

**DOKUZ EYLÜL UNIVERSITY FACULTY OF
ENGINEERING DEPARTMENT OF
COMPUTER ENGINEERING**

CME 2210

Object Oriented Analysis and Design

LIBRARY MANAGEMENT SYSTEM

FINAL REPORT

REVISED WITH NEW DESIGN PATTERNS

by

Hasan Berk Kocabaş 2016510046

1 Introduction

1.1	What the problem is	4
1.2	Goal for the Project	4
1.3	Motivation of the Project	4
1.4	Process Flow Prewiev	5

2 Analysis and Design

2.1	Plan for Requierments Engineering.....	5
2.2	Functional Requierments	7
2.3	Non-Functional Requierments.....	8
2.4	Use Cases	12
2.5	Models.....	13

3 Project Plan

3.1	Task Description.....	18
3.2	Task Assignment.....	19
3.3	Deliverables and Milestones.....	19
3.4	Project Schedule.....	19

4 Testing

4.1	Features to be Tested	20
4.2	Test Cases	20
4.3	Testing Schedule.....	20

5 Conclusion

5.1 The Problem and Solution	21
5.2 The Team and the SE Process.....	21
5.3 Engagement of Umbrella Activities.....	21

6 User Manual

6.1 Software Description.....	22
6.2 How to use the Software.....	22
6.3 Troubleshooting Common Problems.....	22

7 REVISED WITH NEW DESIGN PATTERNS

7.1 Definitions of Design Patterns.....	23
7.2 Code Implementations.....	24
7.3 Diagrams.....	25
7.4 Class Implementations.....	27

1 Introduction

1.1 What the Problem is

Universities has huge libraries. That situation getting harder their jobs for employees and the students. There is too many book and too many shelf for books. Library Management System will organize the books and keep their locations for students and employees. The Information about books can be learned from the Library Management System.

1.2 Goals fort he Project

This software, called 'Library Management System', will allow the univercities to Access their database securely and safely in a user-friendly environment. Allowing for them to change book information ease. The software will be in sync with both the Web App, allowing for real-time up-to-date sevicees for their users.

Both registered andd non-registered users will be able to search books by type, writer and any other potential searches. They will also be able to select and check their availability for books.

The software itself will be available on all computer platforms that are can able to connect internet.

1.3 Motivation for the Project

With this company in need of a beter system, we felt it was our obligation to help them in their time of need. To deelop such a system that would not only ease the burden on the company's customers, but company itself. Our team has an immense amounth of knowledge when it comes to problem solving, and communication. Not only would we strive to give the library management system evriting desired, but we will continue to make sure the sotware is at its very best and beyond. Each one of us will always and will continue to give 100% and more to making transition a breeze for each library management system.

1.4 Process Flow Preview

For our process flow, we plan on taking the iterative route, as we find communication essential throughout the development process. In order to learn all aspects of the Project in detail, we feel that contacting the universities and having those in charge be in the same room when the planning is taking place. It is our way of discussing the requirements and develop important notes that will help in constructing the overall feel and idea. The modeling process in its own right we feel is not a start to finish process. There will be times that we may have to go back to certain portions within the modeling activity to ensure a sufficient model. We wish to make sure that if we miss anything, we do not figure that out in the construction stage.

Lastly we find the construction framework activity require communication with the library as well. It would allow for us to get small tweaks out of way as well as have kind of testing process on the interaction between the constructed elements and those that will use the software directly. We feel that allowing for the users to view how the software is made will ease the transition during the deployment stage.

2 Analysis and Design

2.1 Plan For Requirements Engineering

Inception Task:

The first goal is to define the project. We want to analyze the software to be used in the library, user requests, how it is needed, how often the software will be used, and ensure that the end product can handle all library users. These are a few questions we ask ourselves.

There are a few other questions that require a basic understanding of the project:

- What are the main functions?
- What kind of customers are you targeting with this new website, who will use it?
- What do users expect from this website?

Elicitation Task:

Our aim at this stage is to identify the problem, offer solutions and talk about different approaches. In order to get a more refined understanding, the project team holds a meeting on this issue. The plan is to get a thorough idea of what the system's goals are, what to do, and how the overall system fits into the library system. Descriptions of the technical environment, usage scenarios and a list of requirements are currently being created.

Elaboration Task:

Information collected from the initial and screening stages are brought together in this group. A model is executed that clearly shows the conditions of software functions and behavior. Scenarios are created to understand and help the user interact with the website. How each feature and each function interacts with each other should be defined.

Negotiation Task:

The team reviews the requirements and the tasks that are least important are removed from the software.

Specification Task:

During this task, a software requirements specification template is planned to be created. In this template, we will note the overall purpose of the project and the target audience. Explanations about product features, user classes / features, working environment and design will be included. It also includes security requirements, quality features and which interfaces to use with this software. Along with this written document, a "model" will be created to gain a visual perspective to the project.

Validation Task:

At this stage, it should be ensured that the specified requirements are clearly defined. There should be no false comments. All requirements should be in line with the general objectives and should be easily understood.

Requirements Management:

Any changes that may occur during the project phases should be addressed carefully and carefully. Requirements Management takes place throughout the project process, because changes or changes may occur in any case.

2.2 Functional Requirements

Hardware Requirements:

The software can run on any device connected to the web, regardless of the operating system. It does not require anything other than an internet connection.

Website Interface-Primary Tasks:

- **Collection Page**
In this page books are listed. User can search and request for available book.
- **Manager Page**
Manager can manage all users, books and request in this page.
- **Registration**
That page is for the user registration.
- **Login**
That page is allow to user and manager login.
- **Profile Page**
That page is allow the show user informations and edit them. In this page also show user activity history.

Website Interface-Secondary Tasks:

- **Report Page**
In this page all reports are listed. For example: book requests and logs.
- **Apply Page**
For the users allow the book request. Users can see available date interval to the take book. (In the modal)

2.3 Non Functional Requirements

Performance Requirements

- Hibernate ecosystem was used for ORM operations.
- Fast performance / data transmission
- Show updates quickly
- Accurate and efficient imaging on all devices (responsive view)

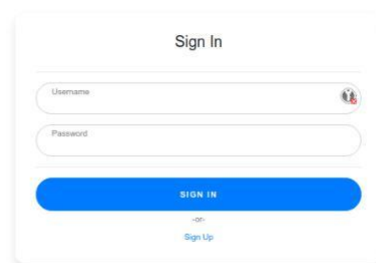
Security Requirements

- Spring security and JWT ecosystem was used for authentication.

