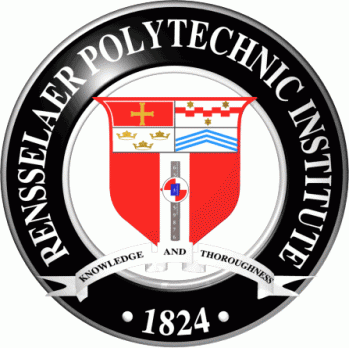
**SOFTWARE DEVELOPMENT 2014 SPRING**

**FINAL PROJECT REPORT**



INSTRUCTOR: PROFESSOR LIU

TEAM:

HUIYU MA

HANYANG GUAN

XUEYANG GUAN

Contents

[**1.** **Project Plan** 3](#_Toc387919898)

[**1.1** **The goal of the project** 3](#_Toc387919899)

[**1.2** **The stakeholders** 3](#_Toc387919900)

[**1.3** **The resource planning** 3](#_Toc387919901)

[**1.4** **The developmental phases** 3](#_Toc387919902)

[**1.5** **The software quality assurance** 4](#_Toc387919903)

[**1.6** **The project management methodology** 4](#_Toc387919904)

[**1.7** **The software management and development tools** 4](#_Toc387919905)

[**1.8** **How to control the change management** 4](#_Toc387919906)

[**1.9** **How to handle the configuration management** 4](#_Toc387919907)

[**1.10** **The progress milestones for delivering the artifacts and the product** 4](#_Toc387919908)

[**1.11** **The product maintenance plan after the delivery** 4](#_Toc387919909)

[**2.** **Functional Specification** 6](#_Toc387919910)

[**2.1** **Introduction** 6](#_Toc387919911)

[**2.1.1** **Summary** 6](#_Toc387919912)

[**2.1.2** **Requirements** 6](#_Toc387919913)

[**2.1.3** **Numbers** 6](#_Toc387919914)

[**2.1.4** **Existing System** 6](#_Toc387919915)

[**2.1.5** **Terminology** 6](#_Toc387919916)

[**2.1.6** **References** 6](#_Toc387919917)

[**2.2** **Functional Description** 6](#_Toc387919918)

[**2.2.1** **Use case Diagram** 6](#_Toc387919919)

[**2.2.2** **User Community** 7](#_Toc387919920)

[**2.2.3** **Administration Functions** 7](#_Toc387919921)

[**2.2.4** **Security** 7](#_Toc387919922)

[**2.2.5** **Help** 7](#_Toc387919923)

[**2.2.6** **Platforms** 7](#_Toc387919924)

[**2.2.7** **Customization** 7](#_Toc387919925)

[**2.2.8** **Configuration Management** 7](#_Toc387919926)

[**2.2.9** **Documentation** 8](#_Toc387919927)

[**3.** **Design** 9](#_Toc387919928)

[**3.1** **System Architecture** 9](#_Toc387919929)

[**3.1.1** **Introduction** 9](#_Toc387919930)

[**3.1.2** **Architecture** 9](#_Toc387919931)

[**3.2** **Domain Model** 9](#_Toc387919932)

[**3.2.1** **Introduction** 9](#_Toc387919933)

[**3.2.2** **Domain Diagram** 9](#_Toc387919934)

[**3.3** **Class Diagram** 10](#_Toc387919935)

[**3.3.1** **Introduction** 10](#_Toc387919936)

[**3.3.2** **Class Diagram** 10](#_Toc387919937)

[**3.4** **Sequential Diagram** 12](#_Toc387919938)

[**3.4.1** **Introduction** 12](#_Toc387919939)

[**3.4.2** **Sequential Diagram** 12](#_Toc387919940)

[**3.5** **Database E-R diagram** 14](#_Toc387919941)

[**4.** **Apply the software management tools** 16](#_Toc387919942)

[**4.1** **Introduction** 16](#_Toc387919943)

[**4.2** **Web application** 17](#_Toc387919944)

[**5.** **Unit testing and Integrated testing** 20](#_Toc387919945)

[**6.** **Configuration and Deployment** 22](#_Toc387919946)

[**6.1** **Configuration** 22](#_Toc387919947)

[**6.2** **Deployment with Ant** 25](#_Toc387919948)

[**Appendix: Use Case Documentation** 26](#_Toc387919949)

[**Reference:** 29](#_Toc387919950)

1. **Project Plan**
   1. **The goal of the project**

The purpose of the project is to apply all the methods and tools we in the software development course in this project, and it’s a chance for practice and have the management experience of management of the software development process of the RPI used book trading platform.

* 1. **The stakeholders**

The customers of this platform include RPI bookstore and student book buyers.

