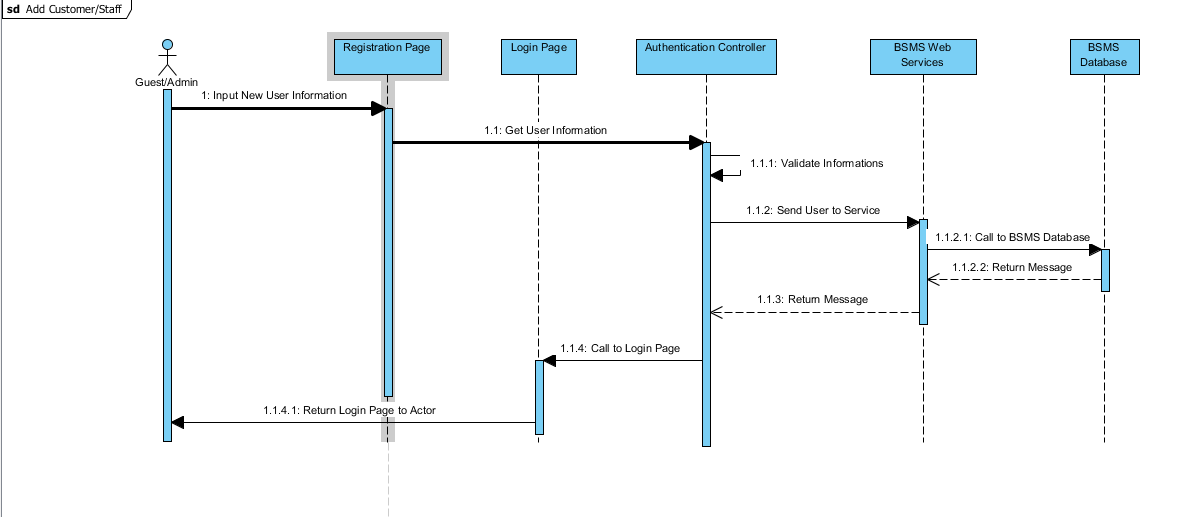


Figure 3.2.3

**Table 4-1:** Register Customer/Staff

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Object/Control Name** | **Type** | **Required** | **Length** | **Description** |
| 1 | Name | Text-Field | Yes | 3-20 | Customer Name |
| 2 | Email | Email-Field | Yes | 7-30 | Customer Email Address |
| 3 | Password | Password-Field | Yes | > 6 | User password |
| 4 | Confirm Password | Password-Field | Yes | > 6 | Password verification |
| 5 | Address | MultiLine-Field | No | Max 50 | Customers Address |
| 6 | cbAccept | CheckBox | N/A | N/A | Check i agree with terms and condition to activate Register button |
| 7 | Register | Button | N/A | N/A | Registration button |

#### Sequence Diagram



### Login

#### Description

Account holder of BSMS who might want to take action on the system, which might be required of him/her to login into the system, this applies to all stockholders of the system apart from the guest, logging in into the system will give the E-U some level of access to the system, assuming the scenario whereby a guest wish to save some book for feature reference, then he/she register an account as described in **3.2.3** after following a normal case, they login into the system to access the feature of wish list.

#### Screen Design

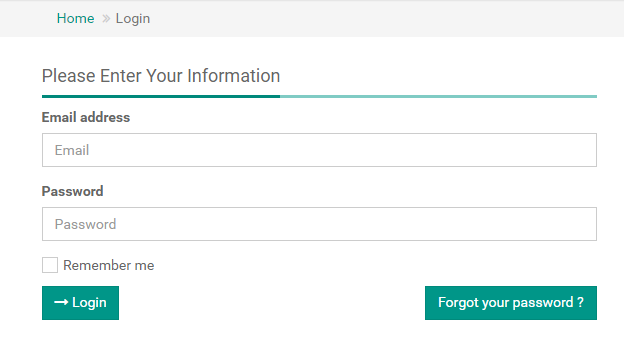
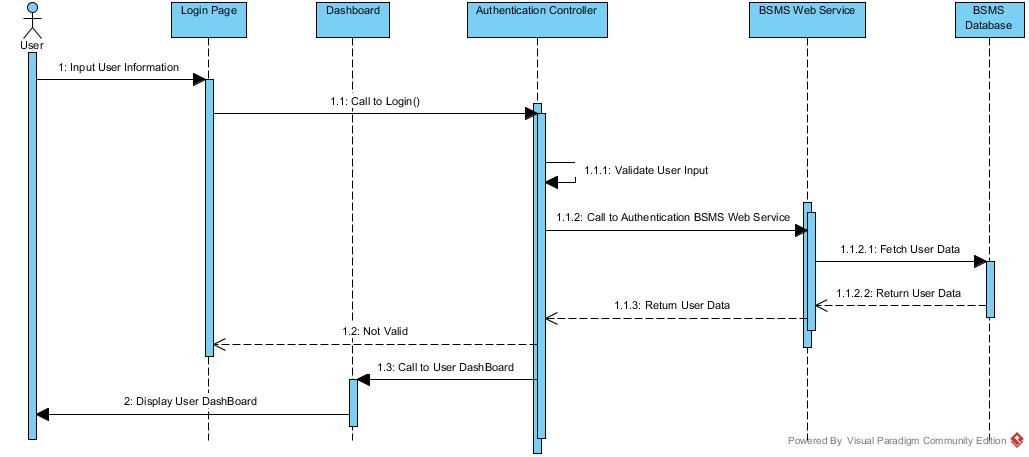


Figure 3.2.3

**Table 4-2:** Login

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Object/Control Name** | **Type** | **Required** | **Length** | **Description** |
| 2 | Email | Email-Field | Yes | 7-30 | Customer Email Address |
| 3 | Password | Password-Field | Yes | > 6 | User password |
| 6 | cbRememberMe | CheckBox | N/A | N/A | Remember me, to keep user detail incase of next login |
| 7 | Login | Button | N/A | N/A | Login button |

#### Sequence Diagram

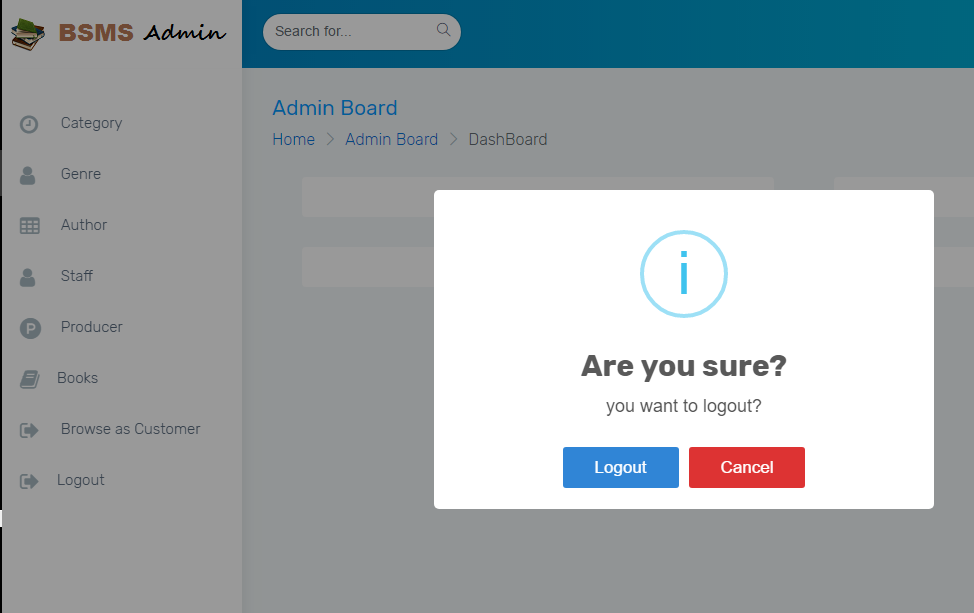


### Logout

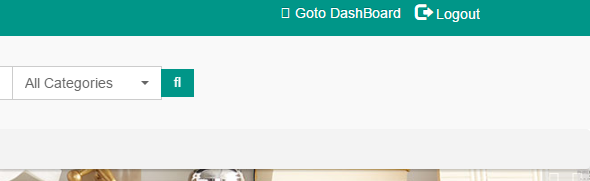
#### Description

Logged user sometime might like to make use of the logout functionality, in order for guest users not to take advantage of their logged session, this feature will be most useful when the E-U uses another use PC/Desktop, in this case there is need to logout.

#### Screen Design

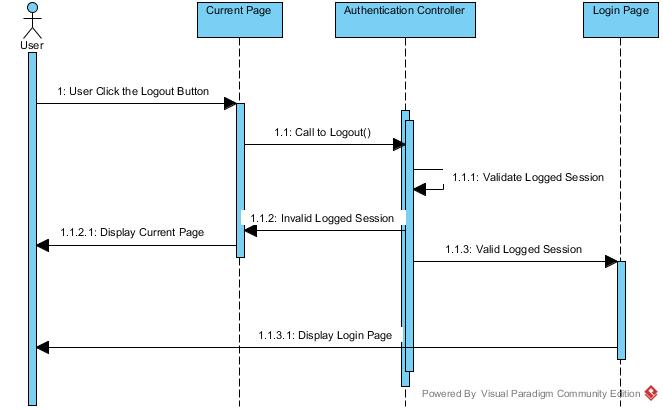


For Admin and Staff users



For customers

#### Sequence Diagram

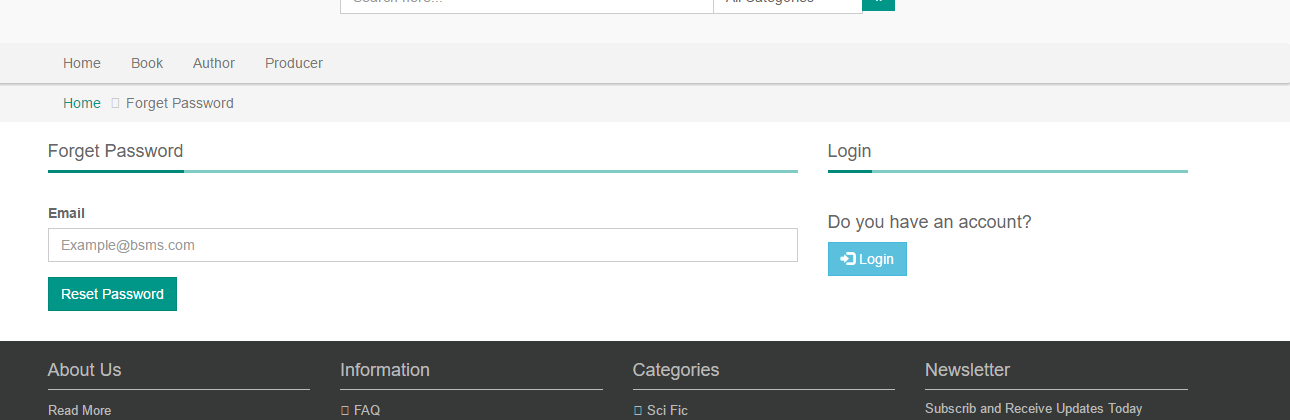


### Forget Password

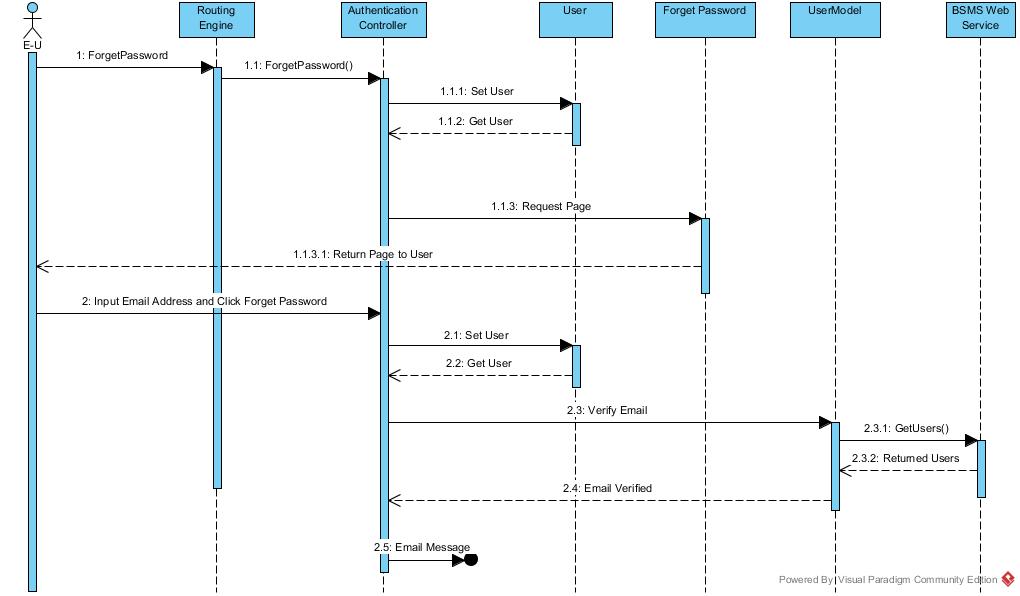
#### Description

BSMS doesn’t expect her E-U to have the brain of computer, in case if any of her users forget their password, BSMS has provide a simple way to recover their password using the same email used while registering, in many case E-U might in one way or the other forget there password, using the forget password feature E-U is allow to reset there password using their email, as email is supposed to be a unique value.

#### Screen Design

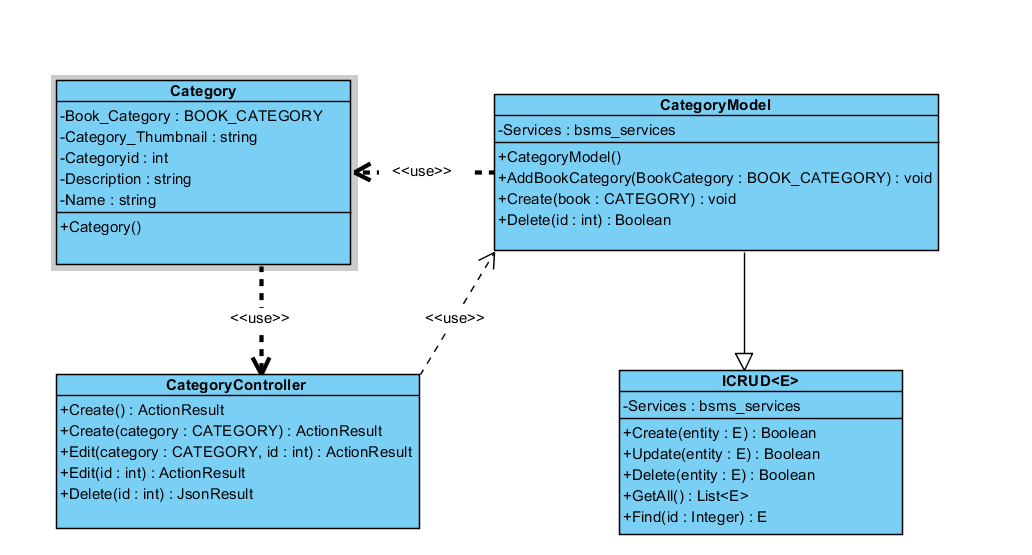


#### Sequence Diagram



## Category Management

### Class Diagram



### Class Description

|  |  |  |
| --- | --- | --- |
| Class | Dependency | Description |
| Category | Role | The category class is a linq mapped entity class that holds all informations regarding category in BSMS. |
| ICRUD | N/A | This Interface is intended to create rules for all model class and sometimes the controller can also make use of this |
| CategoryModel | N/A | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class |
| CategoryController | N/A | This controller class extends from the base class “Controller” which allow request to be mapped to it. |

#### Category

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Category | | |
| **Description** | this class holds all category information including name, and description | | |
| **Base Class** | Object | | |
| **Constructor** | Default | | |
| **Prototype** |  | | |
| **Source File** | Category.cs | | |
| **Namespace** | BSMS.bsms.locahost | | |
| **Attributes** | Name | Type | Description |
| Name | String | N/A |
| Book\_Category | BOOK\_CATEGORY | N/A |
| Category\_Thumbnail | String | This field contains the image of a category |
| CategoryID | Int | This is a unique identifyer for each categories, and this is what serves as key BSMS uses in recorgnising a category |
| Description | String | This field contains description for category |

#### CategoryModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | CategoryModel | | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | | |
| **Base Class** | Object | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | Category.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Attributes** | Name | Type | Description | |
| Services | Bsms\_services | A private instance of bsms soap web services | |
| **Methods** | Name | Input | Output | Description |
| AddBookCategory | BOOK\_CATEGORY | N/A | This method is use to add new book category into BSMS |
| Create | Category | void | This helper method is used to insert new category into the system |
|  | Delete | Int | Boolean | this is a static internal type of method, which allows existing category to be deleted as the name Delete implies |
|  | EditCategory | Category | Boolean | This model method edit information of category and return true if model state is valid, and false otherwise. |
|  | Fliter | Int | Category | This model method takes as input category id and return equivalent category as output if found, return null otherwise |

#### Category Controller

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | CategoryController | | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | | |
| **Base Class** | System.web.controller | | | |
| **Constructor** | N/A | | | |
| **Prototype** |  | | | |
| **Source File** | CategoryController.cs | | | |
| **Namespace** | BSMS.Controller | | | |
| **Methods** | Name | Input | Output | Description |
| Create | N/A | ActionResult | This is serves as http get handler |
| Create | Category | ActionResult | This is a web method that handles user input for create |
|  | Edit | Int | ActionResult | This is a controller web method that handles all http get for Edit functionality |
| Edit | Category, int | ActionResult | This is a controller http post method that handles edit functionalities for categories |
| Index | N/A | ActionResult | http get method that handles list page |
| Delete | Int | JsonResult | http delete method that handles delete request from bsms |

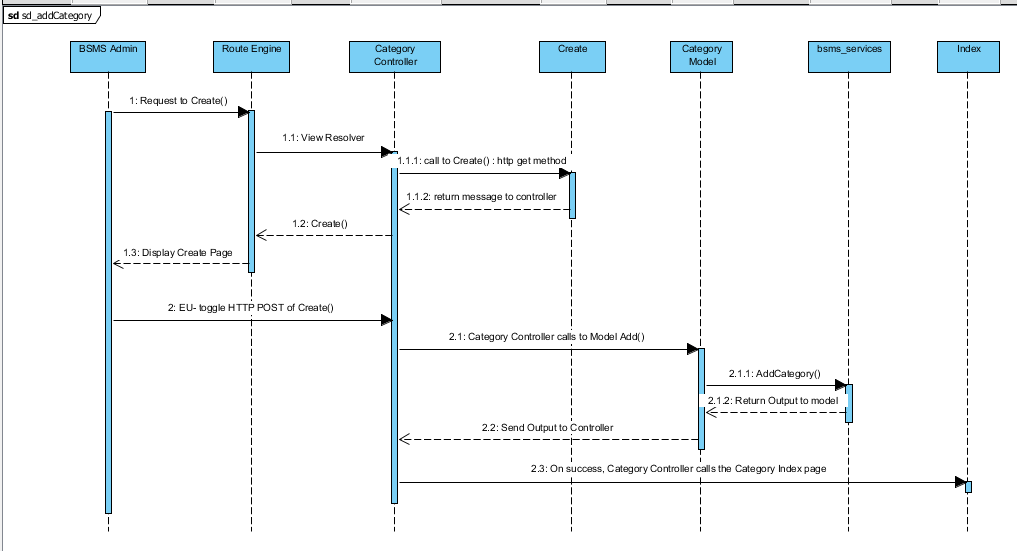
### Add Category

In order to have category in our system, the admin might need to add one or more category so that we can categorize our books.

Adding Category in to the system is only allowed of the admin, which after the staff might add new book under the category.