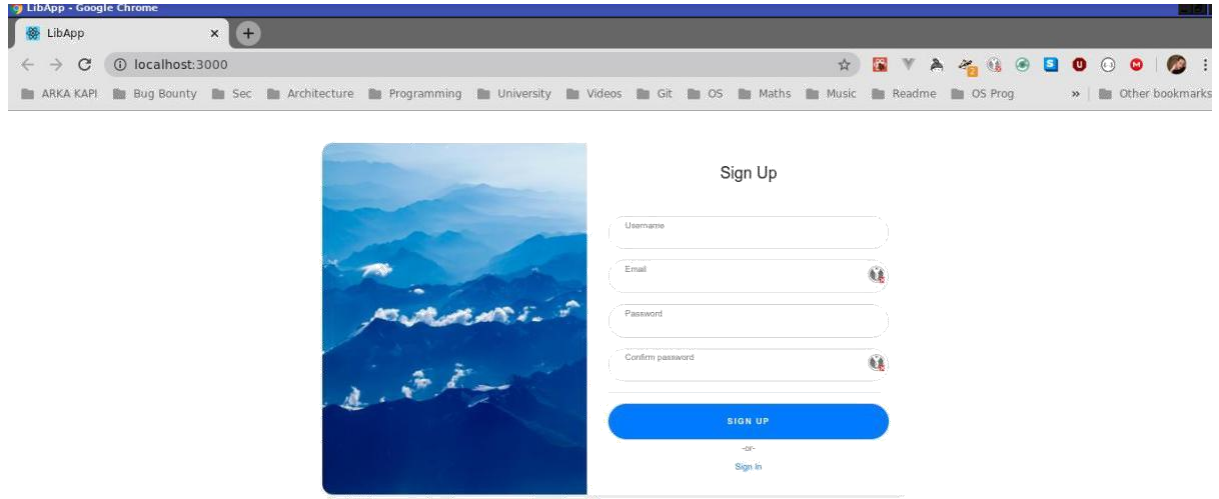
Quality Attributes

- Easy to see and use navigation
- The program must be reliable, support new versions, maintainable for necessary updates






Screenshot Mockups



Sign-in page.



Register page.

Başlık	Yazar/Editor	Yayın evi	Tip	Adet	
 Çırac Dergisi 2. Sayı Baskı: 4 Basım Yılı: 11/11/1988 ISBN: 5579050590	Kolektif	Abakus Kitap	Kitap	2	Detay
 Hackers & Painters Bilgisayar Çağından Büyük Fikirler Baskı: 2 Basım Yılı: 24/06/1992 ISBN: 5285854989	Paul Graham	Odtü	Kitap	1	Detay
 Grasshopper ile Parametrik Modelleme Baskı: 10 Basım Yılı: 29/09/1967 ISBN: 8186098267	Serkan Uysal, Tugrul Yazar	Pusula Yayıncılık	Kitap	2	Detay
 Centos Sistem ve Sunucu Yönetimi Baskı: 4 Basım Yılı: 23/11/1973 ISBN: 1296790681	Deniz Parlak	Koditab	Kitap	3	Detay
 Ethical Hacking Baskı: 5 Basım Yılı: 01/03/2011 ISBN: 2573973905	Ömer Çıtak	Abakus Kitap	Kitap	2	Detay

1 2

Collections page.

Einstein'in Görelilik Kuramı

Tip	Kitap
Yazarlar	Timur Karaçay
ISBN	8341628015
Kategori	Diğer
Baskı	9
Yayın Yılı	30/03/1988
Yayınevi	Abakus Kitap
Kitap Adetli	1
Açıklama	



Kopya No	Lokasyon	Dönüş Tarihi	Durumu
1	İstanbul		Müsait

Book details.