* 1. **The resource planning**

There are totally three members in our team, including Huiyu Ma, Xueyang Guan and Hanyang Guan. And the responsibilities for each member are as follow:

1. Huiyu Ma: Manager of the project and Software Developer

2. Xueyang Guan: Designer of the website and Software Engineer

3. Hanyang Guan: Quality Assurance, SQL constructor and Software Engineer

* 1. **The developmental phases**

Our developmental phases includes as below:

1. Initiation

First begins when a sponsor identifies a need or an opportunity. concept proposal is created

2. System Concept Development

Then defines the scope or boundary of the concepts, includes systems boundary document.

3. Planning

Then develops a project management plan and other planning documents. Provides the basis for acquiring the resources needed to achieve a solution.

4. Requirements analysis

Then analyses user needs and develops user requirements and create a detailed functional requirements documents.

5. Design

Then transforms detailed requirements into complete, detailed systems design document focuses on how to deliver the required functionality.

6. Development

Then converts a design into a complete information system includes acquiring and installing systems environment; creating and testing databases preparing test case procedures; preparing test files, coding, compiling, performing test readiness review and procurement activities.

7. Integration and test

Then demonstrates that developed system conforms to requirements as specified in the functional requirements document, conducted by quality assurance staff and users, produces test analysis reports

8. Implementation

Finally implementation includes implementation preparation, implementation of the system into a production environment, and resolution of problems identified in the integration and test phases.

* 1. **The software quality assurance**

The tools that been used for software quality assurance are version control - iceScrum, Redmine and Junit.

* 1. **The project management methodology**

The project management methodology we choose is Agile.

* 1. **The software management and development tools**

The software management and development tools had been used including Version Control (Tortoise), Project management tool (IceScrum) and Debug track tool (Redmine, Junit).

* 1. **How to control the change management**

The tools that been used to control the change management is version control (Tortoise).

* 1. **How to handle the configuration management**

The tool that been used to handle the configuration management is using company wiki to manage the configuration documents and issues.

* 1. **The progress milestones for delivering the artifacts and the product**

The progress milestones for delivering the artifacts and product are including:

1. Front end jsp website management

2. Back end SQL create and management

3. Test and debugger

* 1. **The product maintenance plan after the delivery**

There are six software maintenance processes as:

1. The implementation process including software preparation and transition activities.

2. The problem and modification analysis process that is executed once the application has become the responsibility of the maintenance group.

3. The process considering the implementation of the modification itself.

4. The process acceptance of the modification, after confirming the modified work with the individual who submitted the request in order to make sure the modification provided a solution.

5. The migration process is not part of daily maintenance tasks. If the software must be ported to another platform without any change in functionality, this process will be used and a maintenance project team is likely to be assigned to this task.

6. Finally, the last maintenance process is the retirement of a piece of software or the whole system.

1. **Functional Specification**
   1. **Introduction**
      1. **Summary**

This project is a web application aiming to provide RPI students with a platform for exchanging the used text books.

* + 1. **Requirements**

This project has two major groups of users: administrator and book buyers. For administrators, this project should be able to provide them with the ability to add books and relevant information (book title, book price, etc.) into the database. The book buyers should be able to register, log in to the system, browse exist book list, add book to cart, delete book from cart, view cart, and check out.

* + 1. **Numbers**

The expected user group is all the students at RPI. The expected user number is about 20,000 (each year about 5,000 new students enrolled in RPI). The expected peak usage time for this application is at the beginning of every semester.

* + 1. **Existing System**

The current RPI used book trading system is an actual book store located in the students’ union. Students must go to the bookstore themselves to look for a certain used book. The problem is that sometimes students need to spend a lot of time to find nothing (No storage, price too high, etc.). Our project solved this problem by moving the used books data from the bookstore to the internet, so that students could use the web application to find whether the books are available, and then they could order the books online.