#### Screen Design

#### Sequence Diagram

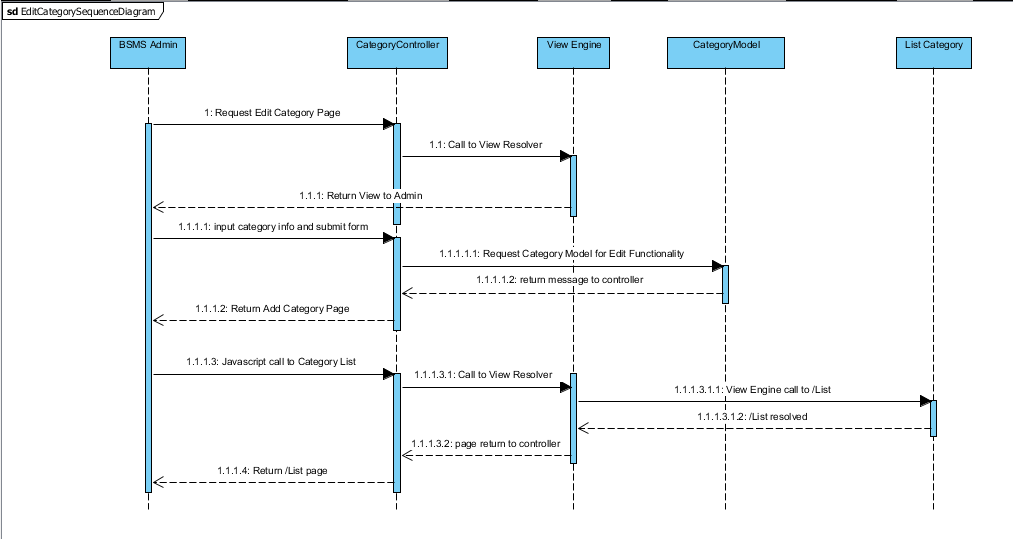


### Edit Category

This functionality allow previously created category to be edited if there exist any wrong information, editing information of category including the category name and other information e.g. changing category name from “Science Fiction” to “Sci Fi”

#### Screen Design

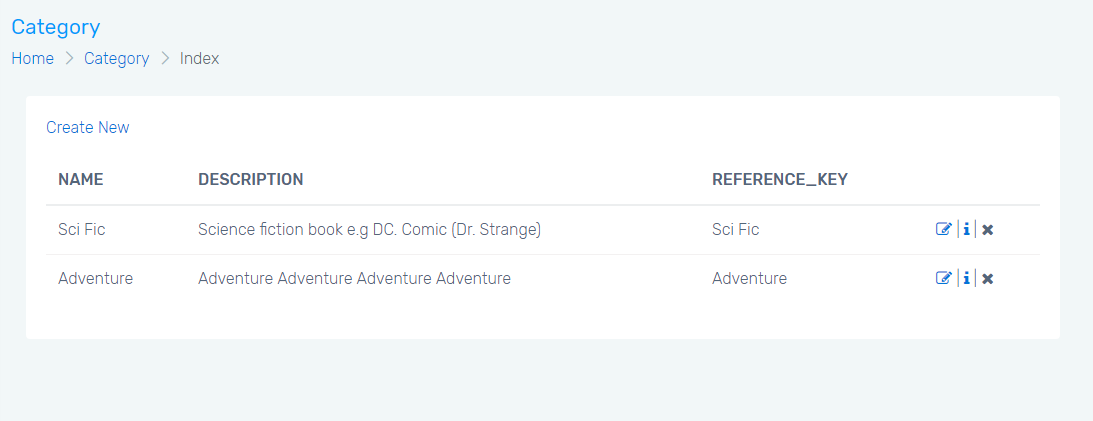
#### Sequence Diagram



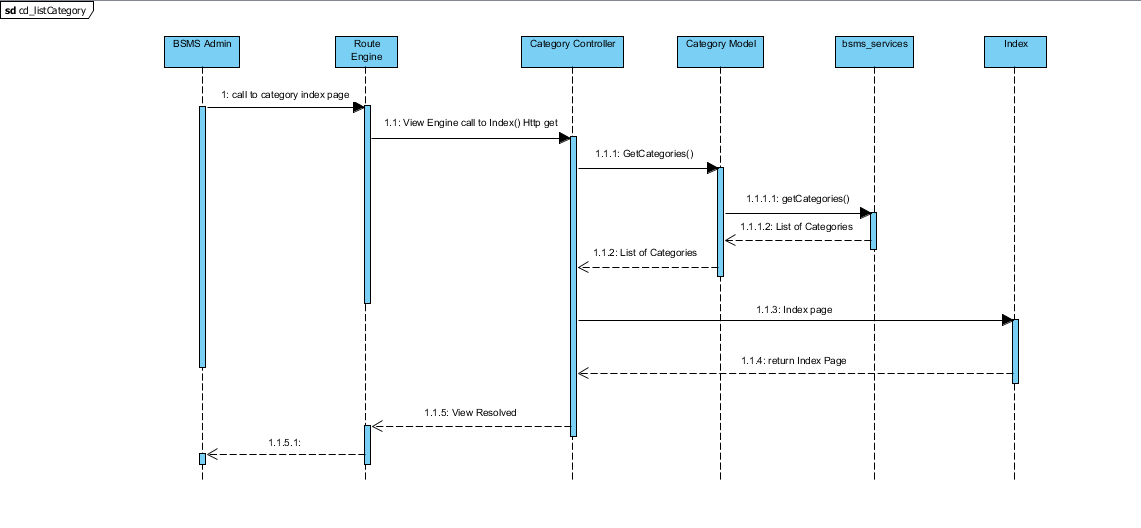
### List Category

The list category feature allow the admin use to view sequence categories that have been added into the system, with each element of the sequence having edit, delete and info icon. On click of each icon triggers equivalent event for each element in that list.

#### Screen Design



#### Sequence Diagram

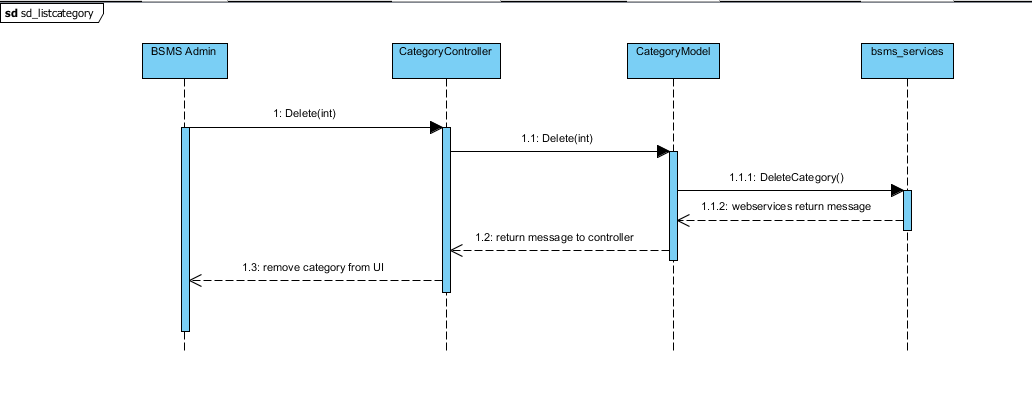


### Delete Category

This feature allow admin users to delete category from the system, this feature is useful in the case where admin user added wrong/duplicate/unnecessary category, this feature will allow admin to delete such category.

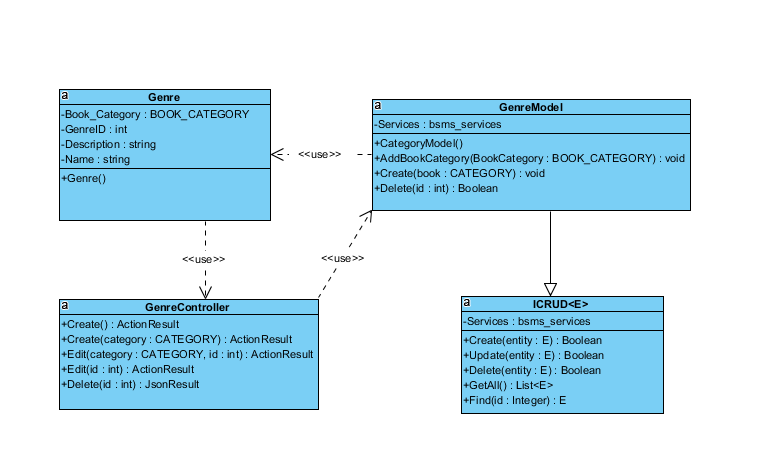
#### Screen Design

#### Sequence Diagram



## Genre Management

### Class Diagram



### Class Description

|  |  |  |
| --- | --- | --- |
| Class | Dependency | Description |
| Genre | N/A | The genre class is a linq mapped entity class that holds all information’s regarding genre in BSMS. |
| ICRUD | N/A | This Interface is intended to create rules for all model class and sometimes the controller can also make use of this |
| GenreModel | N/A | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class |
| GenreController | N/A | This controller class extends from the base class “Controller” which allow request to be mapped to it. |

#### Genre

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Genre | | |
| **Description** | this class holds all genre information including name, and description | | |
| **Base Class** | Object | | |
| **Constructor** | Default | | |
| **Prototype** |  | | |
| **Source File** | Genre.cs | | |
| **Namespace** | BSMS.bsms.localhost | | |
| **Attributes** | Name | Type | Description |
| Name | String | N/A |
| Book\_Genre | BOOK\_GENRE | N/A |
| GenreID | Int | This is a unique identifier for each genre’s, and this is what serves as key BSMS uses in recognising a Genre |
| Description | String | This field contains description for genre |

#### Genre Model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | GenreModel | | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | | |
| **Base Class** | Object | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | GenreModel.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Attributes** | Name | Type | Description | |
| Services | Bsms\_services | A private instance of bsms soap web services | |
| **Methods** | Name | Input | Output | Description |
| AddGenre | BOOK\_CATEGORY | N/A | This method is use to add new book category into BSMS |
| Create | Category | void | This helper method is used to insert new category into the system |
|  | Delete | Int | Boolean | this is a static internal type of method, which allows existing category to be deleted as the name Delete implies |

#### Genre Controller

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | GenreController | | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | | |
| **Base Class** | System.web.controller | | | |
| **Constructor** | N/A | | | |
| **Prototype** |  | | | |
| **Source File** | GenreController.cs | | | |
| **Namespace** | BSMS.Controller | | | |
| **Methods** | Name | Input | Output | Description |
| Create | N/A | ActionResult | This is serves as http get handler |
| Create | Genre | ActionResult | This is a web method that handles user input for create |
|  | Edit | Int | ActionResult | This is a controller web method that handles all http get for Edit functionality |
| Edit | Genre, int | ActionResult | This is a controller http post method that handles edit functionalities for genres |
| Index | N/A | ActionResult | http get method that handles list page |
| Delete | Int | JsonResult | http delete method that handles delete request from bsms |

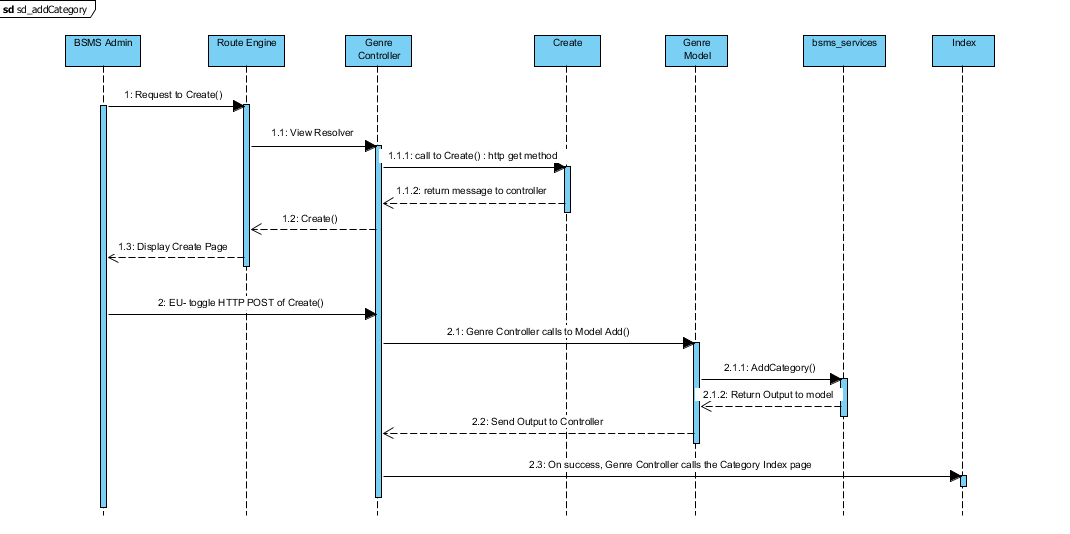
### Add Genre

In order to have genre in our system, the admin might need to add one or more genre so that we can differentiate the different type of book available in the system.

Adding genre in to the system is only allowed of the admin, which after the staff might add new book under the genre.

#### Screen Design

#### Sequence Diagram

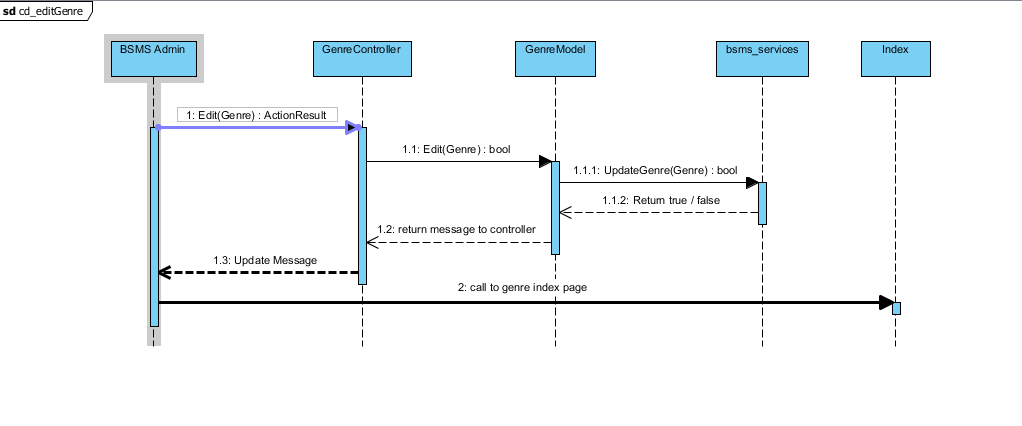


### Edit Genre

This functionality allow previously created genre to be edited if there exist any wrong information, editing information of genre including the genre name and other information e.g. changing genre name from “18+” to “Above 18”

#### Screen Design

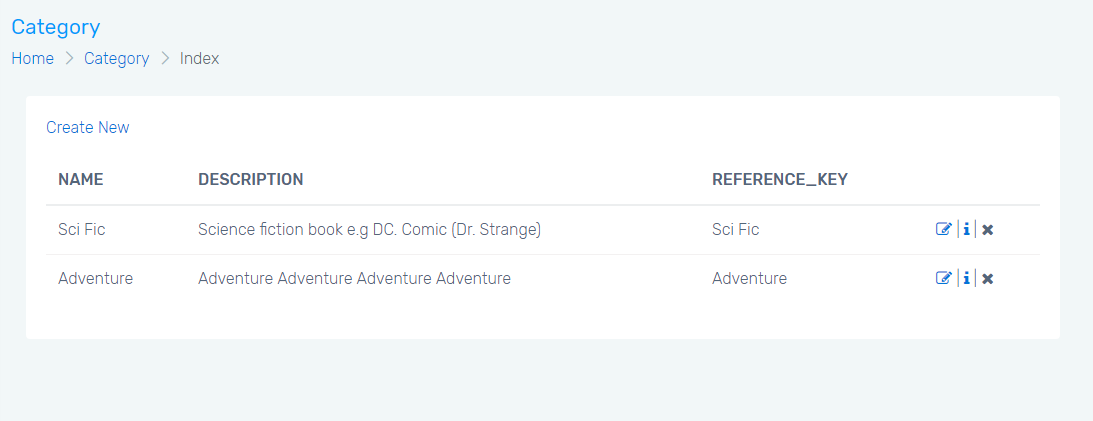
#### Sequence Diagram



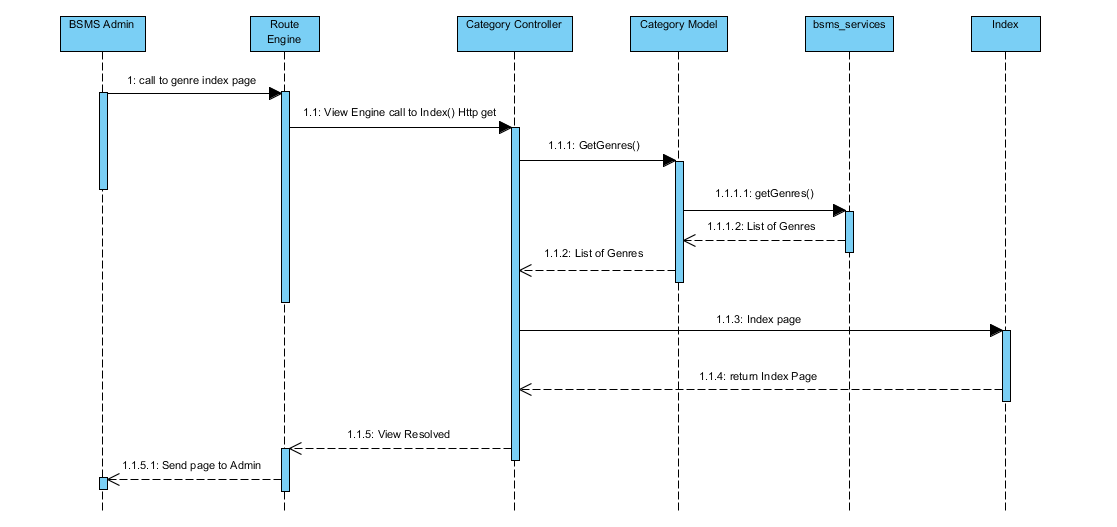
### List Genre

The list category feature allow the admin use to view sequence categories that have been added into the system, with each element of the sequence having edit, delete and info icon. On click of each icon triggers equivalent event for each element in that list.

#### Screen Design



#### Sequence Diagram

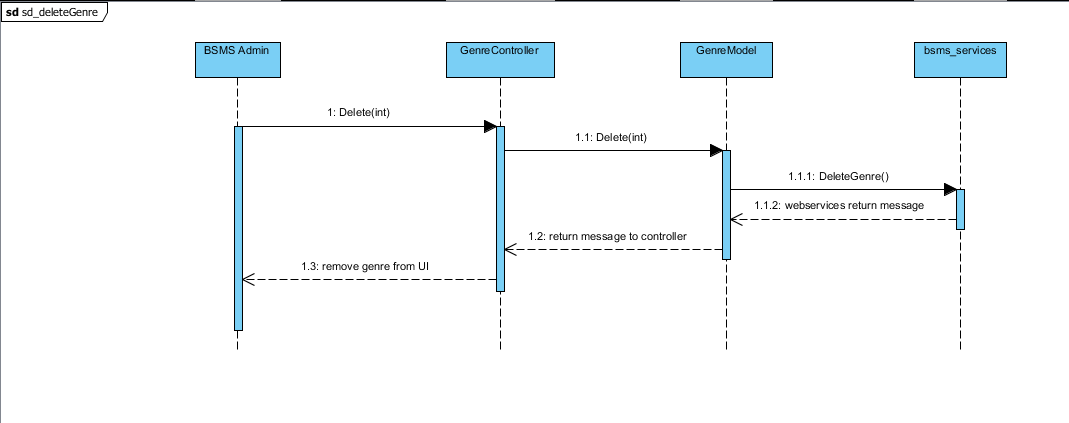


### Delete Genre

This feature allow admin users to delete category from the system, this feature is useful in the case where admin user added wrong/duplicate/unnecessary category, this feature will allow admin to delete such category.