LibraryApp

Anasayfa Kütüphane Disconnectus Erectus ▾

DISCONNECTUS ERECTUS

Profil

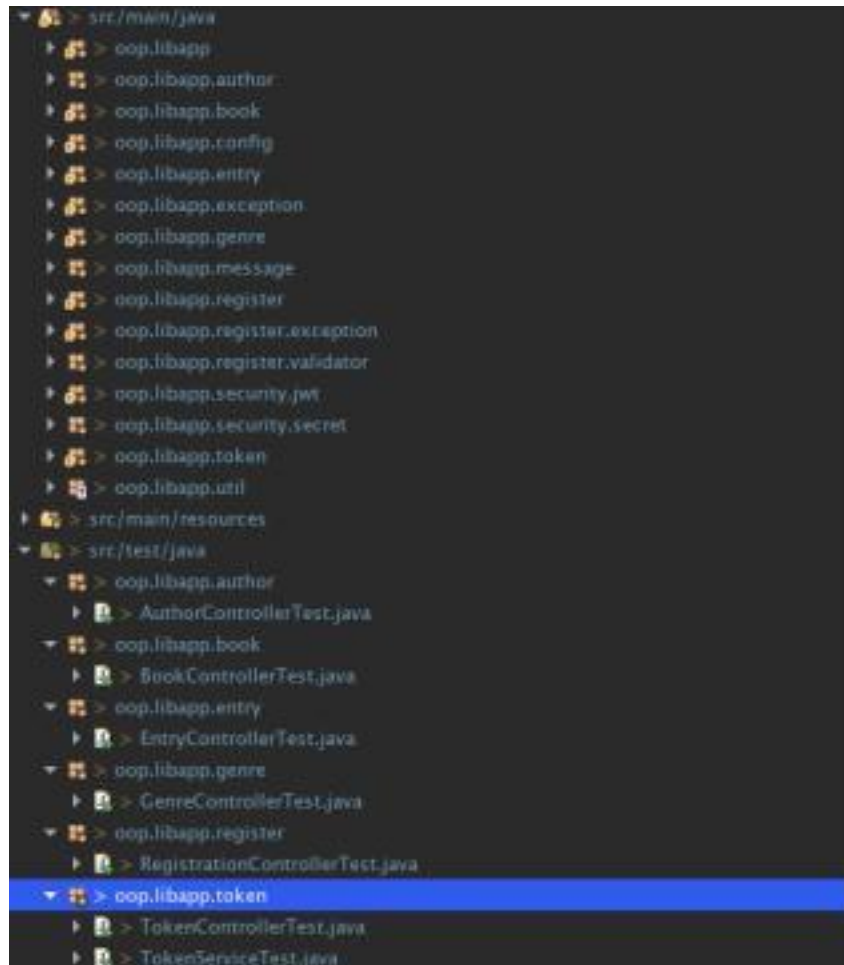
İstek Geçmişi

İstek Geçmişi

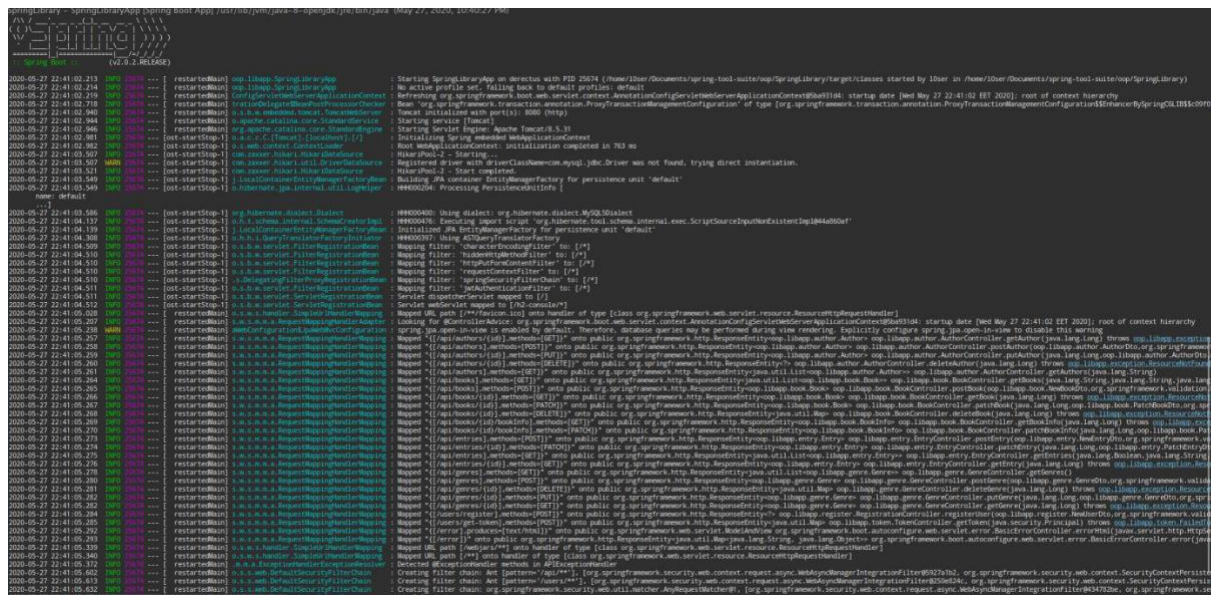
Henüz onaylanmış bir isteğiniz bulunmamaktadır.

Kitap Say

Book request page.



All classes are tested.



Backend initial running and all mapped routes.

2.4 Use Cases

Use Case 1: Checking books

Primary Actor: User

Goal in Content: To see if the book he is looking for is available

Prerequisites: The book should not be on anyone else

Trigger: User searches for books to request.

Scenario:

1. User: Logs on to the library site (enters Username / Password)
2. Book Search: Searches for books
3. Collection Page: Lists and filters all books on the site.
4. User: Requests if the book is available.

Use Case 2: Add Book

Primary Actor: User

Goal in Content: Add book on to the website

Trigger: User request for adding book.

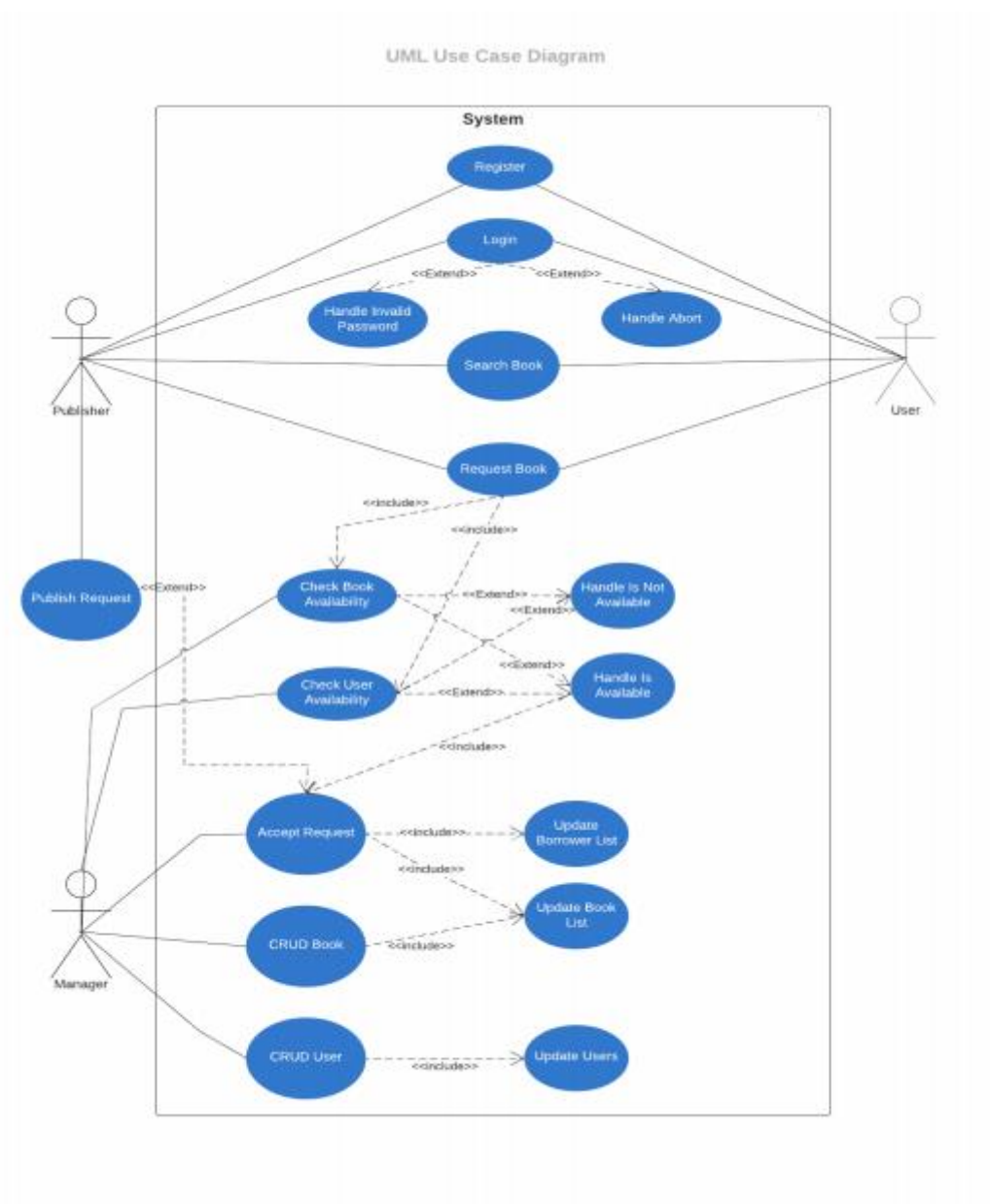
Scenario:

1. User: Logs on to the library site (enters Username / Password)
2. Add Book: Redirect to add book pages
3. Details: User should add details about book.
4. Manager: Manager examine and accept or reject the add request.

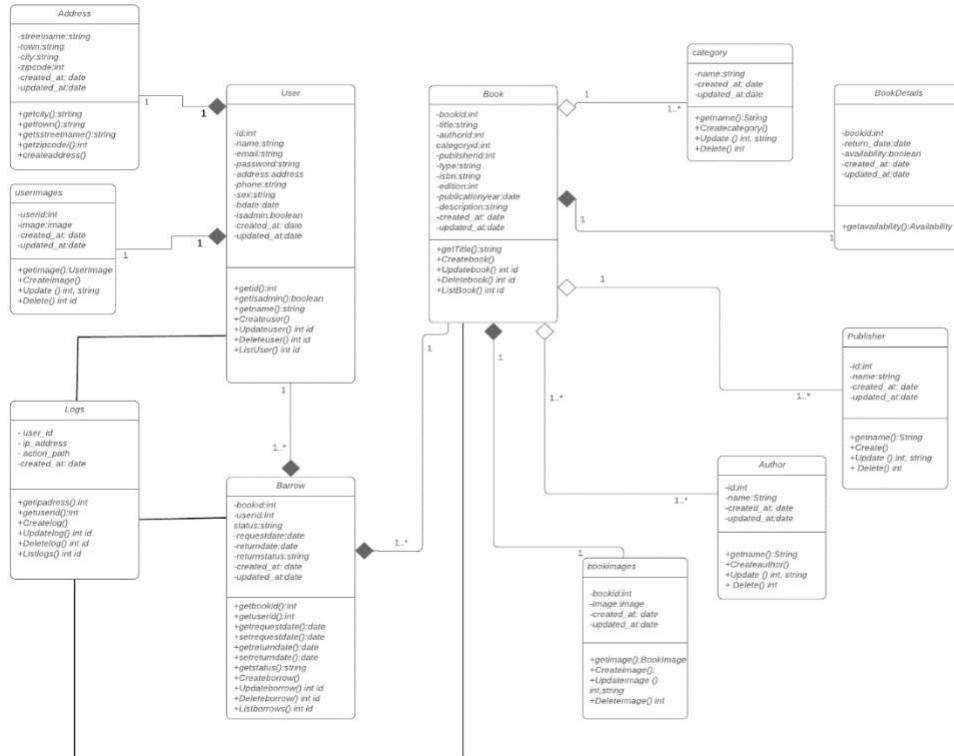
2.5 Models

Use Case Diagram

This diagram shows the abilities of the actors in the program in general. The program has 3 actors which are Manager, Standart User (Generally Students) and Publisher. Standart User has ability for registering, searching and requesting books. Publishers, in addition to Standart User they can publish publication. Managers can access all managerial functions such as deleting, creating, updating and reading all data. Other manager task is the control requests and accept or decline them. Managers perform this task based on book and user status