* + 1. **Terminology**

RPI – Rensselaer Polytechnic Institute

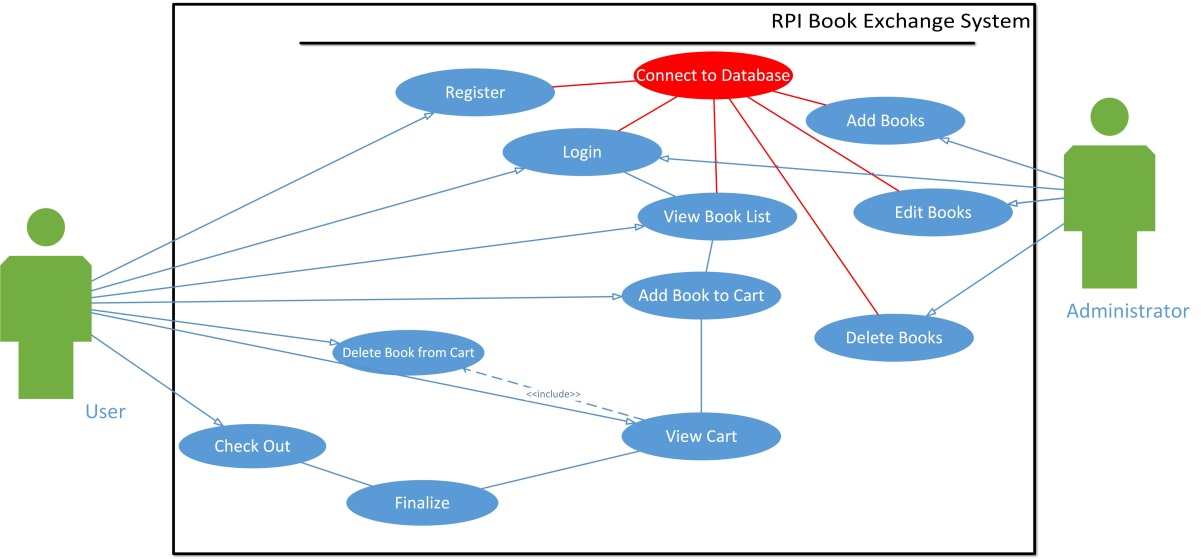
* + 1. **References**

[1]. <http://junit.org/>

[2]. <http://www.mysql.com/>

[3]. <http://tomcat.apache.org/>

* 1. **Functional Description**
     1. **Use case Diagram**



*See the Appendix for the documentation of this use case.*

* + 1. **User Community**

This project is aimed at all the graduate and undergraduate students in RPI.

* + 1. **Administration Functions**

The administrators would be assigned special user type, user name and password, and then they could log in as administrators. Administrators would be able to edit (add, modify or delete) the books from the book list.

* + 1. **Security**

In order to prevent the system abusing (for example, some people would buy all the books available online and then sell them in a higher price), we will set a limit in the number of the same books each person could purchase at a time (e.g. one account could only buy up to 3 same books) (Not in the demo). Besides, we have user terms and agreements before users could browse the book list. Only users who agree with our terms could use the system. Those who abuse or misuse this system will take legal responsibility.

* + 1. **Help**

Our system would provide help document and frequently asked question part to help users to use the application. (Not in the demo)

* + 1. **Platforms**

Our project was developed on Windows-based platform.

* + 1. **Customization**

The customers are not allowed to customize the system.

* + 1. **Configuration Management**

We are going to manage the project versions with version control applications. (We will talk about that in our demo)