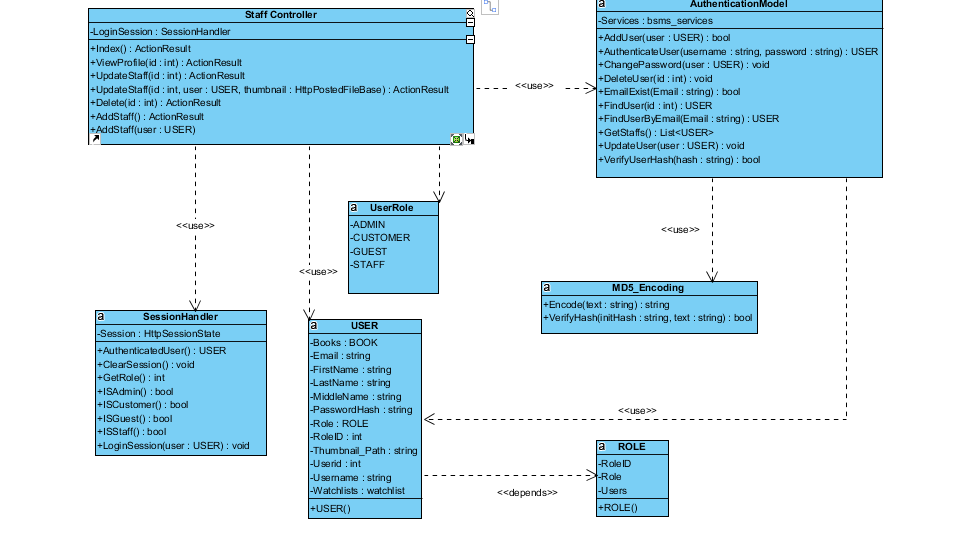
#### Screen Design

#### Sequence Diagram



## Staff Management

### Class Diagram



### Class Description

|  |  |  |
| --- | --- | --- |
| Class | Dependency | Description |
| User | Role | The user class holds all user data and information including all the usernames and passwords in the system, this is known as the BSMS user recognition methodology. |
| Role | N/A | This class is intended to hold unique set of roles for users, since the user uses this class as dependency, user has a role, which is how the system recognise the type of user coming into the system |
| AuthenticationController | N/A | This is a controller class that handles all operations that involving http Operations, once any corresponding http method is call the route engine fires up this controller by requesting it for action. |
| SessionHandler | N/A | This is an handler class that handles all session in the system, this class save and retrieves browsing session. |
| AuthenticationModel | USER | This is a class that handles all operations that involves Authentication including CRUD Operations, thus this class have direct connection to BSMS SOAP Webservices. |
| MD5\_Encoding | N/A | This class handles system encryption, Md5 password encoding and decoding |
| UserRole | N/A | This class contains user role dynamic attribute which save the id of each role. |

#### User

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | User | | | |
| **Description** | User data, this class holds all the user information including the password and username | | | |
| **Base Class** | Object | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | User.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Attributes** | Name | Type | Description | |
| Name | String | N/A | |
| Username | String | Users username is contained in this attribute | |
| PasswordHash | String | This contains 64 digit MD5 encrypted password | |
| Email | String |  | |
| RoleID | int | This hold the role of user in number format ex. (1:”Staff” , 2:”Admin”) | |
|  | Thumbnail | String | This hold the thumbnail path in the system | |
| **Methods** | Name | Input | Output | Description |
| Authentication | Username  Password | User | This function check if user is valid or not, returns null if not valid and user information if valid |
| LoggedOn | N/A | Boolean | This function is intended to check if any user is logged on to the system |

#### Role

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Role | | |
| **Description** | This contain all the user roles that exists in the system | | |
| **Base Class** | Object | | |
| **Constructor** | Default | | |
| **Prototype** |  | | |
| **Source File** | Role.cs | | |
| **Namespace** | BSMS.Model | | |
| **Attributes** | Name | Type | Description |
| Roleid | int | This is a unique integer value that holds a unique id for each role that exist in the system |
| Role | String | Role of this role e.g. (“Admin, Staff” .... ) |

#### Staff Controller

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | StaffController | | | |
| **Description** | This class controls everything they deals with User and Roles | | | |
| **Base Class** | System.Web.Mvc.Controller | | | |
| **Constructor** | N/A | | | |
| **Prototype** |  | | | |
| **Source File** | AuthenticationController.cs | | | |
| **Namespace** | BSMS.Controller | | | |
| **Methods** | Name | Input | Output | Description |
| Index | N/A | ActionResult | This method make a request to the Http Get in order to request for a page |
| ViewProfile | int | ActionResult | This method call makes request to Http Post and calls the Authentication Model class to fetch user detail |
| UpdateStaff | N/A | ActionResult | On this method call E-U will be logged out of the system and redirect to the login page |
| UpdateStaff | N/A | ActionResult | Http Method call to request forget password page |
| Delete | String | ActionResult | Http Post method call to act on user input |
| AddStaff | N/A | ActionResult | Request made to register user or customer page |
| AddStaff | User | ActionResult | Process new user Http Post method |

#### Session Handler

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | SessionHandler | | | |
| **Description** | This class acts as session assistant, in other words this class handles all session related activities | | | |
| **Base Class** | N/A | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | SessionHandler.cs | | | |
| **Namespace** | BSMS.Session | | | |
| **Attributes** | Name | Type | Description | |
| Session | HttpSessionState | The Session is an instance of the HttpSessionState, which contains the application context Session | |
| **Methods** | Name | Input | Output | Description |
|  | AuthenticatedUser | N/A | USER | This method return the currently logged in user, and returns it’s object |
| ClearSession | N/A | N/A | This method is used to clear application context session |
| GetRole | N/A | Integer | This method will be used to get role of currently loggeduser |
| ISAdmin | N/A | Boolean | This method is mean’t to check if currently logged in user is Admin |
| ISCustomer | N/A | Boolean | This method is mean’t to check if currently logged in user is Customer |
| ISGuest | N/A | Boolean | This method is mean’t to check if there is no logged user in the system |
| ISStaff | N/A | Boolean | This method is mean’t to check if currently logged in user is Staff |
| LoginSession | USER | N/A | This method is mainly used to create a login session, which mean this will be used only when users needs to login. |

#### Authentication Model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | AuthenticationModel | | | |
| **Description** | This is a class that handles all operations that involves Authentication including CRUD Operations, thus this class have direct connection to BSMS SOAP Webservices. | | | |
| **Base Class** | N/A | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | Authentication.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Attributes** | Name | Type | Description | |
| Services | bsms\_services | This is a private instance of “BSMS.SERVICES” | |
| **Methods** | Name | Input | Output | Description |
| AddUser | USER | Boolean | This method is use to call the webservices method inorder to add a new user into the system |
| AuthenticateUser | String, String | USER | This method is proposed to check if login credential is correct or not, this return null if username or password is incorrect and User Object otherwise |
| ChangePassword | USER | N/A | This method is used to change user passwords on reset password request |
| DeleteUser | Int | N/A | This static method is used to delete user from BSMS |
| FindUser | Int | USER | This static method return User Object which matches the id of user inpt |
| FindUserByEmail | Int | USER | This method is used to find user by email, since the email address of each user is unique |
| GetStaffs | N/A | List<USER> | This method is used to get all avaliable staff in the system |
| UpdateStaff | USER | N/A | This is used to update staff informations |
| VerifyUserHash | String | Boolean | This method is mainly used to verify hash code of users |

#### MD5\_Encoding

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | MD5\_Encoding | | | |
| **Description** | This class handles encoding, like encode, decode | | | |
| **Base Class** | N/A | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | MD5\_Encoding.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Methods** | Name | Input | Output | Description |
| Encode | String | String | This function mainly encode input string to MD5 encryption |
| VerifyHash | String,String | Boolean | This function is intended to check if user password match its encypted format |

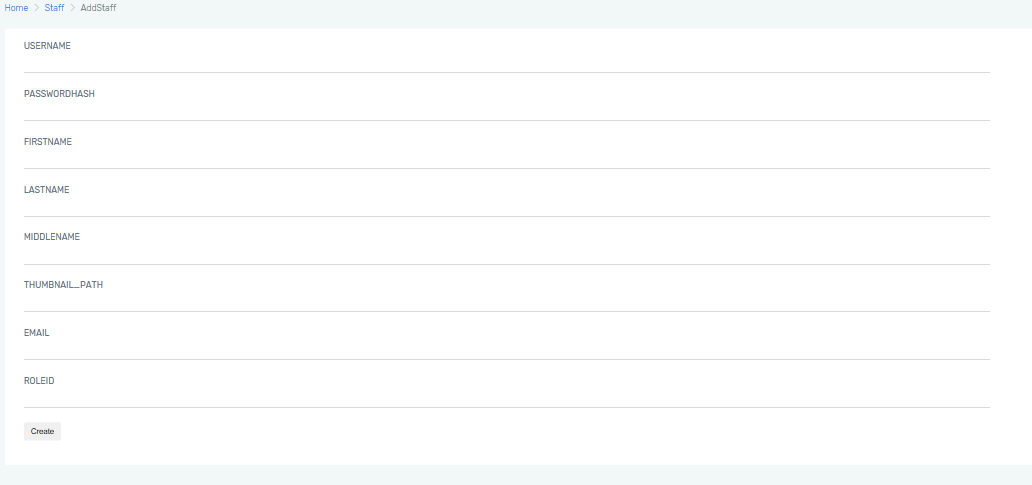
#### UserRole

|  |  |  |  |
| --- | --- | --- | --- |
| Class | UserRole | | |
| **Description** | This class acts like enum class, as this keeps static variables of each types of users that exist in the system | | |
| **Base Class** | Object | | |
| **Constructor** | Default | | |
| **Prototype** |  | | |
| **Source File** | UserRole.cs | | |
| **Namespace** | BSMS.Model | | |
| **Attribute** | Name | Type | Description |
| ADMIN | int | This contains role id of Admin users |
| CUSTOMER | int | This contains role id of Customer users |
|  | GUEST | int | This contains role id of Guest users |
|  | STAFF | int | This contains role id of Staff users |

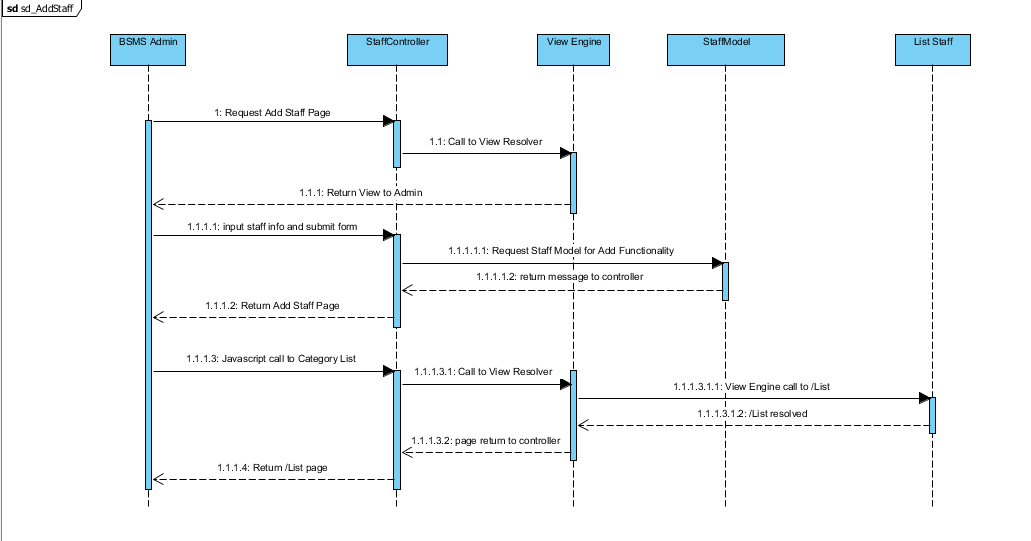
### Add Staff

Adding staff in to the system is one of the many functionality that is only allowed of the admin user, which after the staff will be given some level of access to the system and allowed to perform some operations.

#### Screen Design



#### Sequence Diagram

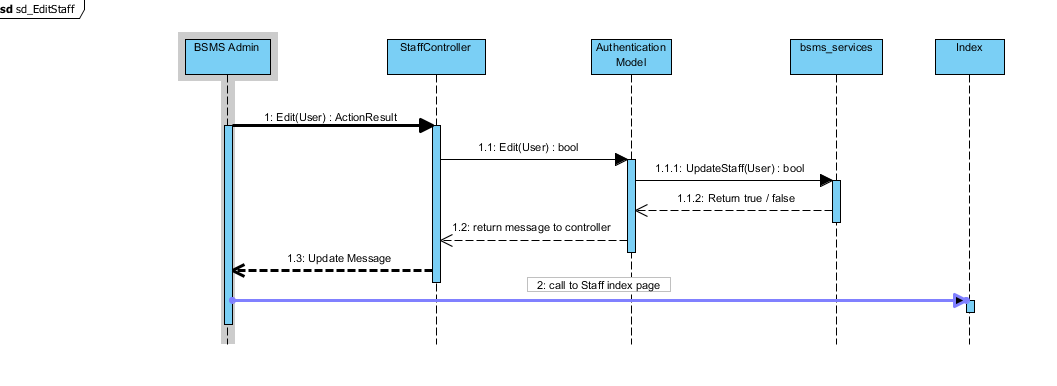


### Edit Staff

This functionality allow previously created staff to be edited if there exist any wrong information, editing information of staff including the name and other information e.g. changing staff first name from “Joe” to “John”

#### Screen Design

#### Sequence Diagram

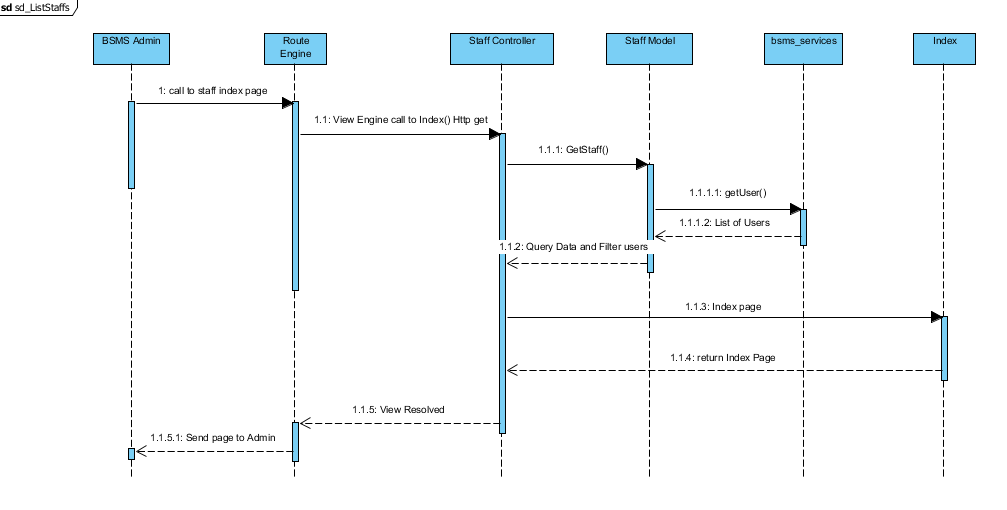


### List Staff

The list staff feature allow the admin use to view sequence staffs that have been added into the system, with each element of the sequence having edit, delete and info icon. On click of each icon triggers equivalent event for each element in that list.

#### Screen Design

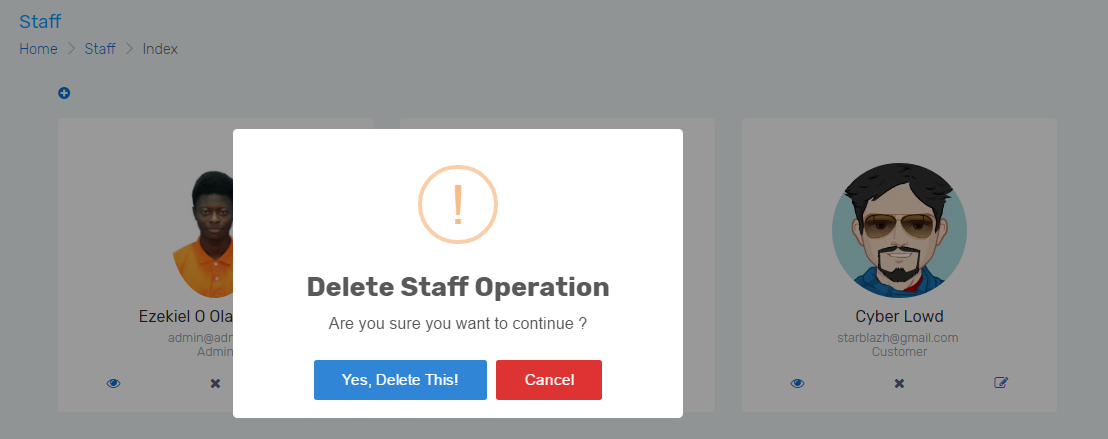
#### Sequence Diagram



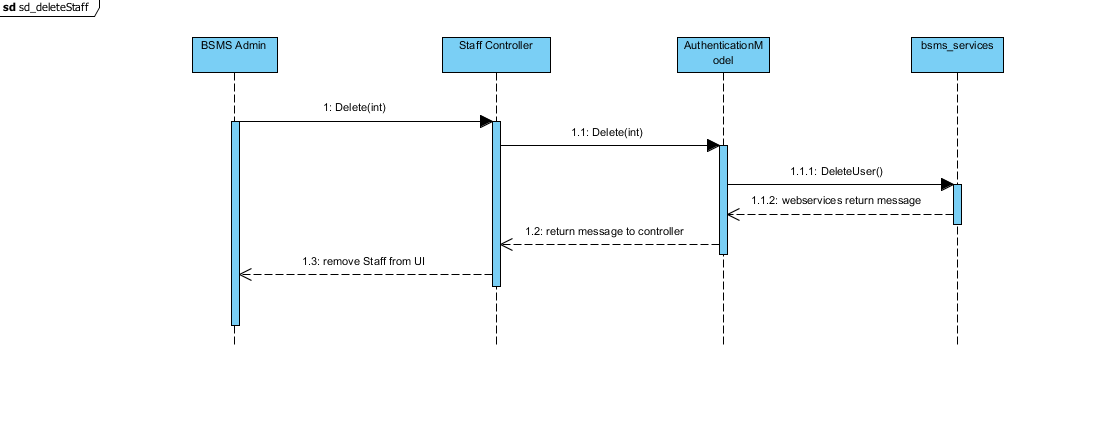
### Delete Staff

This feature is only allowed of the admin users to delete staffs from the system, this feature is useful in the case where admin user added wrong/duplicate/unnecessary category, and this feature will allow admin to remit such mistake.

#### Screen Design

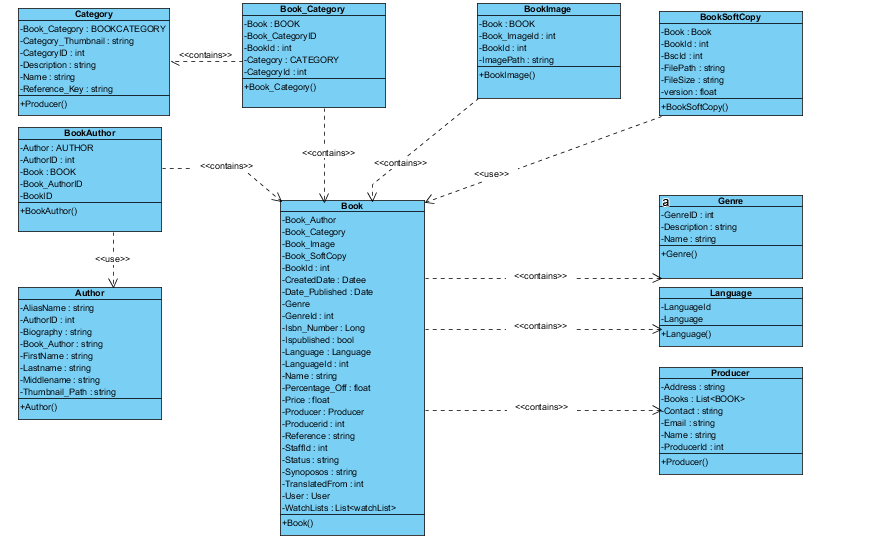


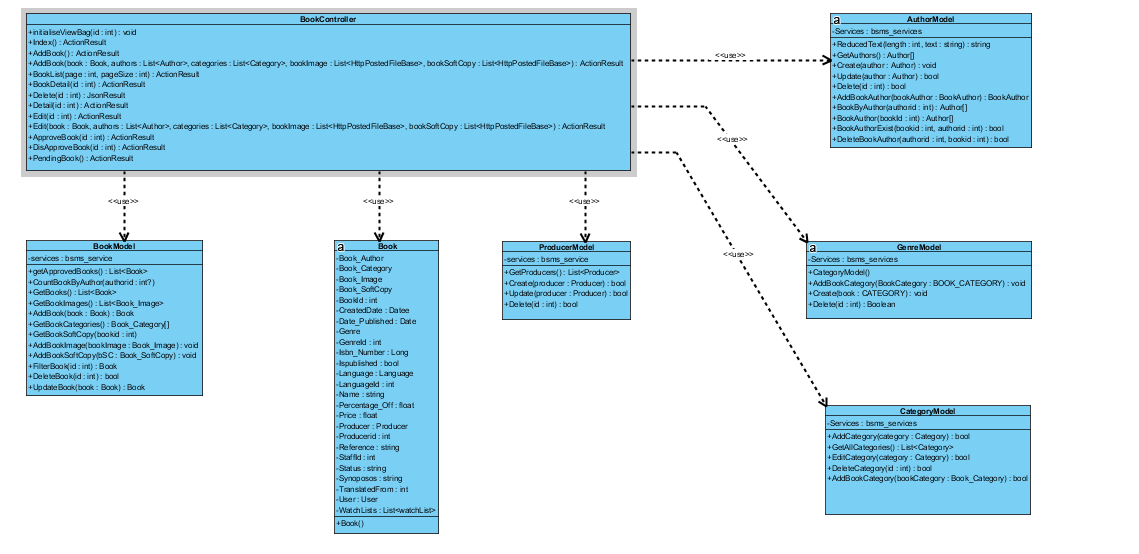
#### Sequence Diagram



## Book Management

### Class Diagram





### Class Description

|  |  |  |
| --- | --- | --- |
| Class | Dependency | Description |
| BookAuthor | N/A | Book author is a linq database mapped entity class, that store the information of Book and that of its Author |
| BookCategory | N/A | Book Category is a linq database mapped entity class, that store the information of Book and that of its Category |
| BookImage | N/A | Book image is a linq database mapped entity class, that store the information of Book and that of its image |
| BookSoftCopy | N/A | Book soft copy is a linq database mapped entity class, that store the information of Book and that of its soft copy, normally .pdf formatted file. |
| Author |  | Author is a linq database mapped entity class, which store the information of Author. |
| Category |  | Category is a linq database mapped entity class, which store the information of category. |
| Producer |  | Producer is a linq database mapped entity class, which store the information of producer. |
| Language |  | Language is a linq database mapped entity class, which store the information of language. |
| Genre |  | Genre is a linq database mapped entity class, which store the information of genre. |
| AuthorModel |  | This is a class that handles all operations that involves Author including CRUD Operations, thus this class have direct connection to BSMS SOAP Webservices. |
| CategoryModel |  | This is a class that handles all operations that involves Category including CRUD Operations, thus this class have direct connection to BSMS SOAP Webservices. |
| BookModel |  | This is a class that handles all operations that involves book including CRUD Operations, thus this class have direct connection to BSMS SOAP Webservices. |
| ProducerModel |  | This is a class that handles all operations that involves Producer including CRUD Operations, thus this class have direct connection to BSMS SOAP Webservices. |
| GenreModel |  | This is a class that handles all operations that involves Genre including CRUD Operations, thus this class have direct connection to BSMS SOAP Webservices. |
| BookController |  | This is a class that handles all operations that involves book including CRUD Operations, thus this class have direct connection to BSMS SOAP Webservices. |
| Book |  |  |