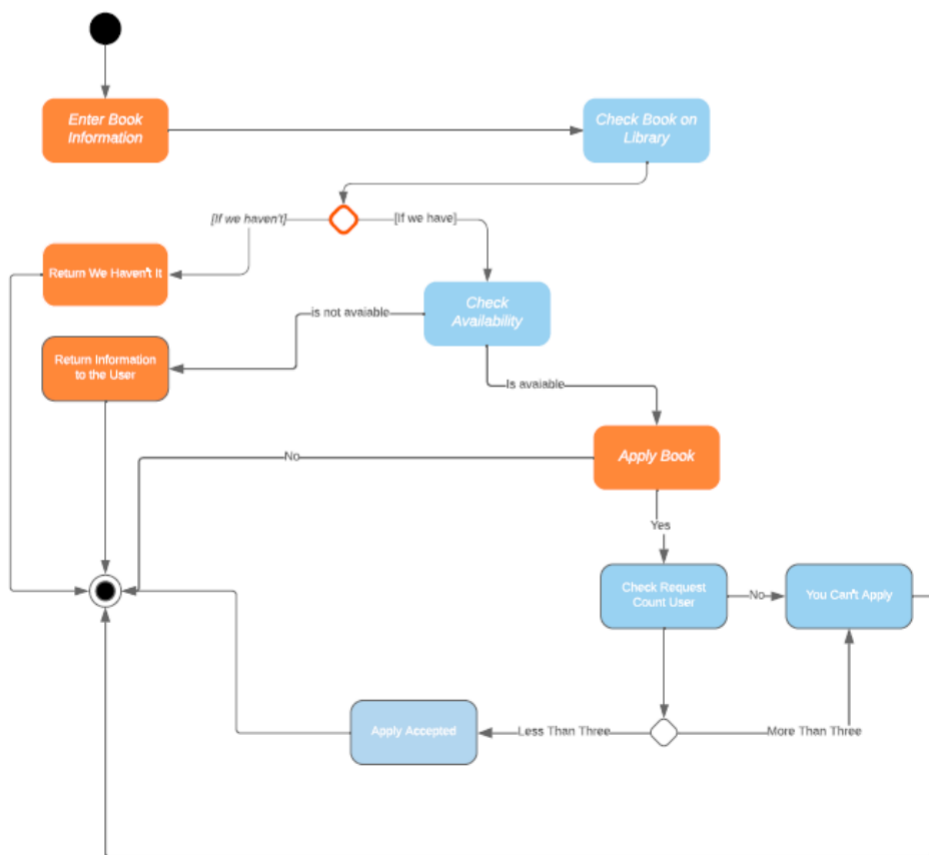
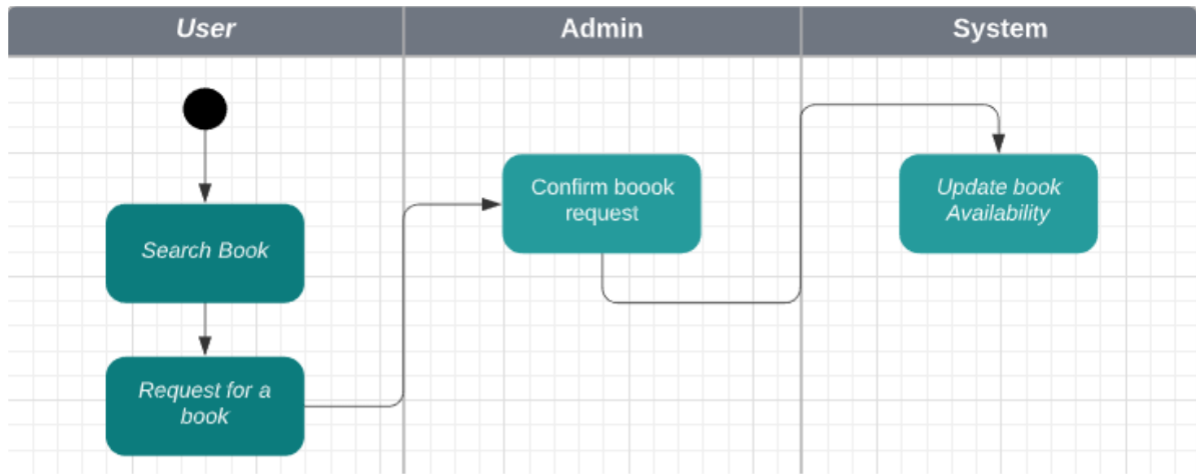


Class Diagram

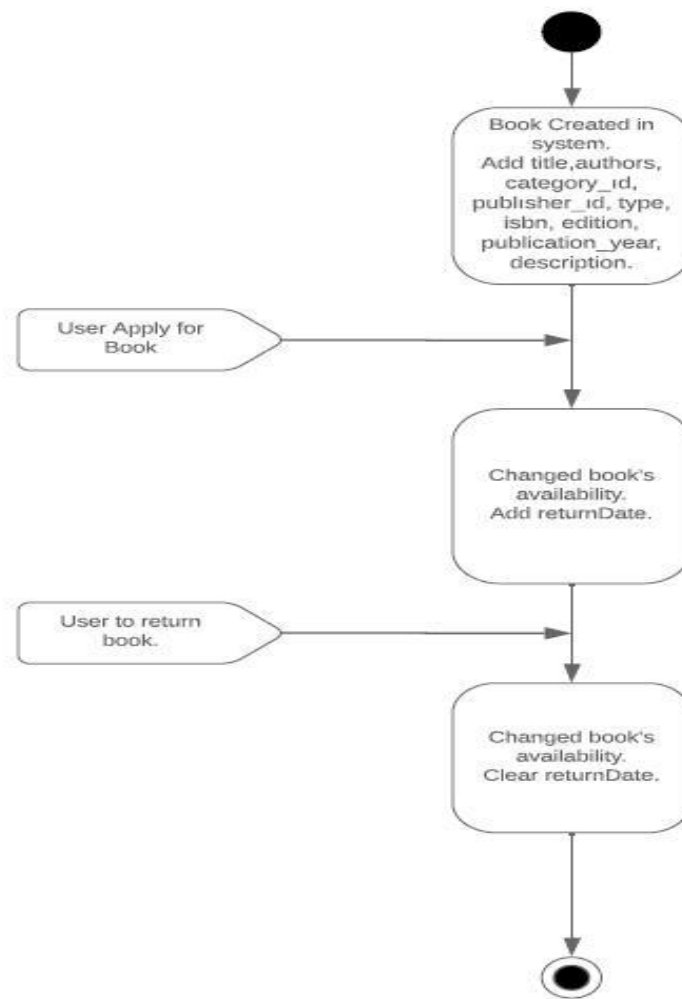


Class diagram is listed above. Each class has its own variables and functions. The features common to several classes are in the abstract class. The function that creates the class outline is in the interface. The User and Book classes use several subclasses because they have extra attributes. While User using attributes such as user address, picture; books have features such as category, details, authors and publisher. The User and book classes behave like master classes. The Logs class keeps records about user and book operations.