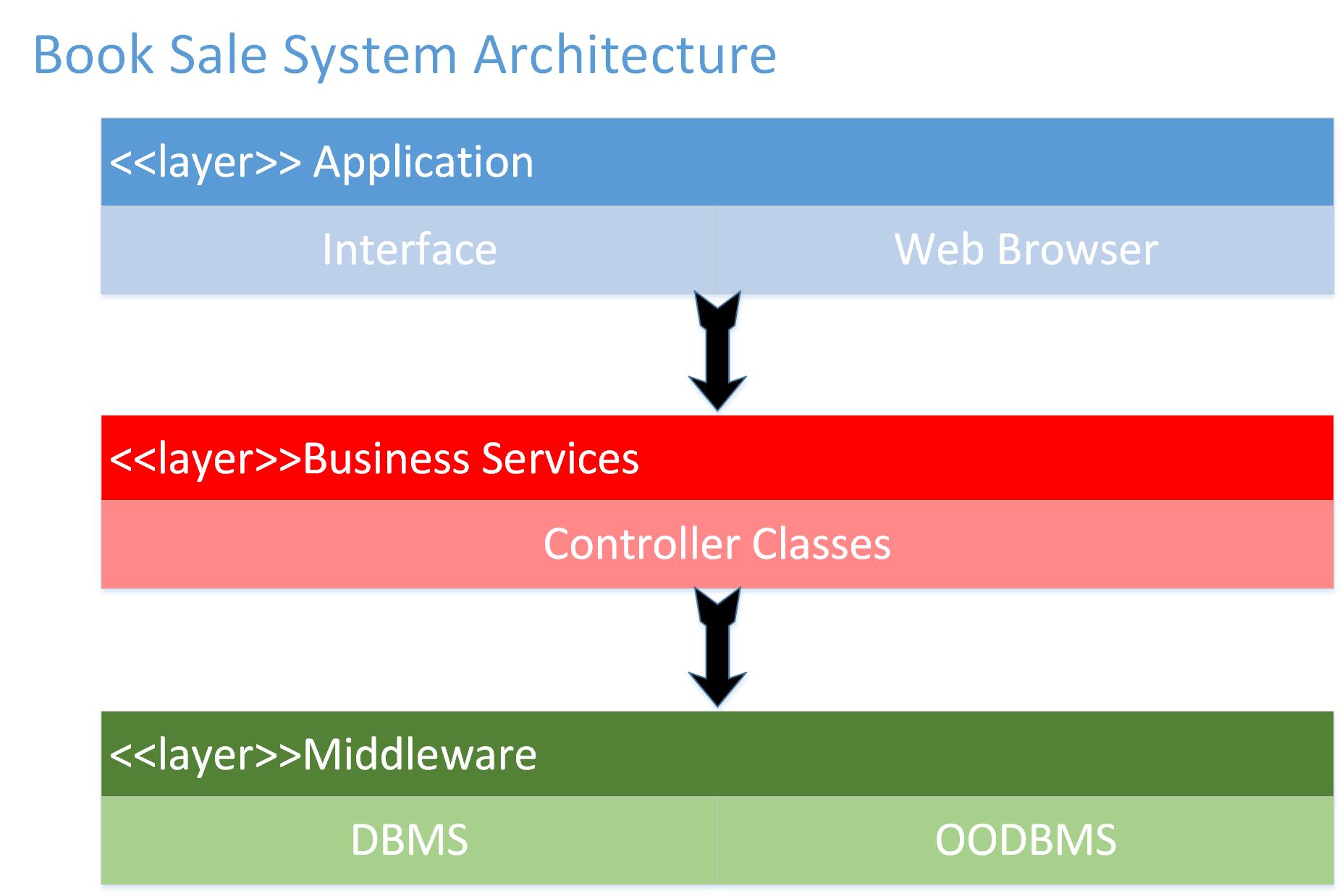
* + 1. **Documentation**

The documents that will be produced:

1. Design documentation (including system architecture, domain model, class diagram and sequential diagram)
2. Configuration and Deployment documentation
3. **Design**
   1. **System Architecture**
      1. **Introduction**

This software architecture part provides an architectural overview of the RPI book exchange system. The RPI book exchange system is being developed by Hanyang Guan, Xueyang Guan and Huiyu Ma as the final project of Software Development course.