#### BookAuthor

|  |  |  |  |
| --- | --- | --- | --- |
| Class | BookAuthor | | |
| **Description** | This is an entity class that holds all information with regards to Book Author, information including book, and author information. | | |
| **Base Class** | Object | | |
| **Constructor** | Default | | |
| **Prototype** |  | | |
| **Source File** | BookAuthor.cs | | |
| **Namespace** | BSMS.bsms.localhost | | |
| **Attributes** | Name | Type | Description |
| Author | Author | N/A |
| AuthorId | Int | N/A |
| Book | Book | This is a unique identifyer for each geries, and this is what serves as key BSMS uses in recorgnising a Genre |
| BookAuthorId | Int | This field contains description for genre |
| BookId | Int |  |

#### BookCategory

|  |  |  |  |
| --- | --- | --- | --- |
| Class | BookCategory | | |
| **Description** | This is an entity class that holds information of category and books, Mapped linq to sql entity class. | | |
| **Base Class** | Object | | |
| **Constructor** | Default | | |
| **Prototype** |  | | |
| **Source File** | BookCategory.cs | | |
| **Namespace** | Object | | |
| **Attribute** | Name | Type | Description |
| book | Book | This method is use to add new book category into BSMS |
| bookId | int | This helper method is used to insert new category into the system |
| categoryId | int | this is a static internal type of method, which allows existing category to be deleted as the name Delete implies |
| category | Category | 44 |
| book\_categoryId | int |  |

#### BookImage

|  |  |  |  |
| --- | --- | --- | --- |
| Class | BookImage | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | |
| **Base Class** | Object | | |
| **Constructor** | N/A | | |
| **Prototype** |  | | |
| **Source File** | BookImage.cs | | |
| **Namespace** | BSMS.bsms.localhost | | |
| **Methods** | Name | Type | Description |
| book | Book | This is serves as http get handler |
| book\_imageId | int | This is a web method that handles user input for create |
|  | bookid | Int | This is a controller web method that handles all http get for Edit functionality |
| imagePath | string | This is a controller http post method that handles edit functionalities for genres |

#### BookSoftCopy

|  |  |  |  |
| --- | --- | --- | --- |
| Class | BookSoftCopy | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | |
| **Base Class** | Object | | |
| **Constructor** | N/A | | |
| **Prototype** |  | | |
| **Source File** | BookSoftCopy.cs | | |
| **Namespace** | BSMS.bsms.localhost | | |
| **Methods** | Name | Type | Description |
| book | Book | This is serves as http get handler |
| bookId | int | This is a web method that handles user input for create |
|  | bscId | Int | This is a controller web method that handles all http get for Edit functionality |
| filePath | string | This is a controller http post method that handles edit functionalities for genres |
|  | fileSize | int |  |
|  | version | float |  |

#### Author

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Author | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | |
| **Base Class** | Object | | |
| **Constructor** | N/A | | |
| **Prototype** |  | | |
| **Source File** | Author.cs | | |
| **Namespace** | BSMS.bsms.localhost | | |
| **`** | Name | Type | Description |
| aliasName | Book | Author’s alias name |
| authorId | Int | Author’s unique identifier |
|  | biography | Int | Biography of author, this might contains the background history of this author. |
| Book\_author | String | This is a controller http post method that handles edit functionalities for genres |
|  | firstName | String | Author first name |
|  | lastName | String | Author last name |
|  | middleName | String | Author’s Middle name |

#### Category

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Category | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | |
| **Base Class** | Object | | |
| **Constructor** | Default | | |
| **Prototype** |  | | |
| **Source File** | Category.cs | | |
| **Namespace** | BSMS.bsms.localhost | | |
| **Attributes** | Name | Type | Description |
| book\_category | BookCategory | This is serves as http get handler |
| category\_thumbnail | String | This is a web method that handles user input for create |
|  | categoryid | Int | This is a controller web method that handles all http get for Edit functionality |
| description | string | This is a controller http post method that handles edit functionalities for genres |
|  | name | String |  |
|  | reference\_key | String |  |

#### Producer

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Producer | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | |
| **Base Class** | Object | | |
| **Constructor** | Default | | |
| **Prototype** |  | | |
| **Source File** | Producer.cs | | |
| **Namespace** | BSMS.bsms.localhost | | |
| **Attributes** | Name | Type | Description |
| address | BookCategory | This is serves as http get handler |
| books | List<Book> | This is a web method that handles user input for create |
|  | email | String | This is a controller web method that handles all http get for Edit functionality |
| name | String | This is a controller http post method that handles edit functionalities for genres |
|  | producerid | Int |  |

#### Language

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Category | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | |
| **Base Class** | Object | | |
| **Constructor** | Default | | |
| **Prototype** |  | | |
| **Source File** | Category.cs | | |
| **Namespace** | BSMS.bsms.localhost | | |
| **Attributes** | Name | Type | Description |
| books | List<Book> | This is serves as http get handler |
| Language | String | This is a web method that handles user input for create |
|  | languageid | Int | This is a controller web method that handles all http get for Edit functionality |

#### Genre

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Genre | | |
| **Description** | this class holds all genre information including name, and description | | |
| **Base Class** | Object | | |
| **Constructor** | Default | | |
| **Prototype** |  | | |
| **Source File** | Genre.cs | | |
| **Namespace** | BSMS.bsms.localhost | | |
| **Attributes** | Name | Type | Description |
| Name | String | N/A |
| Book\_Genre | BOOK\_GENRE | N/A |
| GenreID | Int | This is a unique identifyer for each geries, and this is what serves as key BSMS uses in recorgnising a Genre |
| Description | String | This field contains description for genre |
|  |  |  |

#### AuthorModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | AuthorModel | | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | | |
| **Base Class** | Object | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | AuthorModel.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Attributes** | Name | Type | Description | |
| localhost | Bsms\_services | A private instance of bsms soap web services | |
| **Methods** | Name | Input | Output | Description |
| ReducedText | Int , String | String | This method reduces string values to the number of length passed as parameter to it |
| GetAuthors | N/A | List<Author> | This helper method is used to get all record of author from the soap web services |
|  | Create | Author | Boolean | This helper method is used when creating a new author, which incase if parameter passed is valid then it performs the add functionality on bsms dastabase and returns true and false otherwise |
|  | Update | Author | Boolean | This helper method is used when updating an existing author, which incase if parameter passed is valid then it performs update functionality on bsms dastabase and returns true and false otherwise |
|  | Delete | Int | Boolean | this is a static internal type of method, which allows existing category to be deleted as the name Delete implies |
|  | AddBookAuthor | Book\_Author | Boolean | This helper method is used when creating a new book author, which incase if parameter passed is valid then it performs the add functionality on bsms dastabase and returns true and false otherwise |
|  | BookByAuthor | Int | List<Book> | This Author model static method takes as input, id of author and returns back to invoking class list of book written by the author, if author id is incorrect they system returns and empty list. |
|  | BookAuthor | Int | List<Author> | The static method takes as input id of book and returns back to the invoking parent a sequence of author which is the author that wrote the book |
|  | BookAuthorExist | Int, Int | Boolean | The main intention of creating this method in author is to verify if book author have previously exist in the system. This method takes in book id and author id and check if they exist or not, as the name implies. |
|  | DeleteBookAuthor | Int, Int | Boolean | This method carries out the delete operation for Book Author as a result of invoking this method, book author information that match the invoked parameters will be permanently delete from bsms. |

#### CategoryModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | CategoryModel | | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | | |
| **Base Class** | Object | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | Category.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Attributes** | Name | Type | Description | |
| Services | Bsms\_services | A private instance of bsms soap web services | |
| **Methods** | Name | Input | Output | Description |
| AddBookCategory | BOOK\_CATEGORY | N/A | This method is use to add new book category into BSMS |
| Create | Category | void | This helper method is used to insert new category into the system |
|  | Delete | Int | Boolean | this is a static internal type of method, whcih allows existing category to be deleted as the name Delete implies |
|  | EditCategory | Category | Boolean | This model method edit information of category and return true if model state is valid, and false otherwise. |
|  | Fliter | Int | Category | This model method takes as input category id and return equivalent category as output if found, return null otherwise |

#### BookModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | bookModel | | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | | |
| **Base Class** | Object | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | BookModel.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Attributes** | Name | Type | Description | |
| service | Bsms\_services | A private instance of bsms soap web services | |
| **Methods** | Name | Input | Output | Description |
| GetApprovedBooks | N/A | List<Book> | This method query data from web services and return all books that have been approved |
| CountBookByAuthor | Int | Int | This helper method is used to get the amount of book that have been written by a particular author. |
|  | GetBooks | N/A | List<Book> | This model method main function is to get all books from the web service without filter |
|  | AddBook | Book | Boolean | Model method that performs add functionality into the system, by call the add book web method as describe in the web method description below |
|  | GetBookCategories | N/A | List<BookCategory> | This is a static internal type of method, which get all book categories from the web services without any filter added. |
|  | GetBookSoftCopy | N/A | List<Book\_SoftCopy> | This is a static internal type of method, which get all book book soft copy from the web services without any filter added. |
|  | AddBookImage | Book\_Image | Boolean | This Book model static method takes as input, the book image object and returns back to invoking parent Boolean result, if book image is successfully added in the system returns true and false otherwise. |
|  | AddBookSoftCopy | Book\_SoftCopy | Boolean | This Book model static method takes as input, the book soft copy object and returns back to invoking parent Boolean result, if object is successfully added in the system returns true and false otherwise. |
|  | FilterBook | Int | Book | The aim of this model method is to search for a particular book in the system, and returns book that match the given id, null otherwise |
|  | DeleteBook | Int | Boolean | This method carries out the delete operation for Book as a result of invoking this method, book information that match the invoked parameters will be permanently delete from bsms. |
|  | UpdateBook | Book | Boolean | This method carries out the update operation on book. As a result of invoking this method, book information that match the invoked parameters will be permanently updated and changes committed to bsms database. |

#### ProducerModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | ProducerModel | | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | | |
| **Base Class** | Object | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | Producer.Model.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Attributes** | Name | Type | Description | |
| Services | Bsms\_services | A private instance of bsms soap web services | |
| **Methods** | Name | Input | Output | Description |
| GetProducer | N/A | List<Producer> | This model helper method communicates directly with the bsms web service to fetch data from the database |
| Create | Producer | void | This helper method is used to insert new producer into the system |
|  | Delete | Int | Boolean | this is a static internal type of method, which allows existing producer to be deleted as the name Delete implies. |
|  | Update | Producer | Boolean | This is a static internal method that performs edit operation for producers by calling its equivalent web method in the web service, this returns Boolean result as output. |
|  | Filter | Int | Producer | This method main intention is to find and return producer that match the given unique value, this method will return null is not match is found. |

#### Genre Model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | GenreModel | | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | | |
| **Base Class** | Object | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | GenreModel.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Attributes** | Name | Type | Description | |
| Services | Bsms\_services | A private instance of bsms soap web services | |
| **Methods** | Name | Input | Output | Description |
| GetGenre | Genre | N/A | This method is use to add new book genre into BSMS |
| Create | Genre | void | This helper method is used to insert new genre into the system |
|  | Delete | Int | Boolean | this is a static internal type of method, which allows existing category to be deleted as the name Delete implies |

#### BookController

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | BookController | | | |
| **Description** | This class controls everything they deals with User and Roles | | | |
| **Base Class** | System.Web.Mvc.Controller | | | |
| **Constructor** | N/A | | | |
| **Prototype** |  | | | |
| **Source File** | BookController.cs | | | |
| **Namespace** | BSMS.Controller | | | |
| **Methods** | Name | Input | Output | Description |
| initialiseViewBag | Int | N/A | This is a privater controller method that initialise all view bags values for the view to make use of. |
| Index | N/A | ActionResult | This is the http get method, which return the list of all book as action result to the view to make use of. |
| AddBook | N/A | ActionResult | Http get method to display the add book page. |
| AddBook | Book, List<Author>, List<int>, List<HttpPostFileBase>, List<HttpPostFileBase> | ActionResult | Http Post method of book controller that handles add request by calling the static method add of BookModel class. |
| BookList | N/A | ActionResult | This book list method show list of books for the guest and customer users which display all the list of approved books |
| BookDetail | Int | ActionResult | Http get request method handler that process and display information of book with the given id. |
| Delete | Int | JsonResult | http delete method that handles delete request from bsms |
|  | BookDetail | Int | ActionResult | Http get request method handler that process and display information of book with the given id for admin users. |
|  | Edit | Int | ActionResult | Http get method to display the edit book page o any book that match the given id, if such book is does not exist this will return Http Not found with status code 404. |
|  | Edit | Book, List<Author>, List<int>, List<HttpPostFileBase>, List<HttpPostFileBase> | ActionResult | Http Post method of book controller that handles edit request by calling the static method update of BookModel class. |
|  | ApprovedBook | Int | JsonResult | Http post method that handles the approve functionality of bsms, at the call of this controller method, book will be immediately approve |
|  | DisApprovedBook | Int | JsonResult | Http post method that handles the disapprove functionality of bsms, at the call of this controller method, book will be immediately disapprove |
|  | PendingBooks | N/A | ActionResult | This http post method of book controller main intention is to display list of pending book |

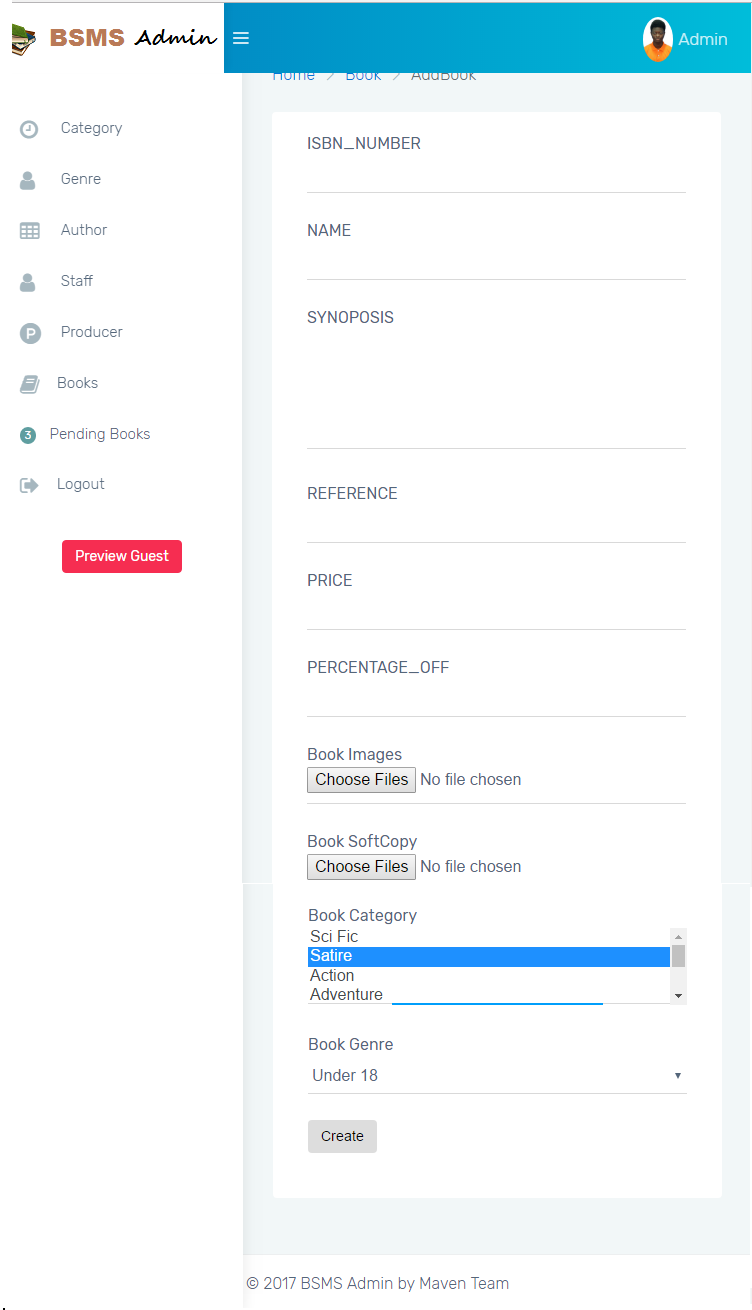
#### Book

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Book | | |
| **Description** | this class holds all book information including name, and synopsis | | |
| **Base Class** | Object | | |
| **Constructor** | Default | | |
| **Prototype** |  | | |
| **Source File** | Book.cs | | |
| **Namespace** | BSMS.bsms.localhost | | |
| **Attributes** | Name | Type | Description |
| Book\_Author | Book\_Author | Book author information |
| Book\_Category | Book\_Category | Book category information |
| Book\_Image | Book\_Image | Book image information |
| BookId | Int | This field contain the unique identifier for each book |
| CreatedDate | Date | Book created date time stamp |
|  | Date\_Published | Date | Book date publish time stamp |
|  | Genre | Genre | Book genre information |
|  | Genreid | Int | Book genre unique id |
|  | Isbn\_number | BigInt | Book isbn number |
|  | IsPublished | Boolean | Boolean value to confirm if book is published or not. |
|  | Language | Language | Book language object that stores the language the book is written in. |
|  | LanguageId | Int | Book language id |
|  | Name | String | Name of the book |
|  | Percentage\_Off | Float | Percentage off from the book |
|  | Price | Float | Unit / Original price of the book |
|  | Producer | Producer | Book producer information |
|  | ProducerId | Int | Book producer id |
|  | Reference | String | Web link that contains reference to the given book |
|  | StaffId | Int | Book staff id |
|  | Status | String | Book status “Approve” or “Not Approve” |
|  | Synopsis | String | Book synopsis information |
|  | TranslatedFrom | Int | Book translated from |
|  | User | User | This field store the id of the staff whom add this book into bsms |
|  | WatchLists | List<WatchList> | List of wishlist |

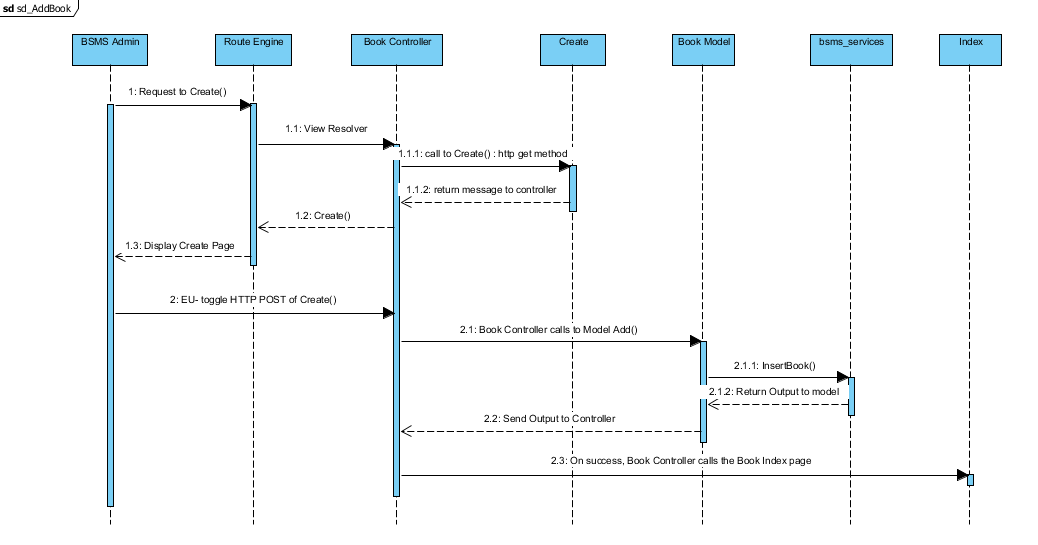
### Add Book

Adding book is one of the most important feature of our system and in order for bsms to work correctly there is need for adding a book. Adding a book requires so many dependencies