Activity Diagram

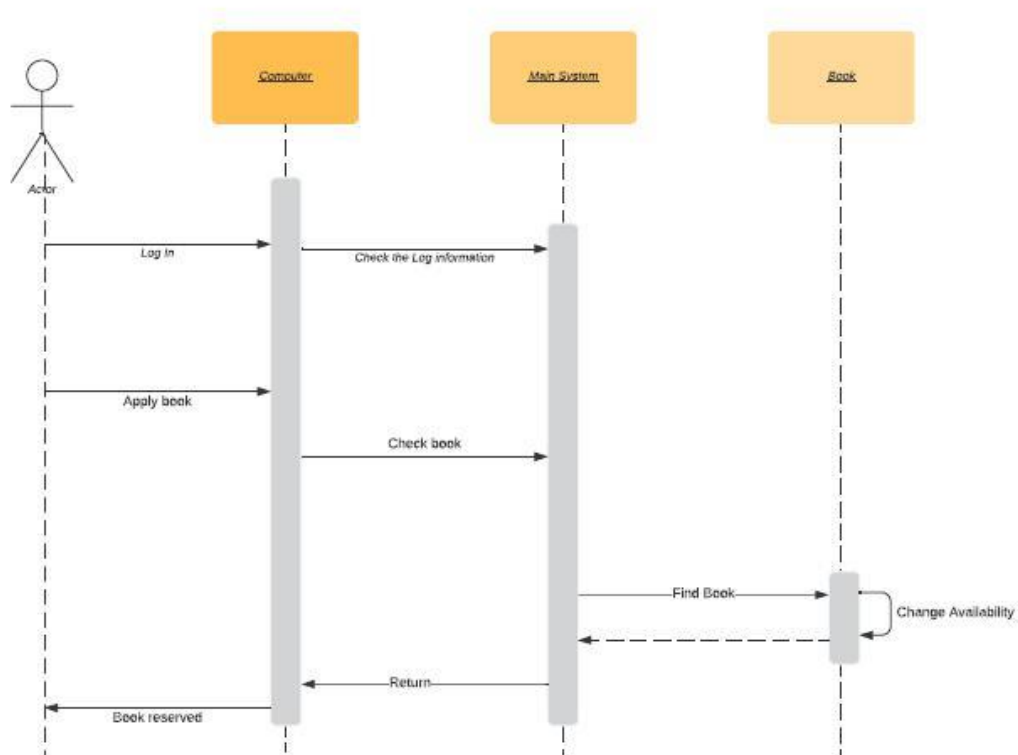


State Diagrams



In the above state machine diagram Worker objects' possible states represented. First Worker object must be created. So worker details should have entered and object have generated. Then customer can visit worker and do some operations that shown in the figure. If worker decides to leave some operations are done.

Sequence Diagram



In this diagram our objects are computer, main system, and book. The first action is logging in to the system by the user, then the system checks the information about the user. If the information is true, the user can apply for books. The second action is applying for a book. In this diagram, we calculate the best case scenario so the user can apply for a book and the system can confirm the book in the system and return the book for borrowing. So the computer returns this information to the user. The book object just changes its availability.

3 Project Plan

3.1 Task Description

Design Model and Mockups:

Designing the models and mockups help to ensure clarity in view of the Project as well as how it works. Univercities are to sit through this process as drawings are created.

Database Creation:

A database is created using the models to provide storage for univercities information, book information and member informations. Testing the database is ensured at this point.

Employee and Members Sofftware Creation:

The software that is to be used by members and employees will be use same application which designed using Java using the guide of mockups, requirements and models. The software will act as a simple and easy to understand user interface to 'browse' and among other functions the database. The information stored including members information , book information and employees information.

Testing :

Testing will be implemented on both the website and software. Test cases may be used to guide and understand the basic actions of both members and employees. Any bugs or errors that occur will be identified and resolved.

Finalization and Reports:

All testing and function processes are finalized at this stage. Reports will be created to ensure all information and functionality is clear in order to maket he user manual and to help ensure employees can use the software with ease.

3.2 Task Assignment :

Assignments were distributed evenly among the group assigned to the Project. All three worked together in the Project planning, sharing in the opportunity of any models and analyzing all specifications made by the proprietor.

All group members were actively involved in all parts of the project. Nuri backend, hasan frontend, mustafa worked predominantly in database and test operations. Nuri was given the tasks of creating the software to using Java .He was also given the objective to test all aspects of software. His findings were reported to each person in group and discussions were made on how to fix them.

Reports were created throughout the process all three group members and gathered to accurately and sufficiently create this final report.

3.3 Deliverables and Milestones:

We had four major Milestones in this Project :

1. Completion of Requirements Gathering.
2. Completion of Design and code.
3. Completion of Testing.
4. Completion of Demonstration.

These milestones were all completed on Schedule and yielded a Deliverable at the end of each.

Our Four corresponding deliverables in this Project were as follows:

1. A List of libraries applications for example.
2. A finished and easily navigational GUI (Graphical user interface).
3. A clean and inmistaked testing phase .
4. Satisfied our customers after demonstration of software.

3.4 Project Schedule

The project started on 26 February and was completed by planning according to the phases given by the lecturers. It is completed in 5 phases in total and the average phase duration is 20 days.

4 Testing

4.1 Features to be tested

Testing is required to ensure that there is no problem in using the software and that it does not cause larger problems for the next steps.

The features we tested are as follows:

- Successful implementation of the app's functions.
- Efficient operation on every device with internet connection.

4.2 Test Cases

The following are examples of test cases we have implemented:

- Unregistered user should not be able to login.
- No request should be made for the book that is not available.
- Books not approved by the manager should not be given to the user
- All pages should be checked for broken links.

4.3 Testing Schedule

The testing should begin right after the project itself begins. Keeping up on testing will ensure that any mistakes are caught early and corrected immediately.

5 Conclusion

5.1 The Problem and Solution

The problem that undergo for Library Management System was a lack of knowledge about Libraries system. We wanted to make our application with user friendly Gui and effective functions. Members should have been able to access and view books without going to library. Request should be transferred directly into the database and displayed on the software .

The solution was to provide Library Management System with user friendly application that would allow for members to access system. The application is designed to stay up to date by giving administrators the ability to change/add/remove any featured books on the app. The application will verify and store any information the user may input when making a book check request. This will appear in real time onto the software used by the employees . This software provides an easy-to-use interface to allow for imple access to rental requests and customer information.

5.2 The Team and the SE process

The Software Engineering process we used was the spiral method. In this method, we start in the middle of the model, and spiral outward, allowing all workers working on the Project to be an active part of every aspect of the engineering. Each individual person will be able to work and test during the concept development, system development, system enchancement and system maintenance phases of development.

5.3 Engagement of Umbrella Activities

Four of the main Umbrella activities we used were as follows:

1. Software Project Management - Which was used to lead the project and ensure that the project was controlled, monitored, and on schedule.
2. Formal technical Reviews - This activity was essentially implemented for peer review. Having new and fresh eyes to view code and ensure that everything met the requirements.

3. Reusability Management - This activity was used to help us create flexible and generic assets that may be reused for future projects or for this project in other regions. This would cut down on cost and help with consistency.

4. Risk Management - This activity was used to assess and identify potential risks with creating the software such as assuring that not too much money be spent in assets on the project.

6 User Manual

6.1 Software Description

Library Management System is a web base application for students, teachers and managers. The system will allow universities to manage their libraries easily with user friendly interfaces and system tools.