* + 1. **Architecture**

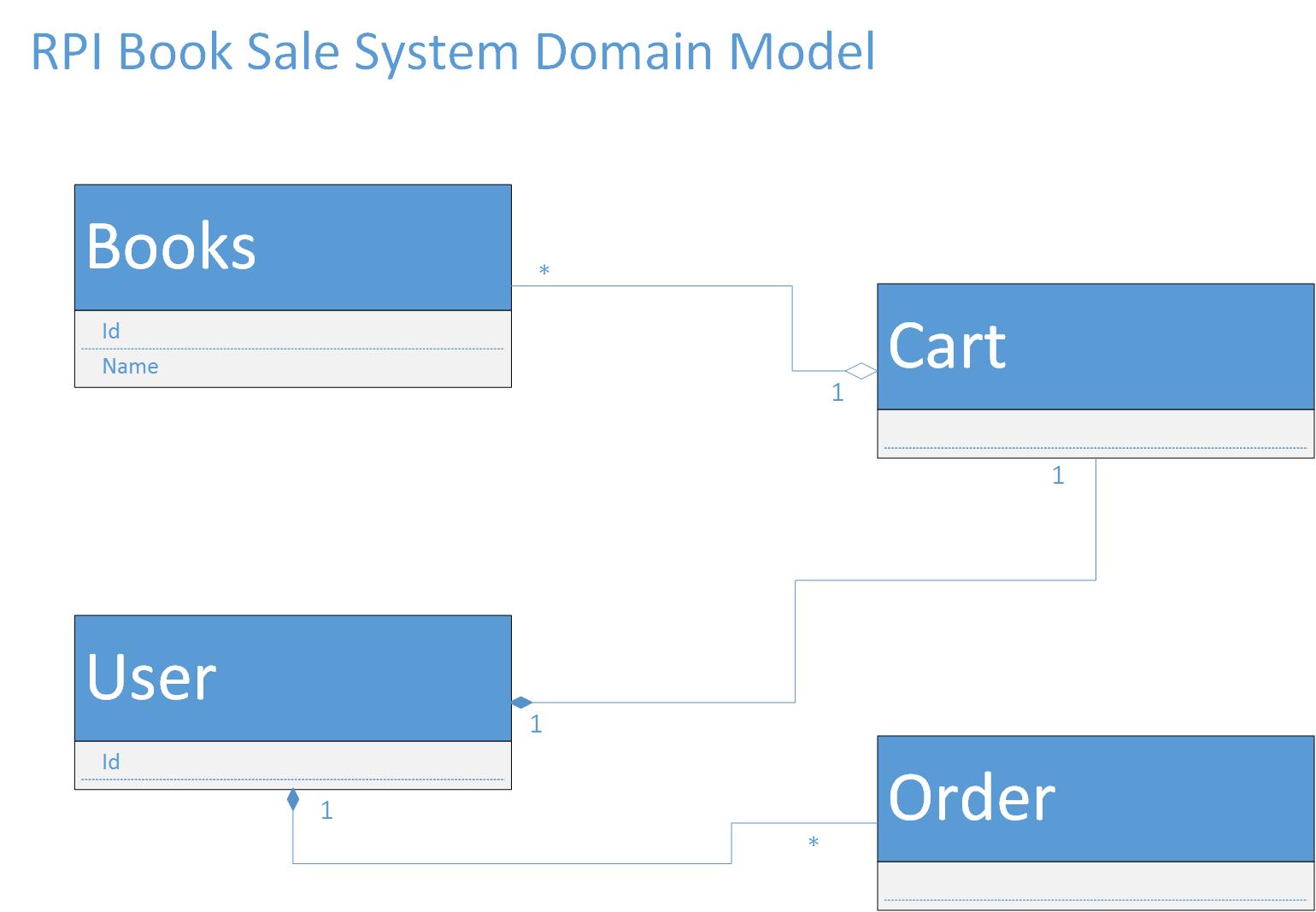


This graph represents the architecture of the RPI book sale system. This system has a 3-tier architecture. The first layer is the application layer, it has all the class on the application interface that users could see. The second layer is the Business Services layer, it has all the controller classes which control the application layer. The third layer is the Middleware layer, which supports the access to the database management systems.

* 1. **Domain Model**
     1. **Introduction**

This part provides the domain model of the RPI book sale system. This model represents the key concepts of the problem domain and identifies the relationships among the entities, which in this case are users, books, shopping cart and orders.