#### Screen Design



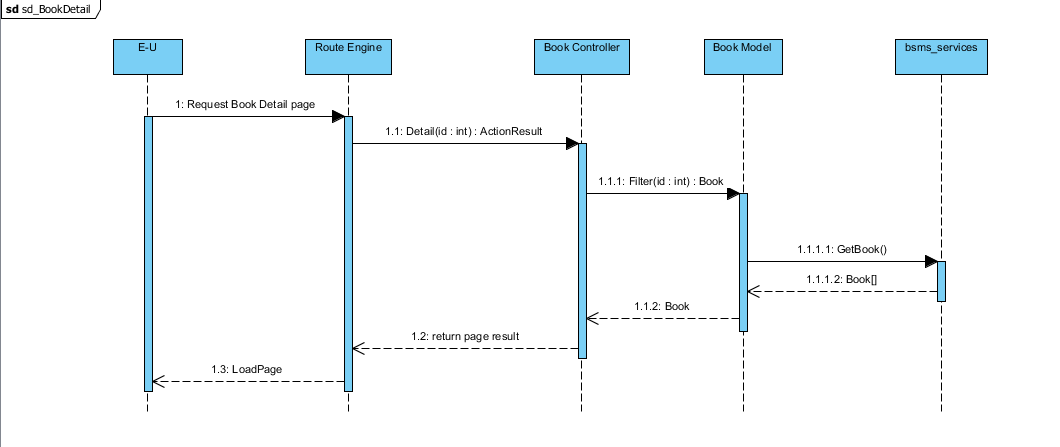
#### Sequence Diagram



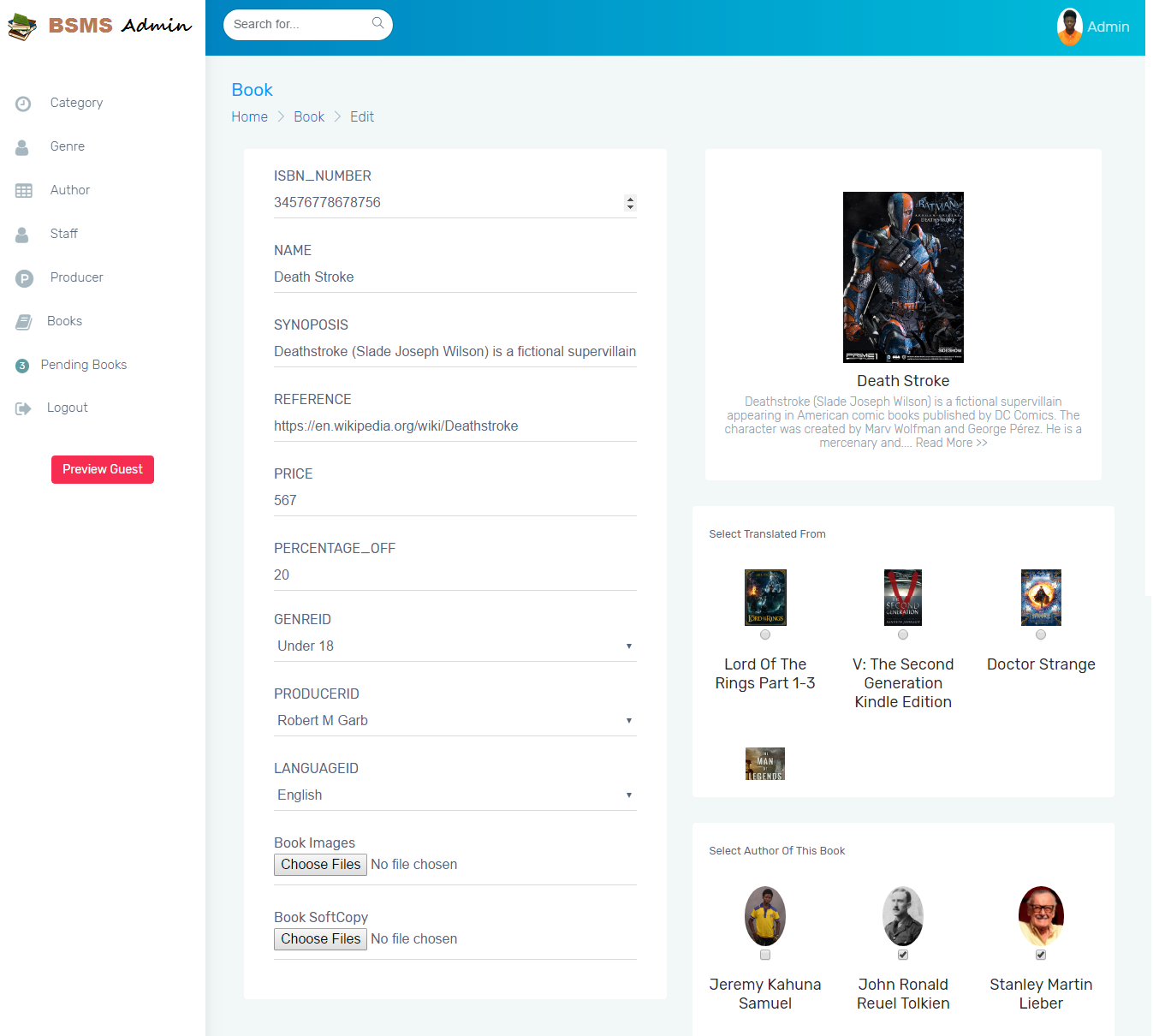
### Book Detail

This functionality allow previously created books to view and all details about the book revealed to the E-U including Authors information, Producer, Genre and many other info.

#### Sequence Diagram



### Edit Book

The edit book functionality allow the staff users to view and update book information, from this same page you can choose to delete book image, book author and other book dependency information

#### Screen Design

#### Sequence Diagram

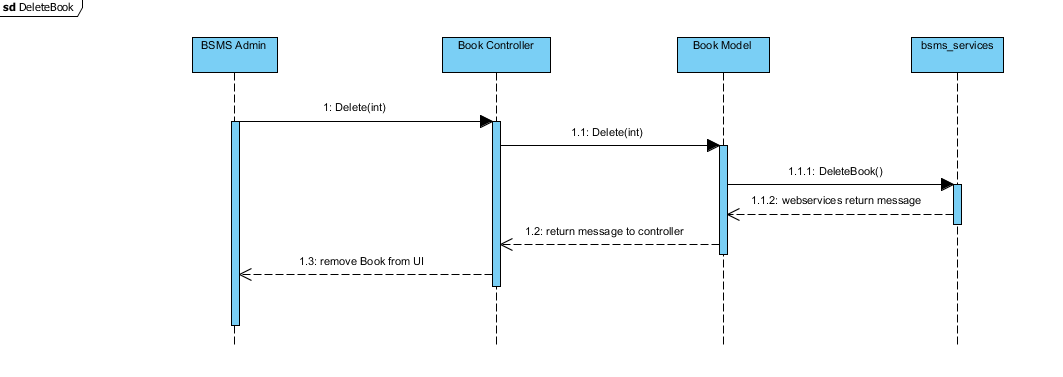


### Delete Book

This feature allow staff users to delete book from the system, this feature is useful in the case where a staff user added wrong/duplicate/ unnecessary book, this feature will allow admin to delete such book.

#### Screen Design

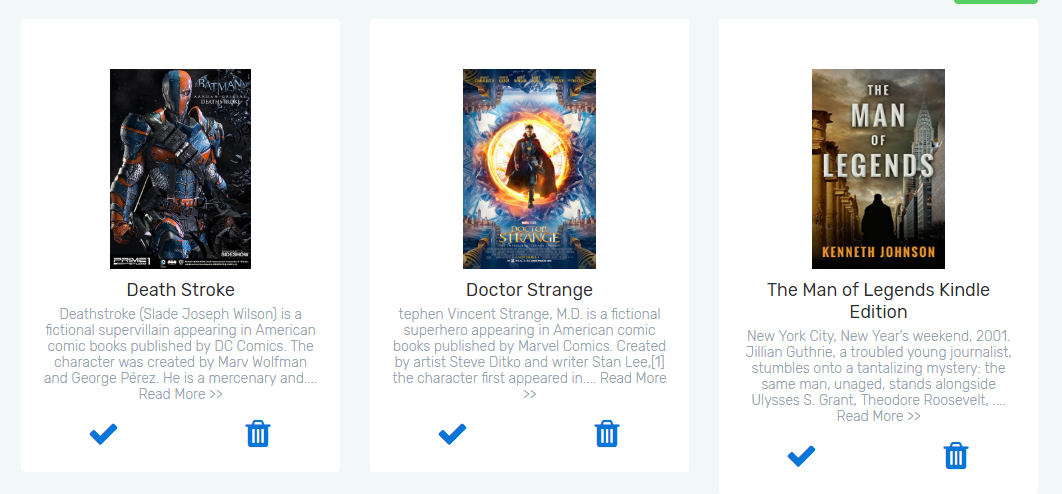
#### Sequence Diagram



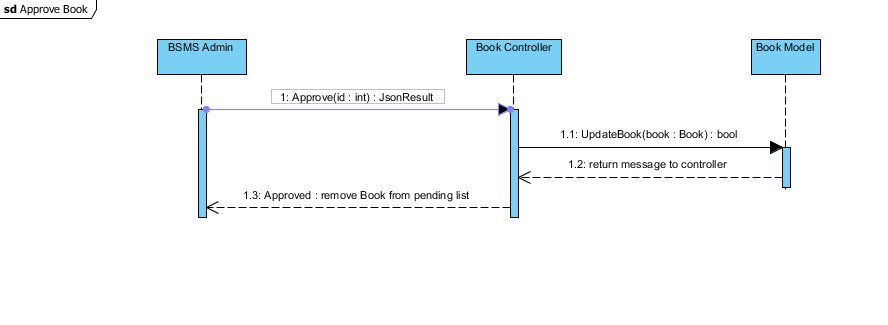
### Approve/Disapprove Book

In order to allow customers and guest to see newly added book there is neeed to approve such book, this functionality is only allowed of the admin, which he/she need to review new books and decide whether the book is allowed or not.

#### Screen Design

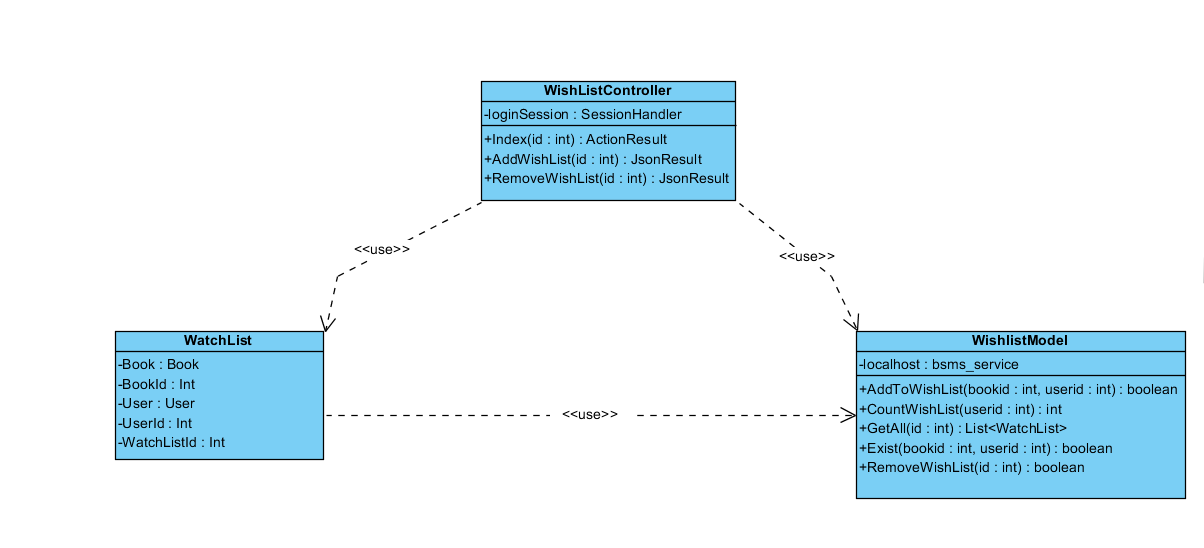


#### Sequence Diagram



## Wishlist Management

### Class Diagram



### Class Description

|  |  |  |
| --- | --- | --- |
| Class | Dependency | Description |
| WatchList | Role | The watchlist class holds all user data and information including all the book and user, linq to sql mapped entity class. |
| WishlistModel | N/A | This model class is intended to perform CRUD operations and some other simple operation for the wishlist including add, remove, count and get functionality. |
| WishListController | N/A | This is a controller class that handles all operations that involving http Operations, once any corresponding http method is call the route engine fires up this controller by requesting it for action. |

#### WatchList

|  |  |  |  |
| --- | --- | --- | --- |
| Class | WatchList | | |
| **Description** | The watchlist class holds all user data and information including all the book and user, linq to sql mapped entity class. | | |
| **Base Class** | Object | | |
| **Constructor** | Default | | |
| **Prototype** |  | | |
| **Source File** | User.cs | | |
| **Namespace** | BSMS.Model | | |
| **Attributes** | Name | Type | Description |
| Book | Book | N/A |
| Bookid | int | Users username is contained in this attribute |
| User | User | This contains 64 digit MD5 encrypted password |
| Userid | int |  |
| wishlistid | int | This hold the role of user in number format ex. (1:”Staff” , 2:”Admin”) |
| Thumbnail | String | This hold the thumbnail path in the system |

#### WishlistModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | WishlistModel | | | |
| **Description** | This model class is intended to perform CRUD operations and some other simple operation for the wishlist including add, remove, count and get functionality. | | | |
| **Base Class** | N/A | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | Role.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Attributes** | Name | Type | Description | |
| localhost | bsms\_service | Localhost is an instance of the soap web service that manages all the action wishlist model takes. | |
| **Methods** | Name | Input | Output | Description |
|  | AddToWishList | WatchList | Boolean | This model method takes as input the watchlist object and output Boolean result, true if wishlist addition is successful and false otherwise |
| CountWishList | Int | Int | This model method query data to count the amount of watchlist a particular user have. |
| GetAll | Int | List<WatchList> | Model method that return all the watchlist a particular user have in the database |
| Exist | Int, Int | Boolean | Model method that check if the specify user and book id already exist in the database, this method helps in eliminating duplicate values. |
| RemoveWishList | Int | Boolean | This model method takes as input the watchlist id and output Boolean result, true if wishlist have been successfully removed and false otherwise |

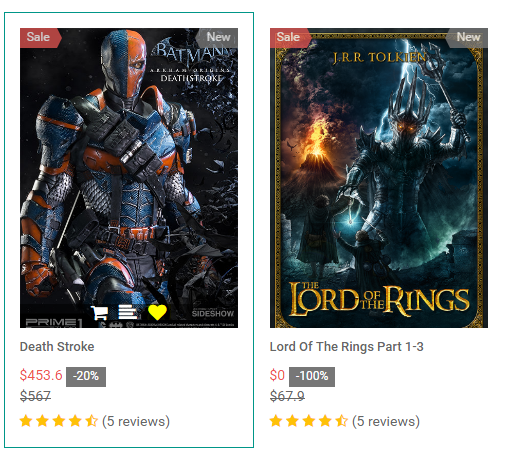
#### WishListController

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | WishListController | | | |
| **Description** | This controller class handles all http method request, and serves as page backend controller. | | | |
| **Base Class** | System.Web.Mvc.Controller | | | |
| **Constructor** | N/A | | | |
| **Prototype** |  | | | |
| **Source File** | WishListController.cs | | | |
| **Namespace** | BSMS.Controller | | | |
| **Methods** | Name | Input | Output | Description |
| Index | Int | ActionResult | This method make a request to the Http Get in order to request for a wishlist a currently logged user |
| AddWishList | int | JsonResult | This method call makes request to Http Post and calls the Wishlist Model class to add a new wishlist. |
| RemoveWishList | N/A | ActionResult | This method call makes request to Http Post and calls the Wishlist Model class to remove an existing wishlist. |

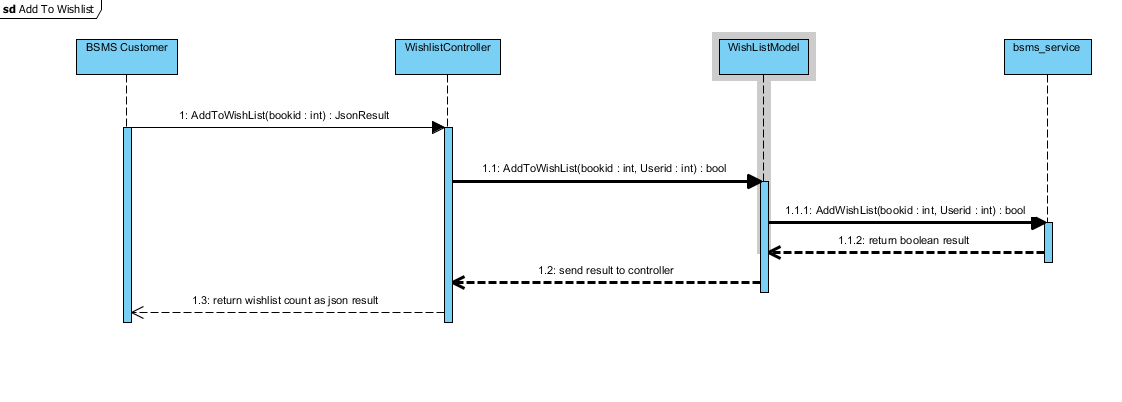
### Add To WishList

Adding wishlist operation in to the system is one of the many functionality that is only allowed of the customer user, which after they can view in the wishlist page and they can view and delete the book from their wishlist page if they so desired.

#### Screen Design



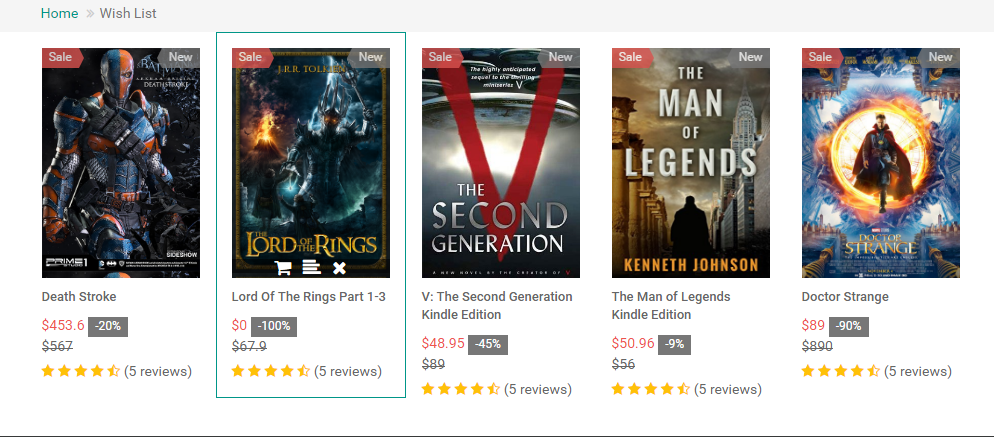
#### Sequence Diagram



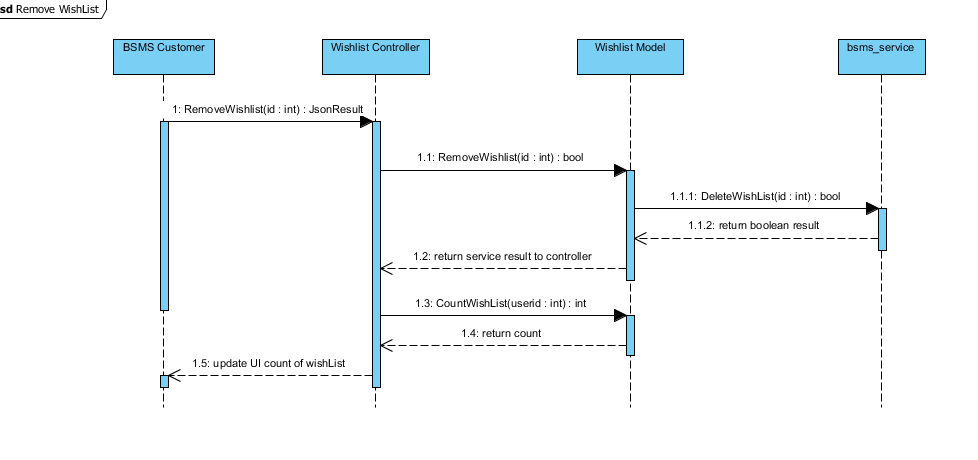
### Remove WishList

This functionality allow previously created wishlist to be delete if customer desires to remove this from his wishlist page.