More specifically, the Library Management System will also allow students to search for books and request them. Teachers also can search books, request them and add publication to system. Managers can edit all information about users, books and respond to book requests.

6.2 How To Use Software

Register or log in after entering the library management system site. After logging in, you will see tabs for basic operations. Here you can review the books and request for them. Book requests will be evaluated by the manager. If you have a post you can share it. Your publications will also be reviewed by the administrator. If you have a book, a date will be set to return it. Sanctions are imposed on the owners of books that were not returned on this date.

Administrators can control and handle all the properties of the books. It can view all reports in the software. These reports include user actions and book actions.

6.3 Troubleshooting Common Problems

Invalid Login:

If the user name or password does not match the database, the website says that login is not possible.

Page Not Found:

Page Not Found error is received when trying to redirect to a page that is not on the site or without permission.

7 REVISED WITH NEW DESIGN PATTERNS

This project has been revised using abstract factory, factory and singleton design patterns.

Factory Design Pattern:

Factory pattern is one of the most used design patterns in Java. This type of design pattern comes under creational pattern as this pattern provides one of the best ways to create an object. This is done by creating objects by calling a factory method either specified in an interface and implemented by child classes, or implemented in a base class and optionally overridden by derived classes rather than by calling a constructor.

Abstract Factory Design Pattern:

Abstract Factory pattern is almost similar to Factory Pattern is considered as another layer of abstraction over factory pattern. Abstract Factory patterns work around a super-factory which creates other factories.

Singleton Design Pattern:

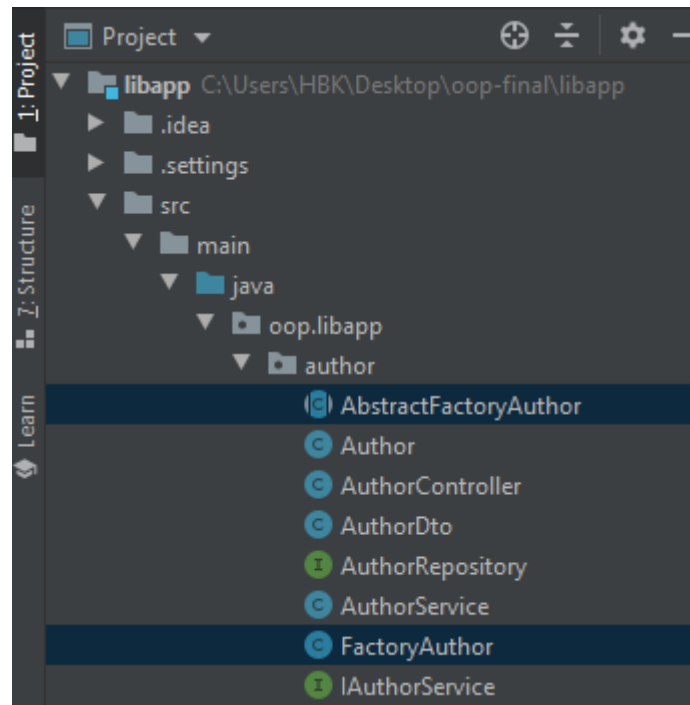
The purpose in this design pattern ensures that only one instance is created from a class. In other words, when an instance is wanted to be created from any class, if there is no previously created instance, it is newly created. If it was created before, the existing instance is used.

```
55  
56     Author author = new Author(authorDto.getName(), authorDto.getDescription());  
57
```

This path was followed to create objects. It took the following form with the new design pattern used.

```
// New Design Pattern Implementation  
Author author = factoryAuthor.getAuthor(authorDto.getName(), authorDto.getDescription());
```

AbstractFactoryAuthor and FactoryAuthor classes have been added in order to use new design patterns.



```
package oop.libapp.author;

public abstract class AbstractFactoryAuthor {
    abstract Author getAuthor();
    abstract Author getAuthor(String name, String description);
}
```

Abstract Factory patterns work around a super-factory which creates other factories. This factory is also called as factory of factories. AbstractFactoryAuthor is an example of Abstract Factory Design Pattern.

```
package oop.libapp.author;

public class FactoryAuthor extends AbstractFactoryAuthor{

    private static FactoryAuthor factoryAuthor = new FactoryAuthor();

    private FactoryAuthor(){};

    @Override
    Author getAuthor() { return new Author(); }

    @Override
    Author getAuthor(String name, String description) { return new Author(name, description); }

    public static FactoryAuthor getFactoryAuthor() {
        return factoryAuthor;
    }
}
```