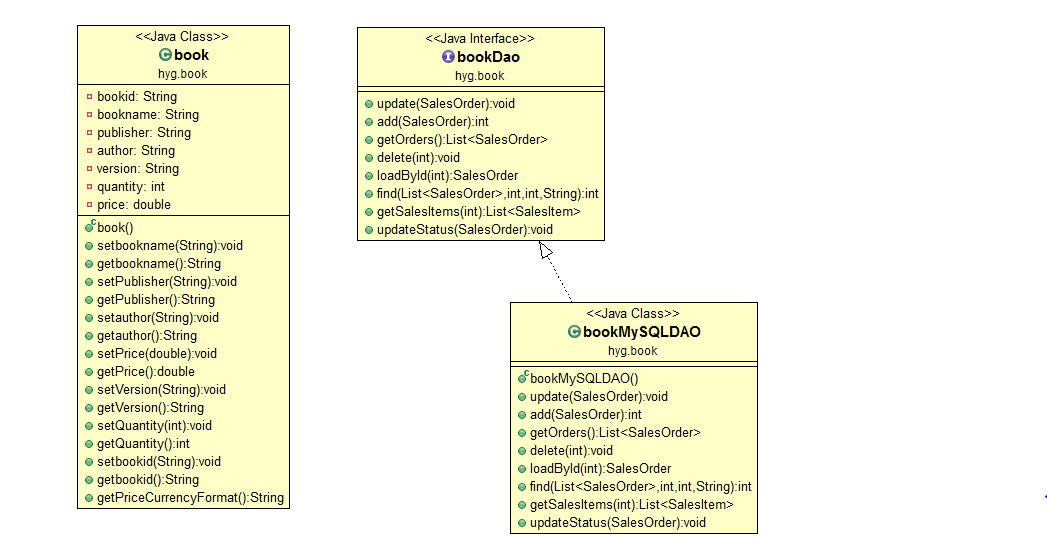
* + 1. **Domain Diagram**

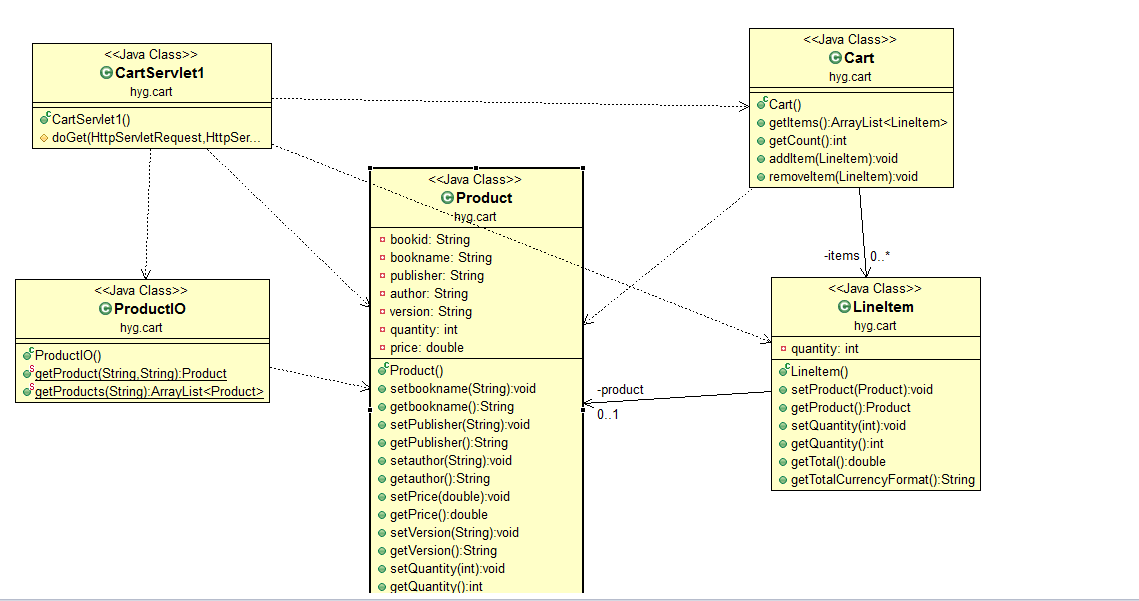


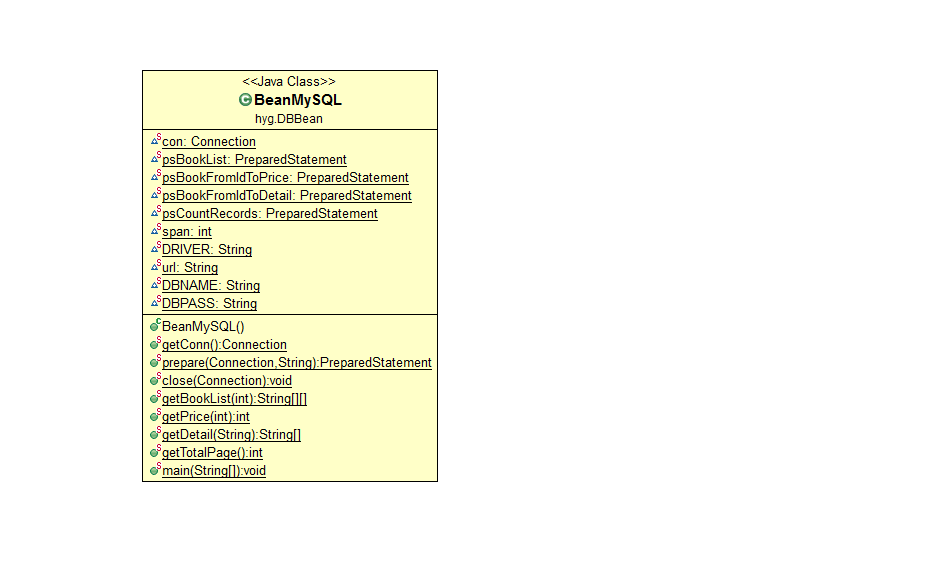
* 1. **Class Diagram**
     1. **Introduction**

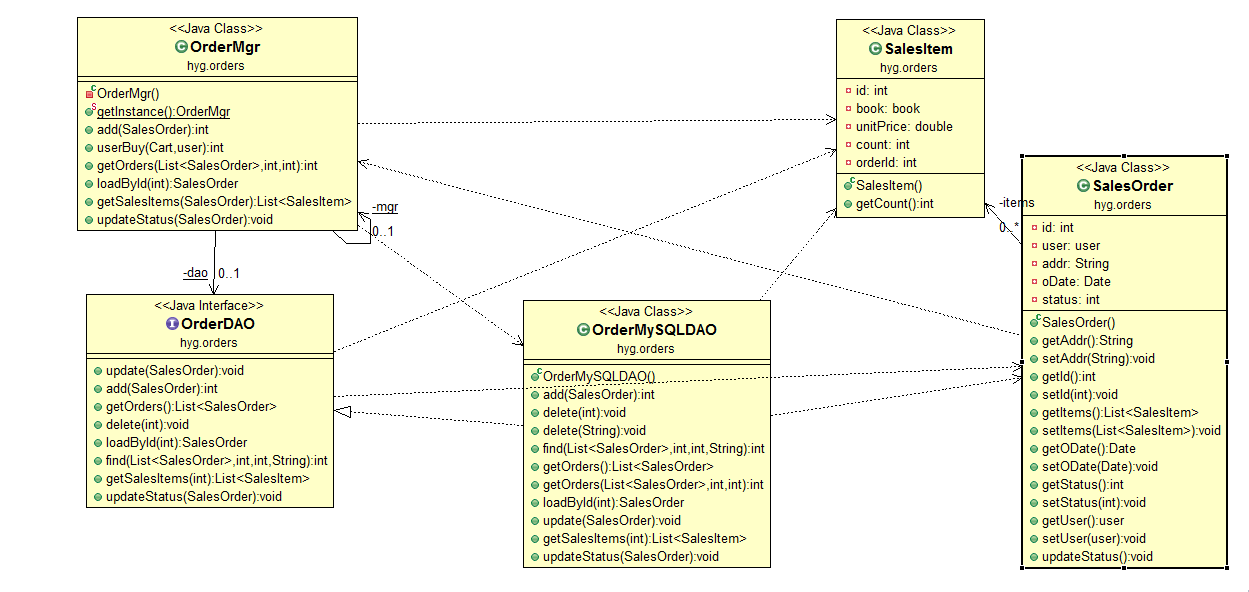
This part provides the UML class diagram of the RPI book sale system. These diagrams are generated directly from Eclipse.