#### Screen Design

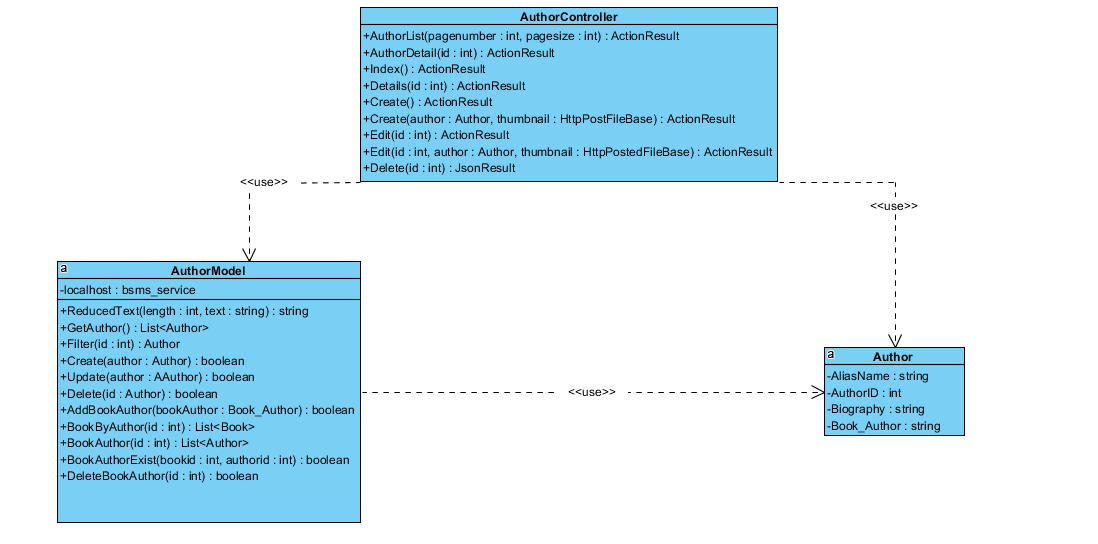


#### Sequence Diagram



## Author Management

### Class Diagram



### Class Description

|  |  |  |
| --- | --- | --- |
| Class | Dependency | Description |
| Author | Role | The user class holds all user data and information including all the usernames and passwords in the system, this is known as the BSMS user recognition methodology. |
| AuthorModel | N/A | This class is intended to hold unique set of roles for users, since the user uses this class as dependency, user has a role, which is how the system recognise the type of user coming into the system |
| AuthorController | N/A | This is a controller class that handles all operations that involving http Operations, once any corresponding http method is call the route engine fires up this controller by requesting it for action. |

#### Author

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Author | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | |
| **Base Class** | Object | | |
| **Constructor** | N/A | | |
| **Prototype** |  | | |
| **Source File** | Author.cs | | |
| **Namespace** | BSMS.bsms.localhost | | |
| **`** | Name | Type | Description |
| aliasName | Book | Author’s alias name |
| authorId | Int | Author’s unique identifier |
|  | biography | Int | Biography of author, this might contains the background history of this author. |
| Book\_author | String | This is a controller http post method that handles edit functionalities for genres |
|  | firstName | String | Author first name |
|  | lastName | String | Author last name |
|  | middleName | String | Author’s Middle name |
|  | Thumbnail\_path | String | Author image path in the web server |

#### AuthorModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | AuthorModel | | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | | |
| **Base Class** | Object | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | AuthorModel.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Attributes** | Name | Type | Description | |
| localhost | Bsms\_services | A private instance of bsms soap web services | |
| **Methods** | Name | Input | Output | Description |
| ReducedText | Int , String | String | This method reduces string values to the number of length passed as parameter to it |
| GetAuthors | N/A | List<Author> | This helper method is used to get all record of author from the soap web services |
|  | Create | Author | Boolean | This helper method is used when creating a new author, which incase if parameter passed is valid then it performs the add functionality on bsms dastabase and returns true and false otherwise |
|  | Update | Author | Boolean | This helper method is used when updating an existing author, which incase if parameter passed is valid then it performs update functionality on bsms dastabase and returns true and false otherwise |
|  | Delete | Int | Boolean | this is a static internal type of method, which allows existing category to be deleted as the name Delete implies |
|  | AddBookAuthor | Book\_Author | Boolean | This helper method is used when creating a new book author, which incase if parameter passed is valid then it performs the add functionality on bsms dastabase and returns true and false otherwise |
|  | BookByAuthor | Int | List<Book> | This Author model static method takes as input, id of author and returns back to invoking class list of book written by the author, if author id is incorrect they system returns and empty list. |
|  | BookAuthor | Int | List<Author> | The static method takes as input id of book and returns back to the invoking parent a sequence of author which is the author that wrote the book |
|  | BookAuthorExist | Int, Int | Boolean | The main intention of creating this method in author is to verify if book author have previously exist in the system. This method takes in book id and author id and check if they exist or not, as the name implies. |
|  | DeleteBookAuthor | Int, Int | Boolean | This method carries out the delete operation for Book Author as a result of invoking this method, book author information that match the invoked parameters will be permanently delete from bsms. |

#### Author Controller

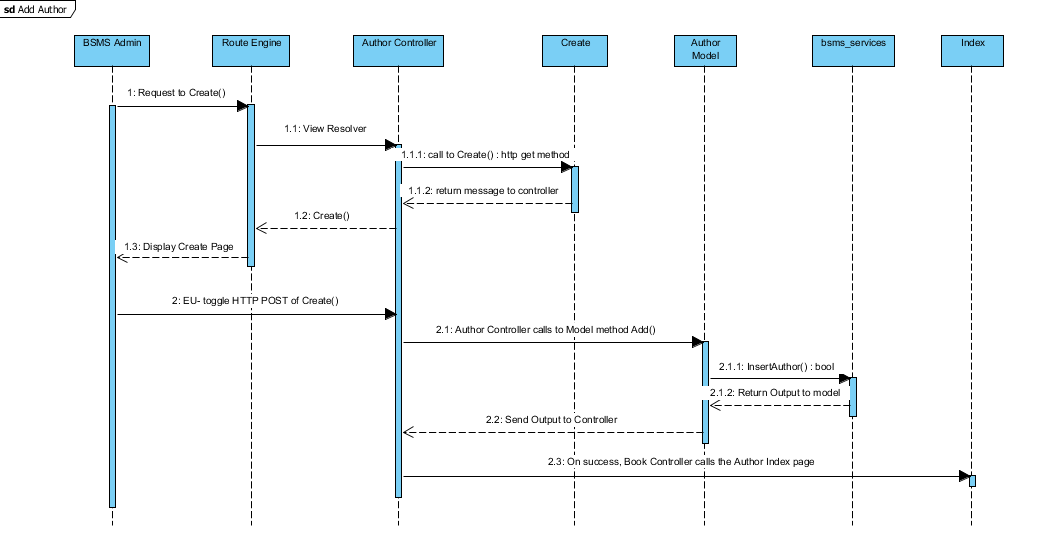
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | AuthorController | | | |
| **Description** | This class controls everything they deals with Author, http hander class for author. | | | |
| **Base Class** | System.Web.Mvc.Controller | | | |
| **Constructor** | N/A | | | |
| **Prototype** |  | | | |
| **Source File** | AuthorController.cs | | | |
| **Namespace** | BSMS.Controller | | | |
| **Methods** | Name | Input | Output | Description |
| AuthorList | Int , Int | ActionResult | This method make a request to the Http Get in order to request for a page |
| AuthorDetail | int | ActionResult | This method call makes request to Http get and calls then Author Model class to fetch author’s detail |
| Index | N/A | ActionResult | On call of this http get method, will in turn return the list author page in the admin and staff dashboard |
| Details | Int | ActionResult | Http Method call to view detail of author |
| Create | N/A | ActionResult | Http get method to display the create page of author. |
| Create | Author, HttppostFileBase | ActionResult | This is an http post request method which save a new author in to the database |
| Edit | Int | ActionResult | This is an http get method to display the edit author page that match the given id |
|  | Edit | Int,Author, HttppostFileBase | ActionResult | This is an http post request method which make changes to the database |
|  | Delete | Int | JsonResult | This is an http post request method which at call of it return Json result back to invoking client and also delete Author information from the database if there exist such user. |

### Add Author

Adding author in to the system is one of the many functionality that is only allowed of the staff user, which after adding a valid author information, the staff can then can book information.

#### Screen Design

#### Sequence Diagram

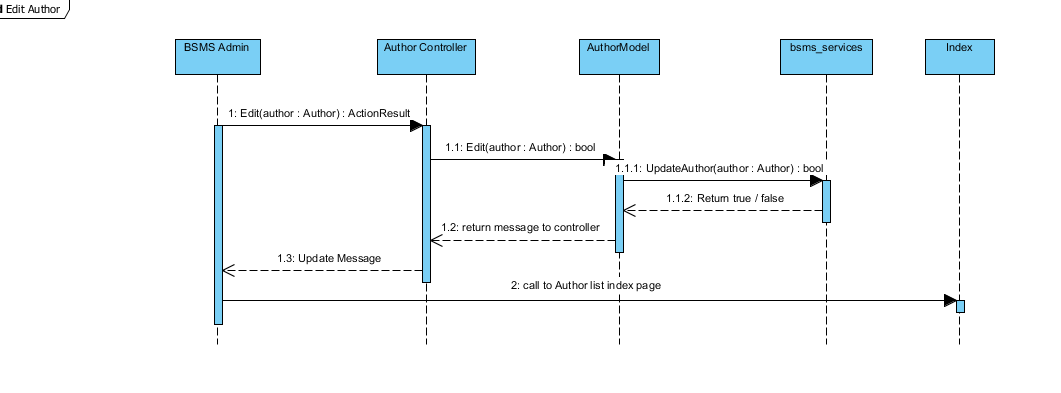


### Edit Author

This functionality allow previously created author to be edited if there exist any wrong information, editing information of author including the name and other information e.g. changing author first name from “Joe” to “John”

#### Screen Design

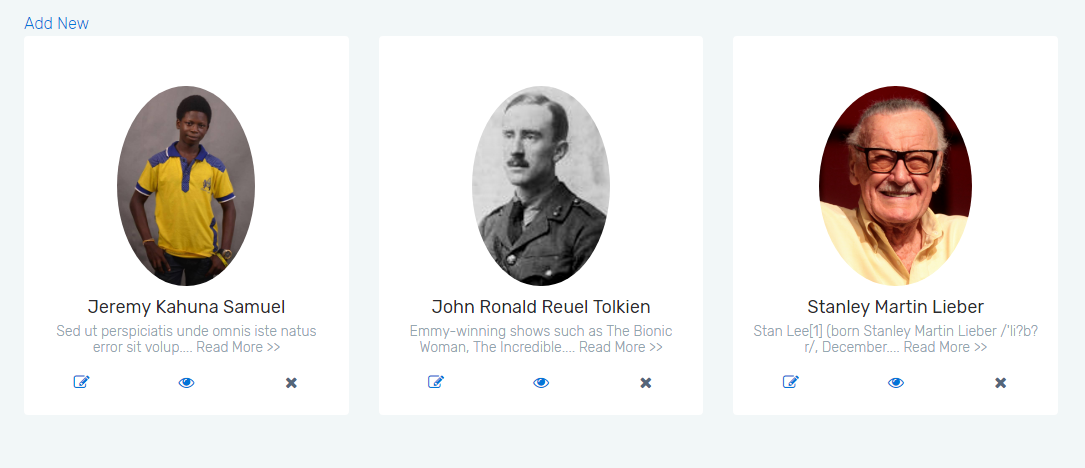
#### Sequence Diagram



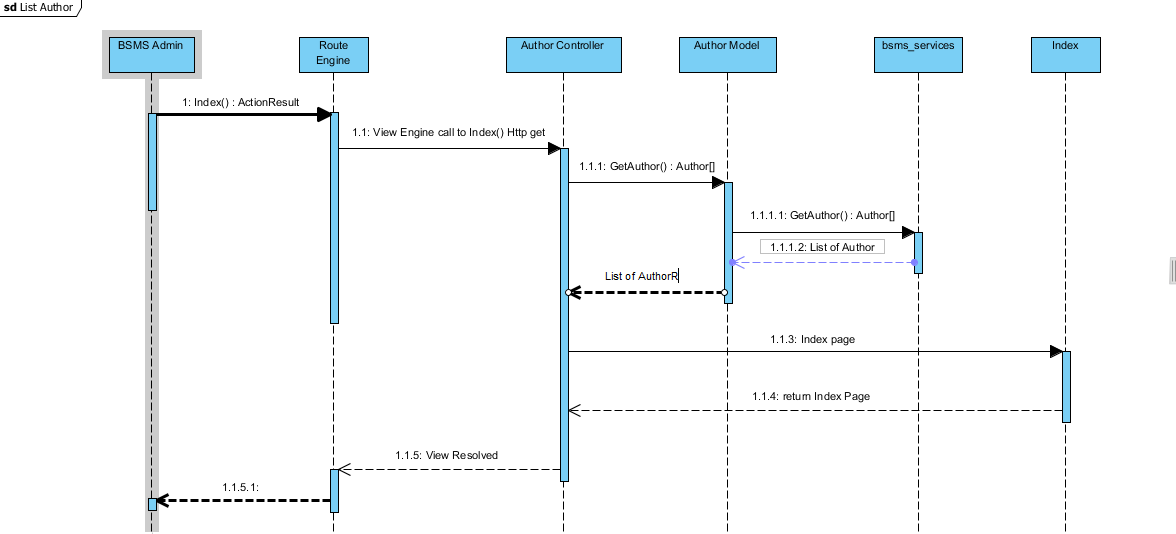
### List Author

The list author feature allow the admin and staff to view sequence authors that have been added into the system, with each element of the sequence having edit, delete and info icon. On click of each icon triggers equivalent event for each element in that list.

#### Screen Design



#### Sequence Diagram

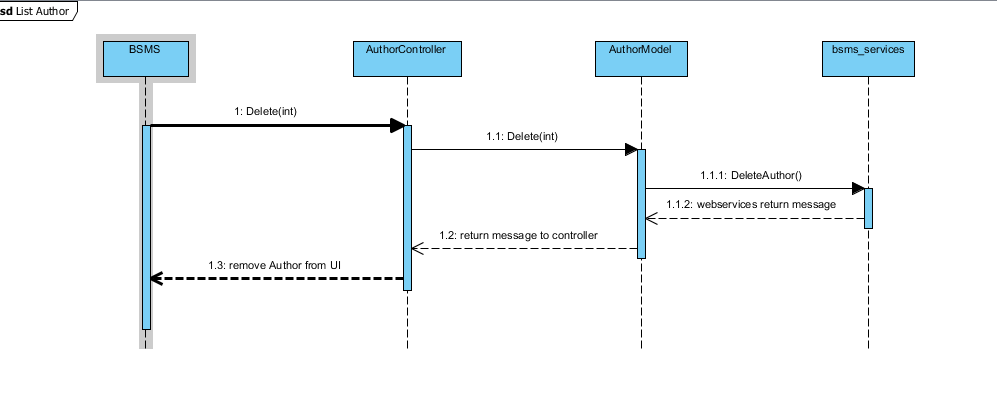


### Delete Author

This feature is only allowed of the staff users to delete author from the system, this feature is useful in the case where staff user added wrong/duplicate/unnecessary category, and this feature will allow admin to remit such mistake.

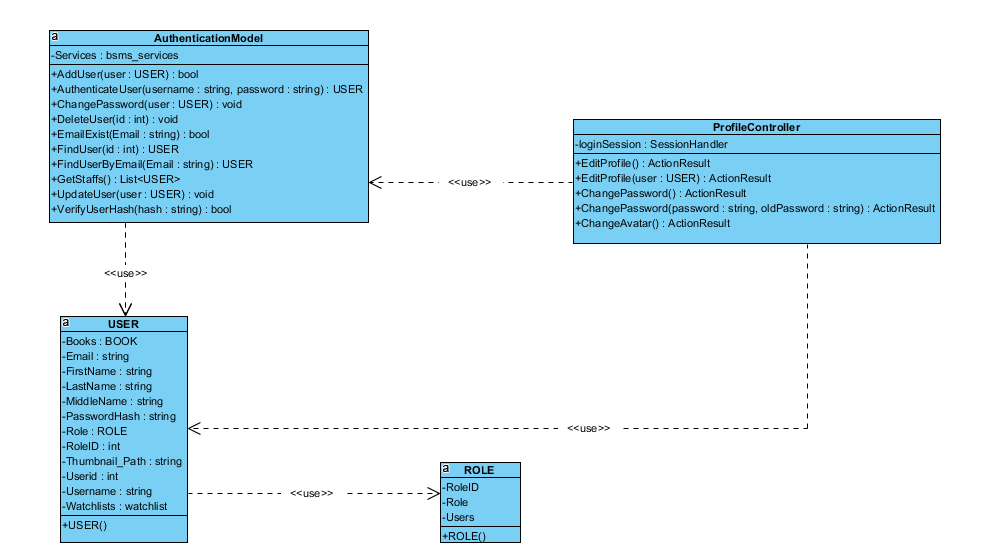
#### Screen Design

#### Sequence Diagram



## Profile Management

### Class Diagram



### Class Description

|  |  |  |
| --- | --- | --- |
| Class | Dependency | Description |
| User | Role | The user class holds all user data and information including all the usernames and passwords in the system, this is known as the BSMS user recognition methodology. |
| Role | N/A | This class is intended to hold unique set of roles for users, since the user uses this class as dependency, user has a role, which is how the system recognise the type of user coming into the system |
| ProfileController | N/A | This is a controller class that handles all operations that involving http Operations, once any corresponding http method is call the route engine fires up this controller by requesting it for action. |
| AuthenticationModel | USER | This is a class that handles all operations that involves Authentication including CRUD Operations, thus this class have direct connection to BSMS SOAP Webservices. |

#### User

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | User | | | |
| **Description** | User data, this class holds all the user information including the password and username | | | |
| **Base Class** | Authentication | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | User.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Attributes** | Name | Type | Description | |
| Name | String | N/A | |
| Username | String | Users username is contained in this attribute | |
| PasswordHash | String | This contains 64 digit MD5 encrypted password | |
| Email | String |  | |
| RoleID | int | This hold the role of user in number format ex. (1:”Staff” , 2:”Admin”) | |
|  | Thumbnail | String | This hold the thumbnail path in the system | |
| **Methods** | Name | Input | Output | Description |
| Authentication | Username  Password | User | This function check if user is valid or not, returns null if not valid and user information if valid |
| LoggedOn | N/A | Boolean | This function is intended to check if any user is logged on to the system |

#### Role

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Role | | |
| **Description** | This contain all the user roles that exists in the system | | |
| **Base Class** | N/A | | |
| **Constructor** | Default | | |
| **Prototype** |  | | |
| **Source File** | Role.cs | | |
| **Namespace** | BSMS.Model | | |
| **Attributes** | Name | Type | Description |
| Roleid | int | This is a unique integer value that holds a unique id for each role that exist in the system |
| Role | String | Role of this role e.g. (“Admin, Staff” .... ) |

#### ProfileController

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Class | ProfileController | | | | | |
| **Description** | This class controls everything they deals with User and Roles | | | | | |
| **Base Class** | System.Web.Mvc.Controller | | | | | |
| **Constructor** | N/A | | | | | |
| **Prototype** |  | | | | | |
| **Source File** | ProfileController.cs | | | | | |
| **Namespace** | BSMS.Controller | | | | | |
| **Attributes** | Name | | Type | | Description | |
| loginSession | | SessionHandler | | This class acts as session assistant, in other words this class handles all session related activities | |
| **Methods** | Name | Input | | Output | | Description |
| EditProfile | N/A | | ActionResult | | This method make a request to the Http Get in order to request for edit profile page |
| EditProfile | USER | | ActionResult | | This method call makes request to Http Post and calls the Authentication Model class to update user detail |
| ChangePassword | N/A | | ActionResult | | This is http get method to display the change password page. |
| ChangePassword | String, String | | ActionResult | | This is http post method to change user password. |
| ChangeAvatar | HttpPostFileBase | | ActionResult | | Http Post that changes profile avatar. |

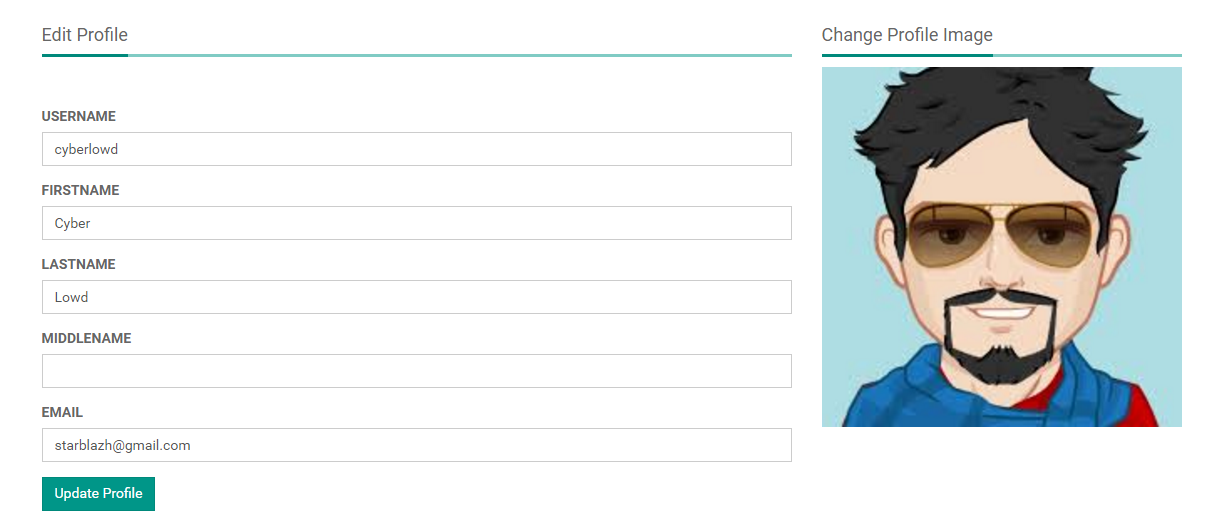
#### Authentication Model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | AuthenticationModel | | | |
| **Description** | This is a class that handles all operations that involves Authentication including CRUD Operations, thus this class have direct connection to BSMS SOAP Webservices. | | | |
| **Base Class** | N/A | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | Authentication.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Attributes** | Name | Type | Description | |
| Services | bsms\_services | This is a private instance of “BSMS.SERVICES” | |
| **Methods** | Name | Input | Output | Description |
| AddUser | USER | Boolean | This method is use to call the webservices method inorder to add a new user into the system |
| AuthenticateUser | String, String | USER | This method is proposed to check if login credential is correct or not, this return null if username or password is incorrect and User Object otherwise |
| ChangePassword | USER | N/A | This method is used to change user passwords on reset password request |
| DeleteUser | Int | N/A | This static method is used to delete user from BSMS |
| FindUser | Int | USER | This static method return User Object which matches the id of user inpt |
| FindUserByEmail | Int | USER | This method is used to find user by email, since the email address of each user is unique |
| GetStaffs | N/A | List<USER> | This method is used to get all avaliable staff in the system |
| UpdateStaff | USER | N/A | This is used to update staff information |
| VerifyUserHash | String | Boolean | This method is mainly used to verify hash code of users |

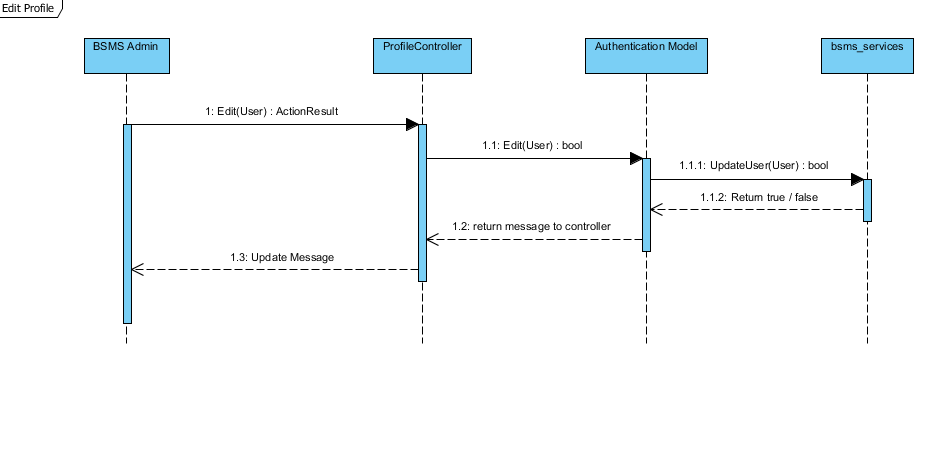
### Edit Profile

Adding staff in to the system is one of the many functionality that is only allowed of the admin user, which after the staff will be given some level of access to the system and allowed to perform some operations.