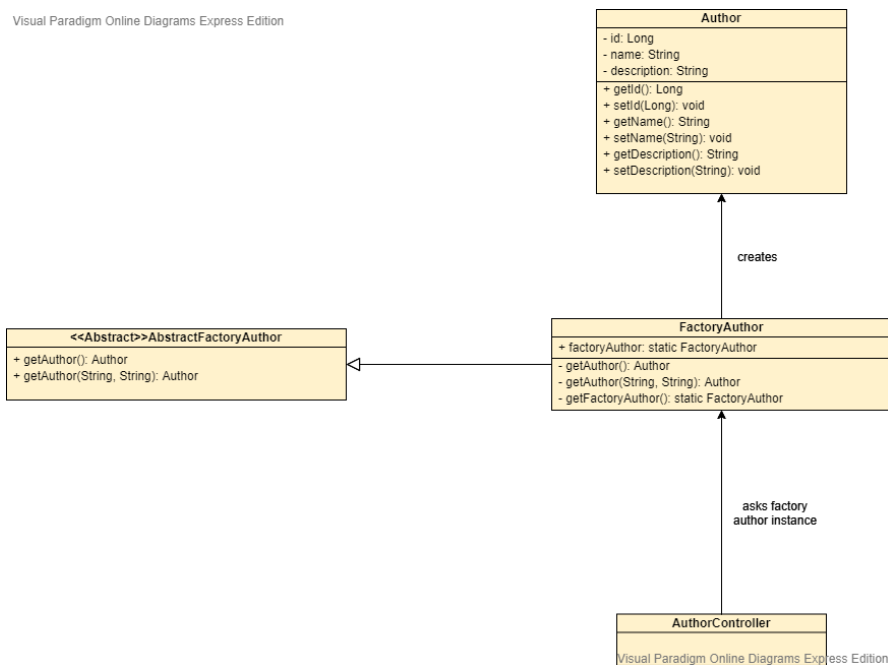
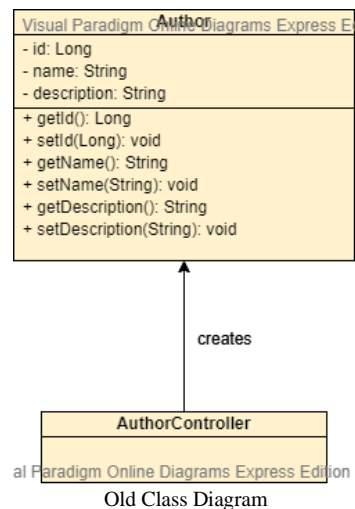
FactoryAuthor is consist of both Factory and Singleton Design Patterns. Singleton has achieved by using private constructor, having private instance variable and it's getter.


```

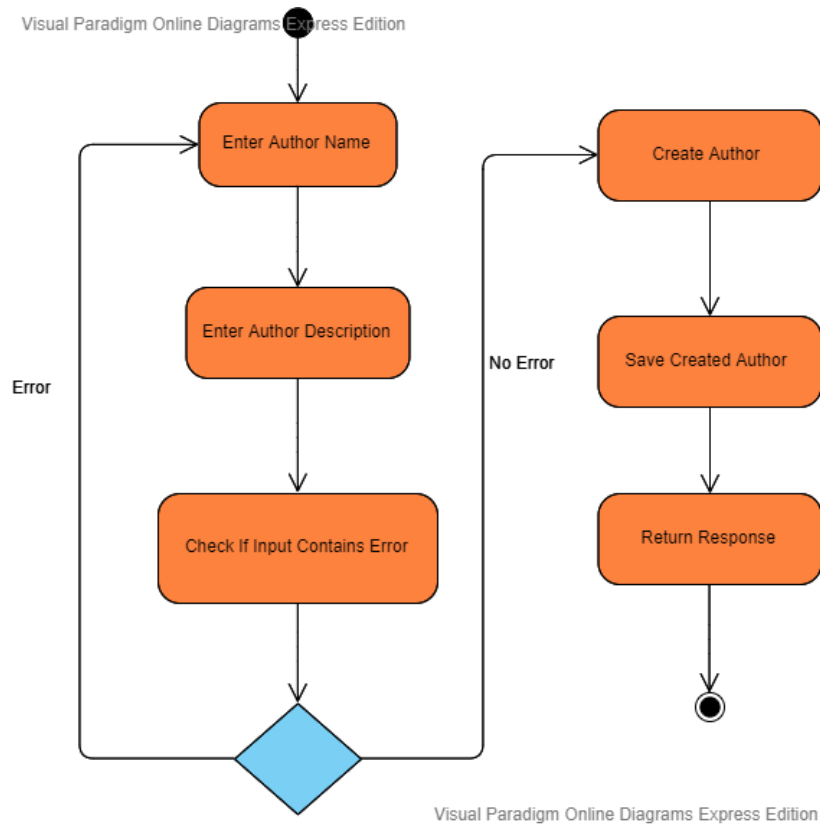
1 package oop.libapp.genre;
2
3 public class FactoryGenre extends AbstractFactoryGenre {
4
5     @Override
6     Genre getGenre() { return new Genre(); }
7
8
9
10    @Override
11    Genre getGenre(String name, String description) { return new Genre(name, description); }
12
13 }
14
15

```

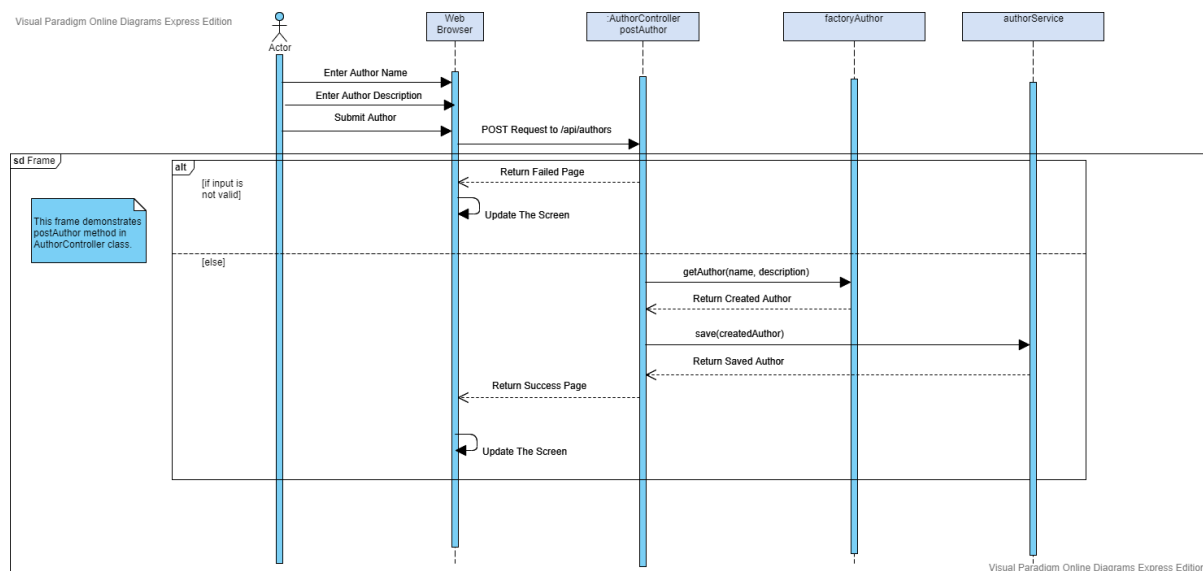
This is an example of using Factory Design Pattern without using Singleton in it.



These are class diagrams that shows AuthorControl to create a new Author. FactoryAuthor and AbstractFactoryAuthor classes have been added with the addition of new design patterns.



This is an activity Diagram of creating Author.



Sequence Diagram shows how to activity diagram example works on the application side in details.

Other packages have been changed as author package. Abstract Factory, Factory and Singleton Design Pattern used in packages which are author, book, entity and register. AbstractFactory and Factory Design Patterns used in Genre package. AbstractFactory.. and Factory.. classes have been added to packages