#### Screen Design



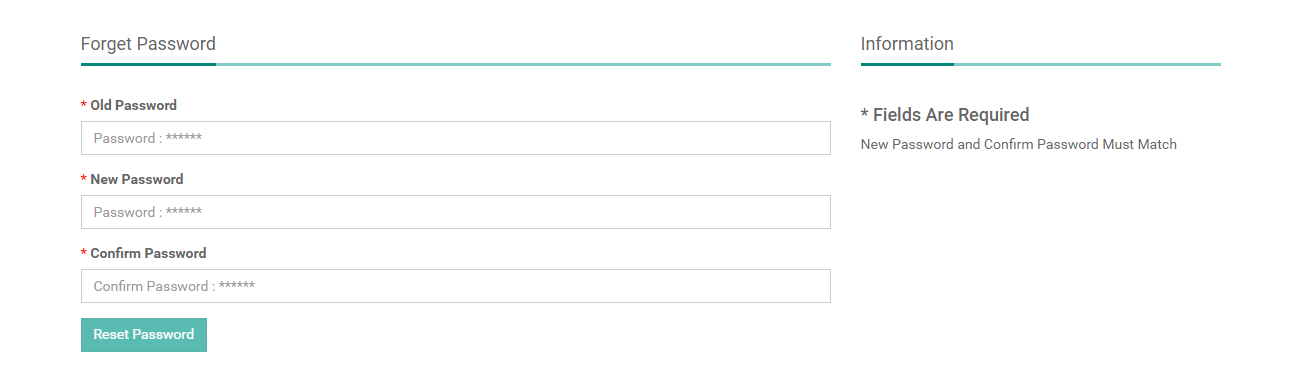
#### Sequence Diagram



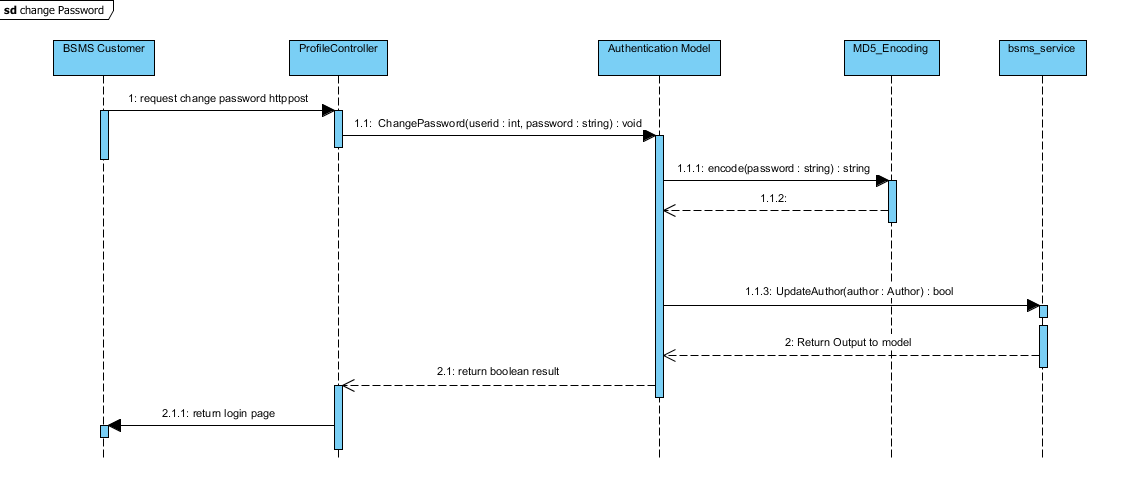
### Change Password

This functionality allow previously created staff to be edited if there exist any wrong information, editing information of staff including the name and other information e.g. changing staff first name from “Joe” to “John”

#### Screen Design



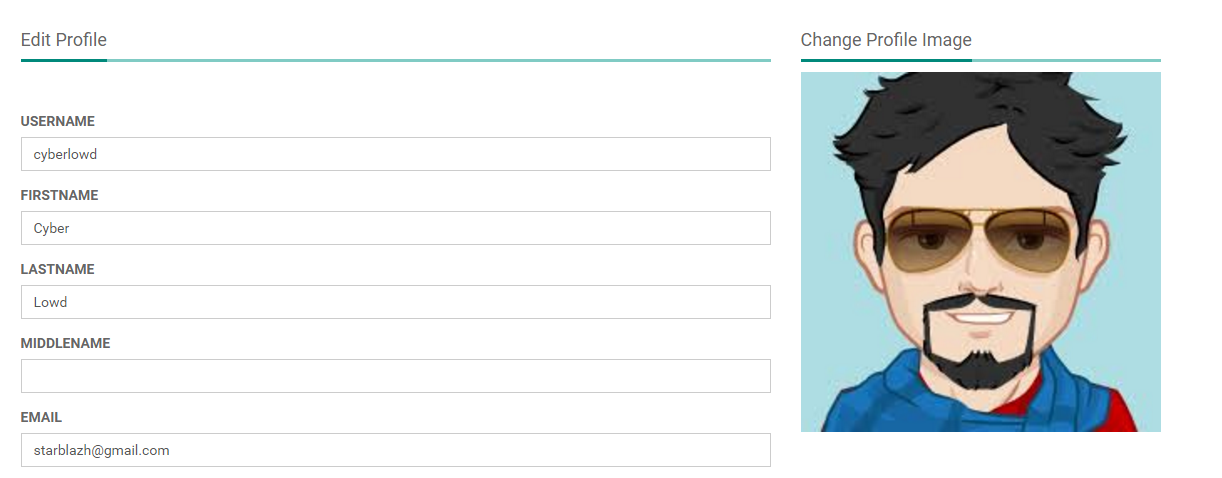
#### Sequence Diagram



### Change Avatar

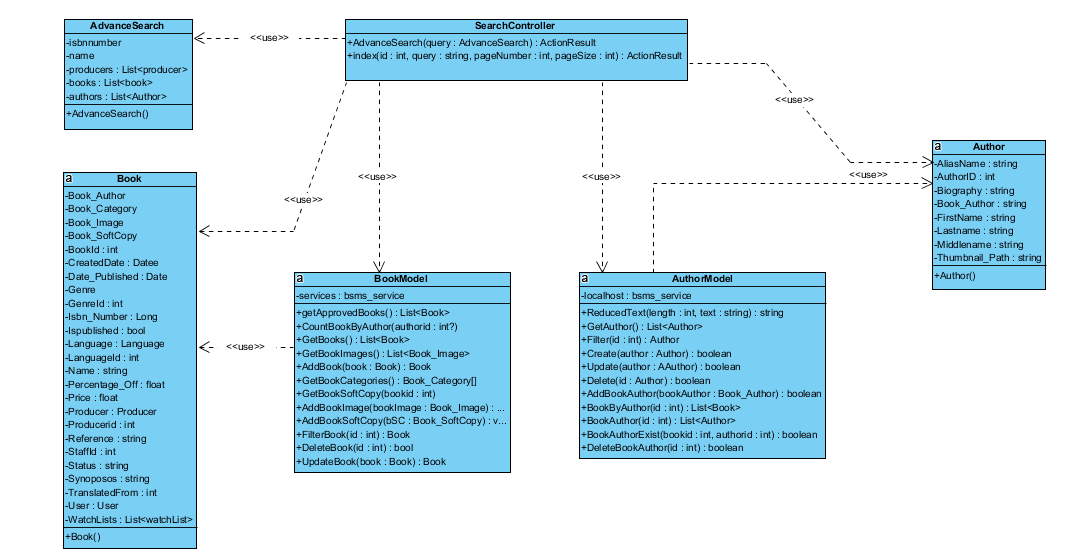
The list staff feature allow the admin use to view sequence staffs that have been added into the system, with each element of the sequence having edit, delete and info icon. On click of each icon triggers equivalent event for each element in that list.

#### Screen Design



## System Search

### Class Diagram



### Class Description

|  |  |  |
| --- | --- | --- |
| Class | Dependency | Description |
| AdvanceSearch | Role | The user class holds all user data and information including all the usernames and passwords in the system, this is known as the BSMS user recognition methodology. |
| Book | N/A | This class is intended to hold unique set of roles for users, since the user uses this class as dependency, user has a role, which is how the system recognise the type of user coming into the system |
| BookModel | N/A | This is a controller class that handles all operations that involving http Operations, once any corresponding http method is call the route engine fires up this controller by requesting it for action. |
| Author | N/A | This is an handler class that handles all session in the system, this class save and retrieves browsing session. |
| AuthorModel | USER | This is a class that handles all operations that involves Authentication including CRUD Operations, thus this class have direct connection to BSMS SOAP Webservices. |
| SearchController | N/A | This class handles system encryption, Md5 password encoding and decoding |

#### Search Controller

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | StaffController | | | |
| **Description** | This class controls everything they deals with Search | | | |
| **Base Class** | System.Web.Mvc.Controller | | | |
| **Constructor** | N/A | | | |
| **Prototype** |  | | | |
| **Source File** | SearchController.cs | | | |
| **Namespace** | BSMS.Controller | | | |
| **Methods** | Name | Input | Output | Description |
| Index | Int, String, Int, Int | ActionResult | This method make a request to the Http Get in order to request for a page, this method handles simple search |
| AdvanceSearch | int | ActionResult | This method make a request to the Http Get in order to request for a page, this method handles Advance search |

#### AuthorModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | AuthorModel | | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | | |
| **Base Class** | Object | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | AuthorModel.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Attributes** | Name | Type | Description | |
| localhost | Bsms\_services | A private instance of bsms soap web services | |
| **Methods** | Name | Input | Output | Description |
| ReducedText | Int , String | String | This method reduces string values to the number of length passed as parameter to it |
| GetAuthors | N/A | List<Author> | This helper method is used to get all record of author from the soap web services |
|  | Create | Author | Boolean | This helper method is used when creating a new author, which incase if parameter passed is valid then it performs the add functionality on bsms dastabase and returns true and false otherwise |
|  | Update | Author | Boolean | This helper method is used when updating an existing author, which incase if parameter passed is valid then it performs update functionality on bsms dastabase and returns true and false otherwise |
|  | Delete | Int | Boolean | this is a static internal type of method, which allows existing category to be deleted as the name Delete implies |
|  | AddBookAuthor | Book\_Author | Boolean | This helper method is used when creating a new book author, which incase if parameter passed is valid then it performs the add functionality on bsms dastabase and returns true and false otherwise |
|  | BookByAuthor | Int | List<Book> | This Author model static method takes as input, id of author and returns back to invoking class list of book written by the author, if author id is incorrect they system returns and empty list. |
|  | BookAuthor | Int | List<Author> | The static method takes as input id of book and returns back to the invoking parent a sequence of author which is the author that wrote the book |
|  | BookAuthorExist | Int, Int | Boolean | The main intention of creating this method in author is to verify if book author have previously exist in the system. This method takes in book id and author id and check if they exist or not, as the name implies. |
|  | DeleteBookAuthor | Int, Int | Boolean | This method carries out the delete operation for Book Author as a result of invoking this method, book author information that match the invoked parameters will be permanently delete from bsms. |

#### Author

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Author | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | |
| **Base Class** | Object | | |
| **Constructor** | N/A | | |
| **Prototype** |  | | |
| **Source File** | Author.cs | | |
| **Namespace** | BSMS.bsms.localhost | | |
| **`** | Name | Type | Description |
| aliasName | Book | Author’s alias name |
| authorId | Int | Author’s unique identifier |
|  | biography | Int | Biography of author, this might contains the background history of this author. |
| Book\_author | String | This is a controller http post method that handles edit functionalities for genres |
|  | firstName | String | Author first name |
|  | lastName | String | Author last name |
|  | middleName | String | Author’s Middle name |
|  | Thumbnail\_path | String | Author image path in the web server |

#### BookModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | bookModel | | | |
| **Description** | This class provide connection to BSMS SOAP services, CRUD operations will be handled by this model class | | | |
| **Base Class** | Object | | | |
| **Constructor** | Default | | | |
| **Prototype** |  | | | |
| **Source File** | BookModel.cs | | | |
| **Namespace** | BSMS.Model | | | |
| **Attributes** | Name | Type | Description | |
| service | Bsms\_services | A private instance of bsms soap web services | |
| **Methods** | Name | Input | Output | Description |
| GetApprovedBooks | N/A | List<Book> | This method query data from web services and return all books that have been approved |
| CountBookByAuthor | Int | Int | This helper method is used to get the amount of book that have been written by a particular author. |
|  | GetBooks | N/A | List<Book> | This model method main function is to get all books from the web service without filter |
|  | AddBook | Book | Boolean | Model method that performs add functionality into the system, by call the add book web method as describe in the web method description below |
|  | GetBookCategories | N/A | List<BookCategory> | This is a static internal type of method, which get all book categories from the web services without any filter added. |
|  | GetBookSoftCopy | N/A | List<Book\_SoftCopy> | This is a static internal type of method, which get all book book soft copy from the web services without any filter added. |
|  | AddBookImage | Book\_Image | Boolean | This Book model static method takes as input, the book image object and returns back to invoking parent Boolean result, if book image is successfully added in the system returns true and false otherwise. |
|  | AddBookSoftCopy | Book\_SoftCopy | Boolean | This Book model static method takes as input, the book soft copy object and returns back to invoking parent Boolean result, if object is successfully added in the system returns true and false otherwise. |
|  | FilterBook | Int | Book | The aim of this model method is to search for a particular book in the system, and returns book that match the given id, null otherwise |
|  | DeleteBook | Int | Boolean | This method carries out the delete operation for Book as a result of invoking this method, book information that match the invoked parameters will be permanently delete from bsms. |
|  | UpdateBook | Book | Boolean | This method carries out the update operation on book. As a result of invoking this method, book information that match the invoked parameters will be permanently updated and changes committed to bsms database. |

#### AdvanceSearch

|  |  |  |  |
| --- | --- | --- | --- |
| Class | AdvanceSearch | | |
| **Description** | This class contains list of books, author, producer and other information that are need in an advance search | | |
| **Base Class** | Object | | |
| **Constructor** | Default | | |
| **Prototype** |  | | |
| **Source File** | AdvanceSearch.cs | | |
| **Namespace** | BSMS.Entity | | |
| **Attributes** | Name | Type | Description |
| isbnnumber | int | This is a unique value for all books |
| name | String | This attribute contains the values of the searched keyword, it might be the name or description, synopsis, biography of a producer, book or author respectively. |
|  | producers | Int | This store the list of producers that match the given search criteria |
| books | string | This store the list of book that match the given search criteria |
|  | authors | int | This store the list of author that match the given search criteria |

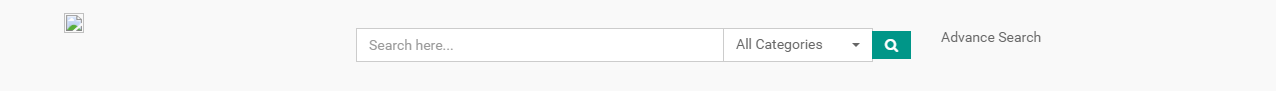
#### Book

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Book | | |
| **Description** | this class holds all book information including name, and synopsis | | |
| **Base Class** | Object | | |
| **Constructor** | Default | | |
| **Prototype** |  | | |
| **Source File** | Book.cs | | |
| **Namespace** | BSMS.bsms.localhost | | |
| **Attributes** | Name | Type | Description |
| Book\_Author | Book\_Author | Book author information |
| Book\_Category | Book\_Category | Book category information |
| Book\_Image | Book\_Image | Book image information |
| BookId | Int | This field contain the unique identifier for each book |
| CreatedDate | Date | Book created date time stamp |
|  | Date\_Published | Date | Book date publish time stamp |
|  | Genre | Genre | Book genre information |
|  | Genreid | Int | Book genre unique id |
|  | Isbn\_number | BigInt | Book isbn number |
|  | IsPublished | Boolean | Boolean value to confirm if book is published or not. |
|  | Language | Language | Book language object that stores the language the book is written in. |
|  | LanguageId | Int | Book language id |
|  | Name | String | Name of the book |
|  | Percentage\_Off | Float | Percentage off from the book |
|  | Price | Float | Unit / Original price of the book |
|  | Producer | Producer | Book producer information |
|  | ProducerId | Int | Book producer id |
|  | Reference | String | Web link that contains reference to the given book |
|  | StaffId | Int | Book staff id |
|  | Status | String | Book status “Approve” or “Not Approve” |
|  | Synopsis | String | Book synopsis information |
|  | TranslatedFrom | Int | Book translated from |
|  | User | User | This field store the id of the staff whom add this book into bsms |
|  | WatchLists | List<WatchList> | List of wishlist |

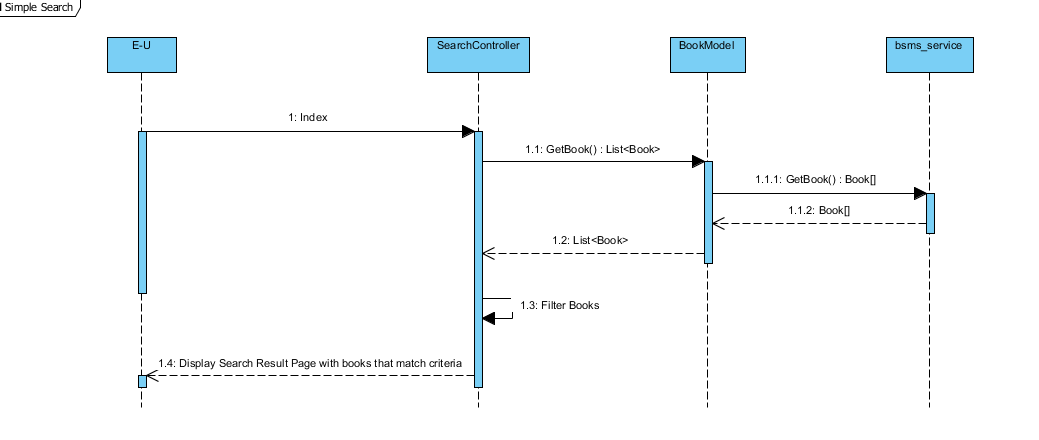
### Simple Search

The list staff feature allow the admin use to view sequence staffs that have been added into the system, with each element of the sequence having edit, delete and info icon. On click of each icon triggers equivalent event for each element in that list.

#### Screen Design



#### Sequence Diagram



### Advance Search

This feature is only allowed of the admin users to delete staffs from the system, this feature is useful in the case where admin user added wrong/duplicate/unnecessary category, and this feature will allow admin to remit such mistake.

#### Screen Design

# Web Service API

## Web Service and Method

|  |  |
| --- | --- |
| Services | Operations |
| BSMS\_SERVICES | AddAuthor  AddUser  AddWishList  DeleteAuthor  DeleteBook  DeleteBookAuthor  DeleteBookCategory  DeleteBookImage  DeleteBookSoftCopy  DeleteCategory  DeleteGenre  DeleteProducer  DeleteUser  DeleteWishList  GetAllInJson  GetAllWishList  GetAuthors  GetBookAuthors  GetBookCategories  GetBookImages  GetBookSoftCopy  GetBooks  GetCategories  GetGenres  GetLanguages  GetProducers  GetRoles  GetUsers  GetWishList  InserCategory  InserGenre  InsertBook  InsertBookAuthor  InsertBookCategory  InsertBookImage  InsertBookSoftCopy  InsertProducer  UpdateAuthor  UpdateBook  UpdateBookAuthor  UpdateBookCategory  UpdateBookImage  UpdateBookSoftCopy  UpdateCategory  UpdateGenre  UpdateProducer  UpdateUser |

## Web Method Description

### AddAuthor

Insert a new Author into the bsms azure database

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| author | AUTHOR | Author’s information to be added. |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | Return true if the author is added into the database and false otherwise |

### AddUser

Insert a new User into the bsms azure database

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| user | USER | User information to be added. |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | Return true if the user is added into the database and false otherwise |

### AddWishList

Insert a new wish list into the bsms azure database

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| wishlist | WishList | This web method will be called when customers request to add a new wishlist to their account |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | Return true if the wishlish is added into the database and false otherwise |

### DeleteAuthor

Delete web method of bsms, this web method create permanent changes to the database, which cannot be reverted by any means once successful

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| authorid | Integer | This web method is meant to trigger delete action on the database, and if this method is triggered correctly and return true, permanent changes will be made to the database. |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | Return true if the author is removed from the database and false otherwise |

### DeleteBook

Delete web method of bsms, this web method create permanent changes to the database, which cannot be reverted by any means once successful

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| bookid | Integer | This web method is meant to trigger delete action on the database, and if this method is triggered correctly and return true, permanent changes will be made to the database. |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | Return true if the author is removed from the database and false otherwise |

### DeleteBookAuthor

Delete web method of bsms, this web method create permanent changes to the database, which cannot be reverted by any means once successful

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| Book\_authorid | Integer | This web method is meant to trigger delete action on the database, and if this method is triggered correctly and return true, permanent changes will be made to the database. |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | Return true if the author is removed from the database and false otherwise |

### DeleteBookCategory

Delete web method of bsms, this web method create permanent changes to the database, which cannot be reverted by any means once successful

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| Book\_category | Integer | This web method is meant to trigger delete action on the database, and if this method is triggered correctly and return true, permanent changes will be made to the database. |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | Return true if the author is removed from the database and false otherwise |

### DeleteBookImage

Delete web method of bsms, this web method create permanent changes to the database, which cannot be reverted by any means once successful

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| Book\_imageid | Integer | This web method is meant to trigger delete action on the database, and if this method is triggered correctly and return true, permanent changes will be made to the database. |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | Return true if the author is removed from the database and false otherwise |

### DeleteBookSoftCopy

Delete web method of bsms, this web method create permanent changes to the database, which cannot be reverted by any means once successful

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| Book\_softcopy\_id | Integer | This web method is meant to trigger delete action on the database, and if this method is triggered correctly and return true, permanent changes will be made to the database. |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | Return true if the author is removed from the database and false otherwise |

### DeleteCategory

Delete web method of bsms, this web method create permanent changes to the database, which cannot be reverted by any means once successful

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| categoryid | Integer | This web method is meant to trigger delete action on the database, and if this method is triggered correctly and return true, permanent changes will be made to the database. |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | Return true if the author is removed from the database and false otherwise |

### DeleteGenre

Delete web method of bsms, this web method create permanent changes to the database, which cannot be reverted by any means once successful

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| genreid | Integer | This web method is meant to trigger delete action on the database, and if this method is triggered correctly and return true, permanent changes will be made to the database. |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | Return true if the author is removed from the database and false otherwise |

### DeleteProducer

Delete web method of bsms, this web method create permanent changes to the database, which cannot be reverted by any means once successful

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| producerid | Integer | This web method is meant to trigger delete action on the database, and if this method is triggered correctly and return true, permanent changes will be made to the database. |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | Return true if the author is removed from the database and false otherwise |

### DeleteUser

Delete web method of bsms, this web method create permanent changes to the database, which cannot be reverted by any means once successful

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| userid | Integer | This web method is meant to trigger delete action on the database, and if this method is triggered correctly and return true, permanent changes will be made to the database. |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | Return true if the author is removed from the database and false otherwise |

### DeleteWishList

Delete web method of bsms, this web method create permanent changes to the database, which cannot be reverted by any means once successful

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| wishlistid | Integer | This web method is meant to trigger delete action on the database, and if this method is triggered correctly and return true, permanent changes will be made to the database. |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | Return true if the author is removed from the database and false otherwise |

### GetAllWishList

Get web method bsms services, this have no input parameter and returns sequence of initialised entities values

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| N/A | N/A | N/A |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| List<WishList> | List of wishlist is an ordered sequence of wishlist sort according to the natural order, which is the order in which they were added into bsms db. |

### GetAuthors

Get web method bsms services, this have no input parameter and returns sequence of initialised entities values

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| N/A | N/A | N/A |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| List<Author> | List of author is an ordered sequence of authors sort according to the natural order, which is the order in which they were added into bsms db. |

### GetBookAuthors

Get web method bsms services, this have no input parameter and returns sequence of initialised entities values

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| N/A | N/A | N/A |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| List<BookAuthor> | List of book author is an ordered sequence of book sort according to the natural order, which is the order in which they were added into bsms db. |

### GetBookCategories

Get web method bsms services, this have no input parameter and returns sequence of initialised entities values

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| N/A | N/A | N/A |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| List<BookCategory > | List of book category is an ordered sequence of book category sort according to the natural order, which is the order in which they were added into bsms db. |

### GetBookImages

Get web method bsms services, this have no input parameter and returns sequence of initialised entities values

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| N/A | N/A | N/A |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| List<BookImage> | List of book image is an ordered sequence of book image sort according to the natural order, which is the order in which they were added into bsms db. |

### GetBookSoftCopy

Get web method bsms services, this have no input parameter and returns sequence of initialised entities values

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| N/A | N/A | N/A |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| List<BookSoftCopy> | List of book soft copy is an ordered sequence of book soft copy sort according to the natural order, which is the order in which they were added into bsms db. |

### GetBooks

Get web method bsms services, this have no input parameter and returns sequence of initialised entities values

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| N/A | N/A | N/A |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| List<Book> | List of book is an ordered sequence of book sort according to the natural order, which is the order in which they were added into bsms db. |

### GetCategories

Get web method bsms services, this have no input parameter and returns sequence of initialised entities values

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| N/A | N/A | N/A |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| List<Category> | List of category is an ordered sequence of book sort according to the natural order, which is the order in which they were added into bsms db. |

### GetGenres

Get web method bsms services, this have no input parameter and returns sequence of initialised entities values

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| N/A | N/A | N/A |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| List<Genre> | List of genre is an ordered sequence of genre sort according to the natural order, which is the order in which they were added into bsms db. |

### GetLanguages

Get web method bsms services, this have no input parameter and returns sequence of initialised entities values

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| N/A | N/A | N/A |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| List<Language> | List of language is an ordered sequence of language sort according to the natural order, which is the order in which they were added into bsms db. |

### GetProducers

Get web method bsms services, this have no input parameter and returns sequence of initialised entities values

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| N/A | N/A | N/A |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| List<Producer> | List of producer is an ordered sequence of producer sort according to the natural order, which is the order in which they were added into bsms db. |

### GetRoles

Get web method bsms services, this have no input parameter and returns sequence of initialised entities values

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| N/A | N/A | N/A |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| List<Book> | List of role is an ordered sequence of role sort according to the natural order, which is the order in which they were added into bsms db. |

### GetUsers

Get web method bsms services, this have no input parameter and returns sequence of initialised entities values

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| N/A | N/A | N/A |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| List<User> | List of user is an ordered sequence of user sorted according to the natural order, which is the order in which they were added into bsms db. |

### GetWishList

Get web method bsms services, this have no input parameter and returns sequence of initialised entities values

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| UserID | Integer | N/A |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| List<WishList> | This web method return wish list that belongs to a particular user which id is input, and returns null if the user id is invalid or does not exist. |

### InsertCategory

Insert method of bsms services, this method handles insert functions, this web method communicate with bsms database using the linq to sql.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| Category | Category | This web method accept category object as parameter, which as a result will be added into the database |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | This web method return Boolean result, true if the soap service method run under a normal case which as a result have been added into the database and false otherwise. |

### InserGenre

Insert method of bsms services, this method handles insert functions, this web method communicate with bsms database using the linq to sql.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| Genre | Genre | This web method accept genre object as parameter, which as a result will be added into the database |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | This web method return Boolean result, true if the soap service method run under a normal case which as a result have been added into the database and false otherwise. |

### InsertBook

Insert method of bsms services, this method handles insert functions, this web method communicate with bsms database using the linq to sql.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| book | Book | This web method accept book object as parameter, which as a result will be added into the database |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | This web method return Boolean result, true if the soap service method run under a normal case which as a result have been added into the database and false otherwise. |

### InsertBookAuthor

Insert method of bsms services, this method handles insert functions, this web method communicate with bsms database using the linq to sql.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| bookAuthor | BookAuthor | This web method accept book author object as parameter, which as a result will be added into the database |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | This web method return Boolean result, true if the soap service method run under a normal case which as a result have been added into the database and false otherwise. |

### InsertBookImage

Insert method of bsms services, this method handles insert functions, this web method communicate with bsms database using the linq to sql.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| bookImage | BookImage | This web method accept book image object as parameter, which as a result will be added into the database |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | This web method return Boolean result, true if the soap service method run under a normal case which as a result have been added into the database and false otherwise. |

### InsertBookSoftCopy

Insert method of bsms services, this method handles insert functions, this web method communicate with bsms database using the linq to sql.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| bookSoftCopy | BookSoftCopy | This web method accept book soft copy object as parameter, which as a result will be added into the database |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | This web method return Boolean result, true if the soap service method run under a normal case which as a result have been added into the database and false otherwise. |

### InsertProducer

Insert method of bsms services, this method handles insert functions, this web method communicate with bsms database using the linq to sql.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| Producer | Producer | This web method accept producer object as parameter, which as a result will be added into the database |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | This web method return Boolean result, true if the soap service method run under a normal case which as a result have been added into the database and false otherwise. |

### UpdateAuthor

Update web method of bsms that handles edit functionality for its equivalent entities, this web method makes permanent changes to the database and such changes cannot be reverted by any means.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| Author | author | This web method accept author object as parameter, which as a result will be update and changes made to the database |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | This web method return Boolean result, true if the soap service method run under a normal case which as a result have been added into the database and false otherwise. |

### UpdateBook

Update web method of bsms that handles edit functionality for its equivalent entities, this web method makes permanent changes to the database and such changes cannot be reverted by any means.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| Book | book | This web method accept book object as parameter, which as a result will be update and changes made to the database |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | This web method return Boolean result, true if the soap service method run under a normal case which as a result have been added into the database and false otherwise. |

### UpdateBookAuthor

Update web method of bsms that handles edit functionality for its equivalent entities, this web method makes permanent changes to the database and such changes cannot be reverted by any means.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| bAuthor | Book\_Author | This web method accept book author object as parameter, which as a result will be update and changes made to the database |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | This web method return Boolean result, true if the soap service method run under a normal case which as a result have been added into the database and false otherwise. |

### UpdateBookCategory

Update web method of bsms that handles edit functionality for its equivalent entities, this web method makes permanent changes to the database and such changes cannot be reverted by any means.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| BookCategory | Book\_Category | This web method accept book category object as parameter, which as a result will be update and changes made to the database |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | This web method return Boolean result, true if the soap service method run under a normal case which as a result have been added into the database and false otherwise. |

### UpdateBookImage

Update web method of bsms that handles edit functionality for its equivalent entities, this web method makes permanent changes to the database and such changes cannot be reverted by any means.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| BookImage | Book\_Image | This web method accept book image object as parameter, which as a result will be update and changes made to the database |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | This web method return Boolean result, true if the soap service method run under a normal case which as a result have been added into the database and false otherwise. |

### UpdateBookSoftCopy

Update web method of bsms that handles edit functionality for its equivalent entities, this web method makes permanent changes to the database and such changes cannot be reverted by any means.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| BookSoftCopy | Book\_SoftCopy | This web method accept book soft copy object as parameter, which as a result will be update and changes made to the database |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | This web method return Boolean result, true if the soap service method run under a normal case which as a result have been added into the database and false otherwise. |

### UpdateGenre

Update web method of bsms that handles edit functionality for its equivalent entities, this web method makes permanent changes to the database and such changes cannot be reverted by any means.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| Genre | Genre | This web method accept genre object as parameter, which as a result will be update and changes made to the database |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | This web method return Boolean result, true if the soap service method run under a normal case which as a result have been added into the database and false otherwise. |

### UpdateProducer

Update web method of bsms that handles edit functionality for its equivalent entities, this web method makes permanent changes to the database and such changes cannot be reverted by any means.

#### Parameters

|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| Producer | Producer | This web method accept producer object as parameter, which as a result will be update and changes made to the database |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | This web method return Boolean result, true if the soap service method run under a normal case which as a result have been added into the database and false otherwise. |

### UpdateUser

Update web method of bsms that handles edit functionality for its equivalent entities, this web method makes permanent changes to the database and such changes cannot be reverted by any means.

#### Parameters

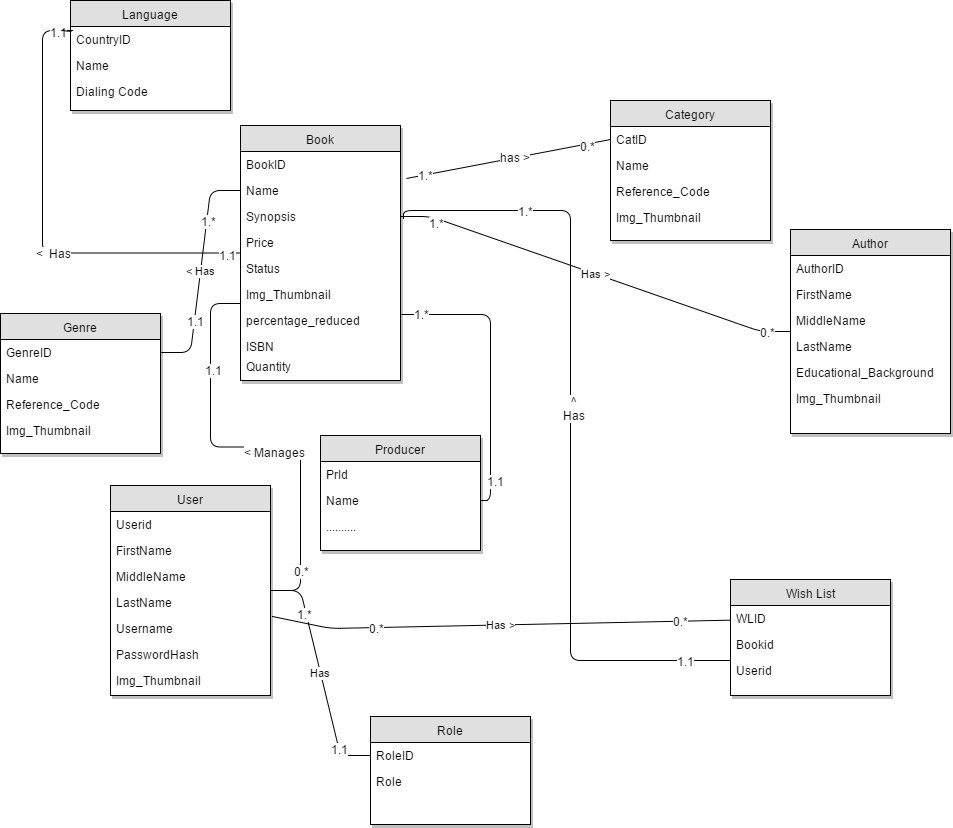
|  |  |  |
| --- | --- | --- |
| Input Name | Parameters | Description |
| User | User | This web method accept user object as parameter, which as a result will be update and changes made to the database |

#### Response

|  |  |
| --- | --- |
| Output | Output Description |
| Boolean | This web method return Boolean result, true if the soap service method run under a normal case which as a result have been added into the database and false otherwise. |

# DATABASE DESIGN

## Entity Relationship Diagram



## Database Diagram

## Table Descriptions

### User

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Attribute | Type | Constraints | Description |
| 1 | USERID | INT | Primary key | User’s unique identifier |
| 2 | USERNAME | VARCHAR | Not null |  |
| 3 | PASSWORDHASH | VARCHAR | Not Null | This contains the password hash of users in MD5 format |
| 4 | FIRSTNAME | VARCHAR | Not null | User’s name |
| 5 | LASTNAME | VARCHAR | N/A | User’s name |
| 6 | MIDDLENAME | VARCHAR | N/A | User’s name |
| 7 | THUMBNAIL\_PATH | VARCHAR |  | This attribute contains user’s image path |
| `8 | EMAIL | VARCHAR | Not null | User’s unique email address |
| 9 | ROLEID | INT | Foreign Key | Role foreign key, this is how the system recognise different type of users |

### Role

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Attribute | Type | Constraints | Description |
| 1 | ROLEID | INT | Primary Key | Role unique identifier, auto increment field |
| 2 | ROLE | VARCHAR | Not Null | Role name |

### Author

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Attribute | Type | Constraints | Description |
| 1 | AUTHORID | INT | Primary Key | Author unique identifier, auto increment field |
| 2 | FIRSTNAME | VARCHAR | Not null | Author’s First name |
| 3 | LASTNAME | VARCHAR | Not Null | Author’s Last name |
| 4 | MIDDLENAME | VARCHAR | N/A | Author’s Middle name |
| 5 | BIOGRAPHY | VARCHAR | Not null | Full detail about author |
| 6 | ALIASNAME | VARCHAR | N/A | Author’s nick, prefix or any name the is popularly use to refer to such author |
| 7 | THUMBNAIL\_PATH | VARCHAR | Not Null | Image path of author is store in this attribute |

### Category

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Attribute | Type | Constraints | Description |
| 1 | CATEGORYID | INT | Primary Key | Category unique identifier, auto increment field |
| 2 | NAME | VARCHAR | Not null | Category name |
| 3 | DESCRIPTION | VARCHAR | Not null | Category description |
| 4 | REFERENCE\_KEY | VARCHAR | N/A | The reference key set the link to each category |
| 5 | CATEGORY\_THUMBNAIL | VARCHAR | Not Null | Image path of category is store in this attribute |

### Genre

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Attribute | Type | Constraints | Description |
| 1 | GENREID | INT | Primary Key | Genre unique identifier, auto increment field |
| 2 | NAME | VARCHAR | Not null | Genre name |
| 3 | DESCRIPTION | VARCHAR | Not null | Genre description |
| 4 | REFERENCE\_KEY | VARCHAR | N/A | The reference key set the link to each Genre |

### Book

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Attribute | Type | Constraints | Description |
| 1 | BOOKID | INT | Primary Key | Book unique identifier, auto increment field |
| 2 | ISBN\_NUMBER | BIGINT | Not null | Unique book identifier |
| 3 | NAME | VARCHAR | Not null | Book name |
| 4 | THUMBNAIL\_PATH | VARCHAR | N/A | Book image path address |
| 5 | STATUS | BIT | Null | Book status Null = “Not attended”, 0 = “Rejected” and 1=”Approved” |
| 6 | ISPUBLISHED | BIT | Not Null |  |
| 7 | SYNOPOSIS | VARCHAR | Not null | this holds information of book overview. |
| 8 | REFERENCE | VARCHAR | N/A | This contains information all available references |
| 9 | DATE\_PUBLISHED | DATETIME | N/A | Date this book was published, normally the date admin approves the book |
| 10 | CREATED\_DATE | DATETIME | N/A | Date this book was initially created |
| 11 | PRICE | FLOAT | Not null | Price of this book |
| 12 | PERCENTAGE\_OFF | INT | null | Percentage discount |
| 13 | GENREID | INT | Foreign Key | Book genre |
| 14 | STAFFID | INT | Foreign Key | Staff who controls this book |
| 15 | PRODUCERID | INT | Foreign Key | Producer of this book |
| 16 | LANGUAGEID | INT | Foreign Key | Language id of the language this book is written in. |

### Book\_Category

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Attribute | Type | Constraints | Description |
| 1 | BOOK\_CATEGORYID | INT | Primary Key | Book category unique identifier |
| 2 | BOOKID | INT | Foreign Key | Book id references |
| 3 | CATEGORYID | INT | Foreign Key | Category id references |

### Producer

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Attribute | Type | Constraints | Description |
| 1 | PRODUCTID | INT | Primary Key |  |
| 2 | NAME | INT | Not Null |  |
| 3 | ADDRESS | INT | Not Null |  |
| 4 | CONTACT | VARCHAR | Null |  |
| 5 | EMAIL | VARCHAR | Null |  |

### Book\_SoftCopy

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Attribute | Type | Constraints | Description |
| 1 | BSCID | INT | Primary Key |  |
| 2 | BOOKID | INT | Foreign Key |  |
| 3 | VERSION | FLOAT | Not Null |  |
| 4 | FILEPATH | VARCHAR | Null |  |
| 5 | FILESIZE | INT | Null |  |

### Book\_Author

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Attribute | Type | Constraints | Description |
| 1 | BOOK\_AUTHORID | INT | Primary Key |  |
| 2 | BOOKID | INT | Foreign Key |  |
| 3 | AUTHORID | INT | Foreign Key |  |

### WatchList

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Attribute | Type | Constraints | Description |
| 1 | WATCHLISTID | INT | Primary Key |  |
| 2 | BOOKID | INT | Foreign Key |  |
| 3 | USERID | INT | Foreign Key |  |

### BOOK\_IMAGE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Attribute | Type | Constraints | Description |
| 1 | BOOK\_IMAGEID | INT | Primary Key | Book image unique identifier |
| 2 | BOOKID | INT | Foreign Key | Book unique id (Identity) foreign key references |
| 3 | IMAGEPATH | VARCHAR | N/A | Server location of image |

### LANGUAGE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Attribute | Type | Constraints | Description |
| 1 | LANGUAGEID | INT | Primary Key | Language unique identifier |
| 2 | LANGUAGE | VARCHAR | N/A | Language name |