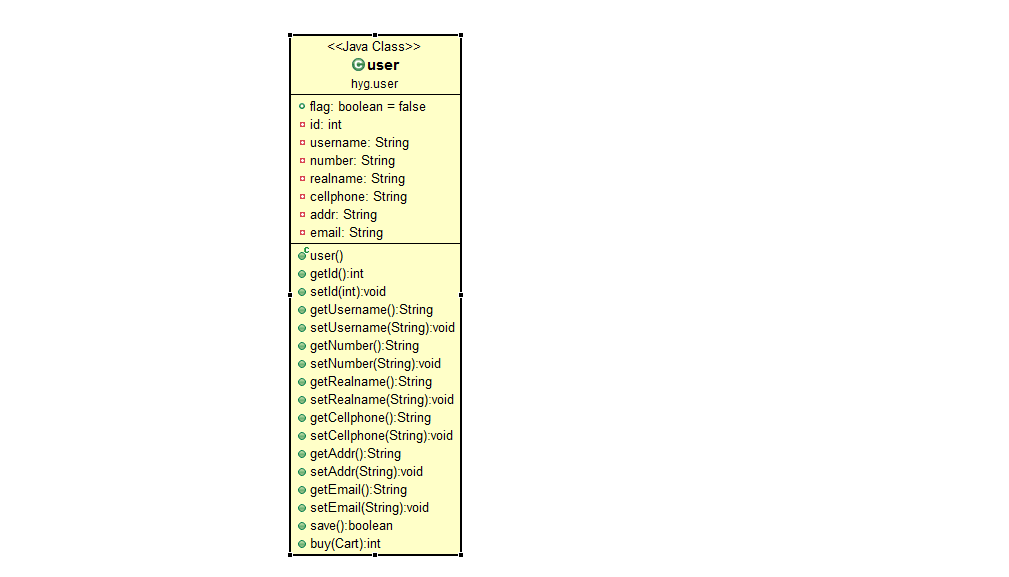
* + 1. **Class Diagram**







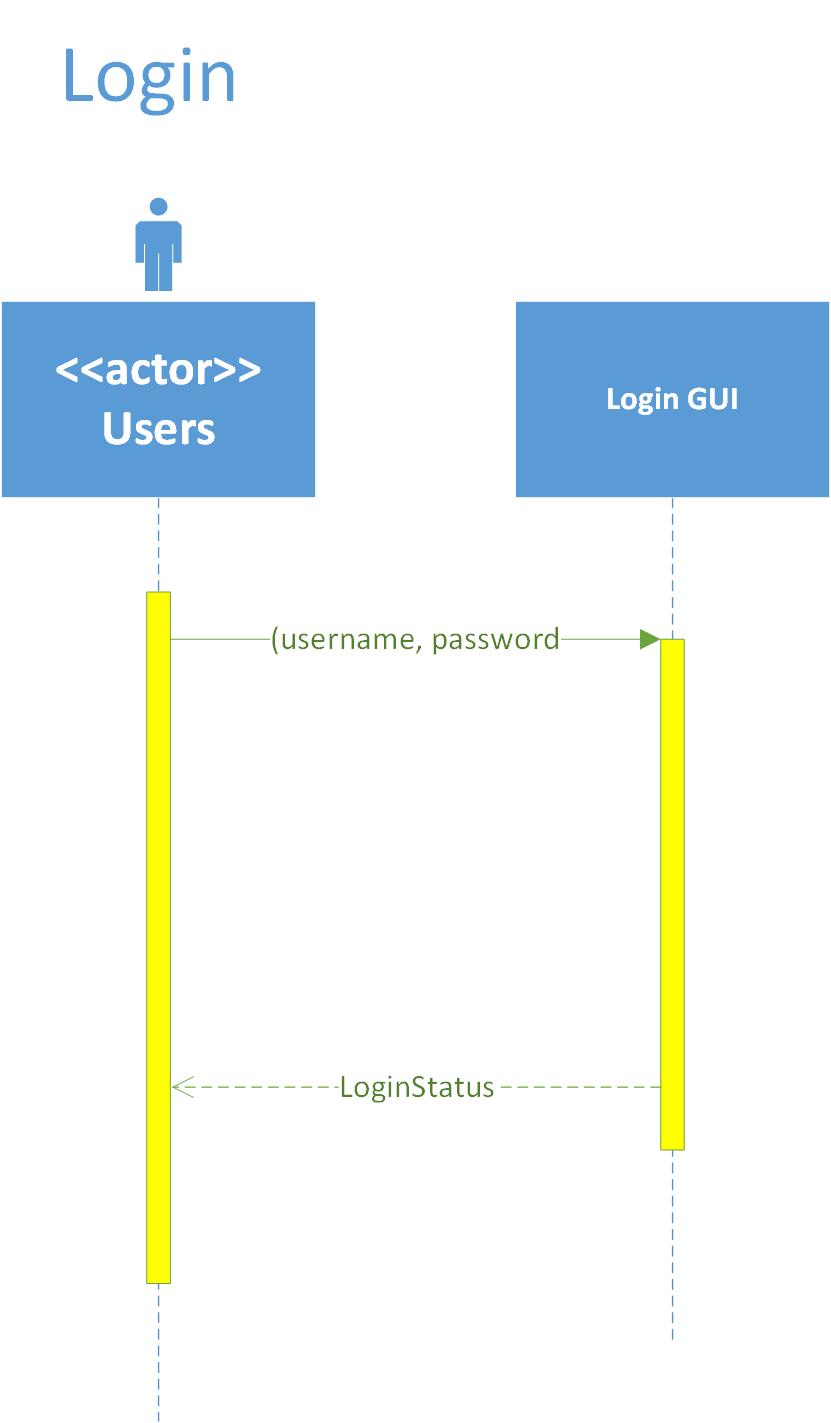




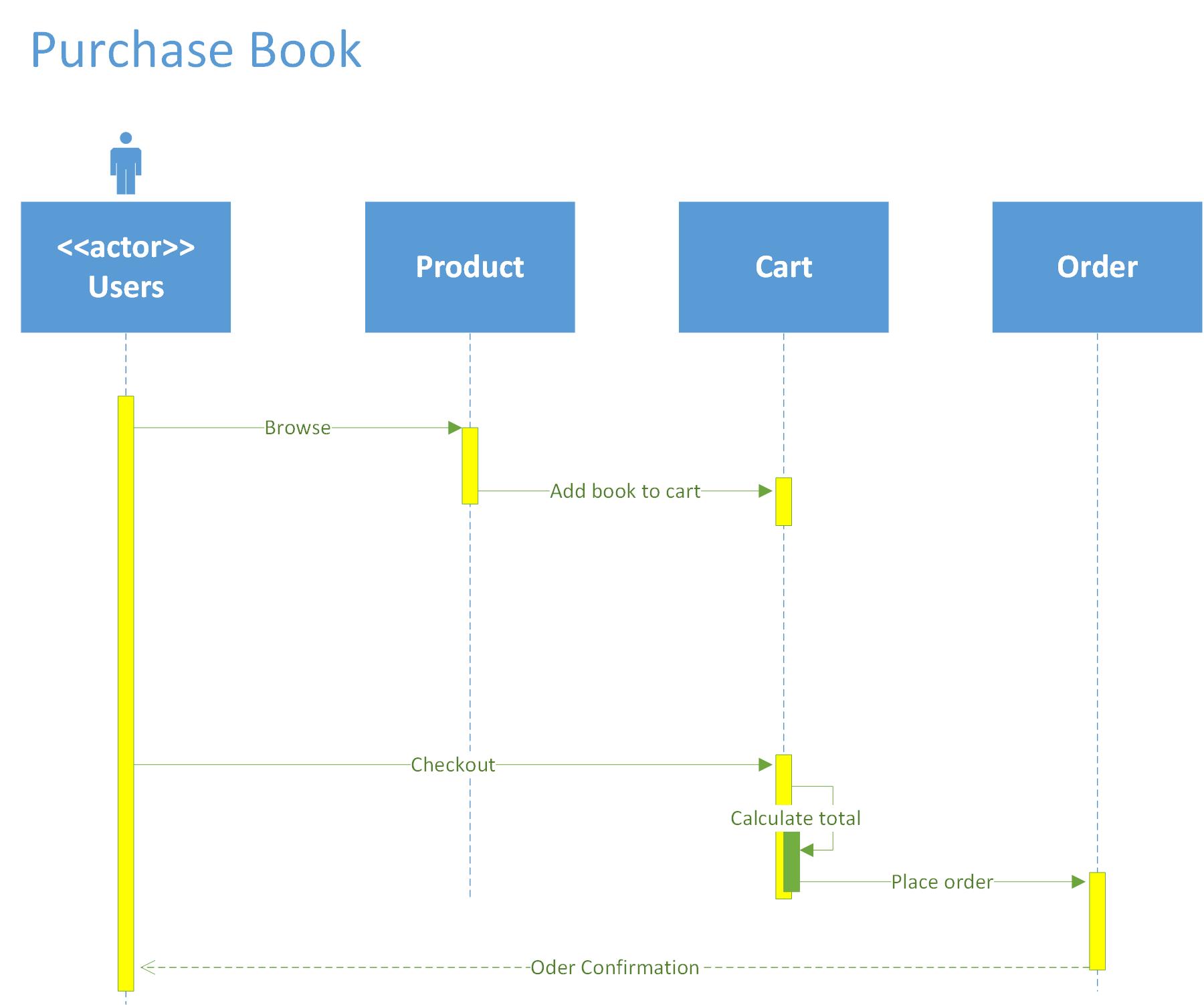
* 1. **Sequential Diagram**
     1. **Introduction**

This part provides the sequential diagram of the main activities of the RPI book sale system. The diagrams show the orders of processes operating with one another. The sequential diagrams of this book sale system include two parts: the login sequence and the purchase sequence.

* + 1. **Sequential Diagram**



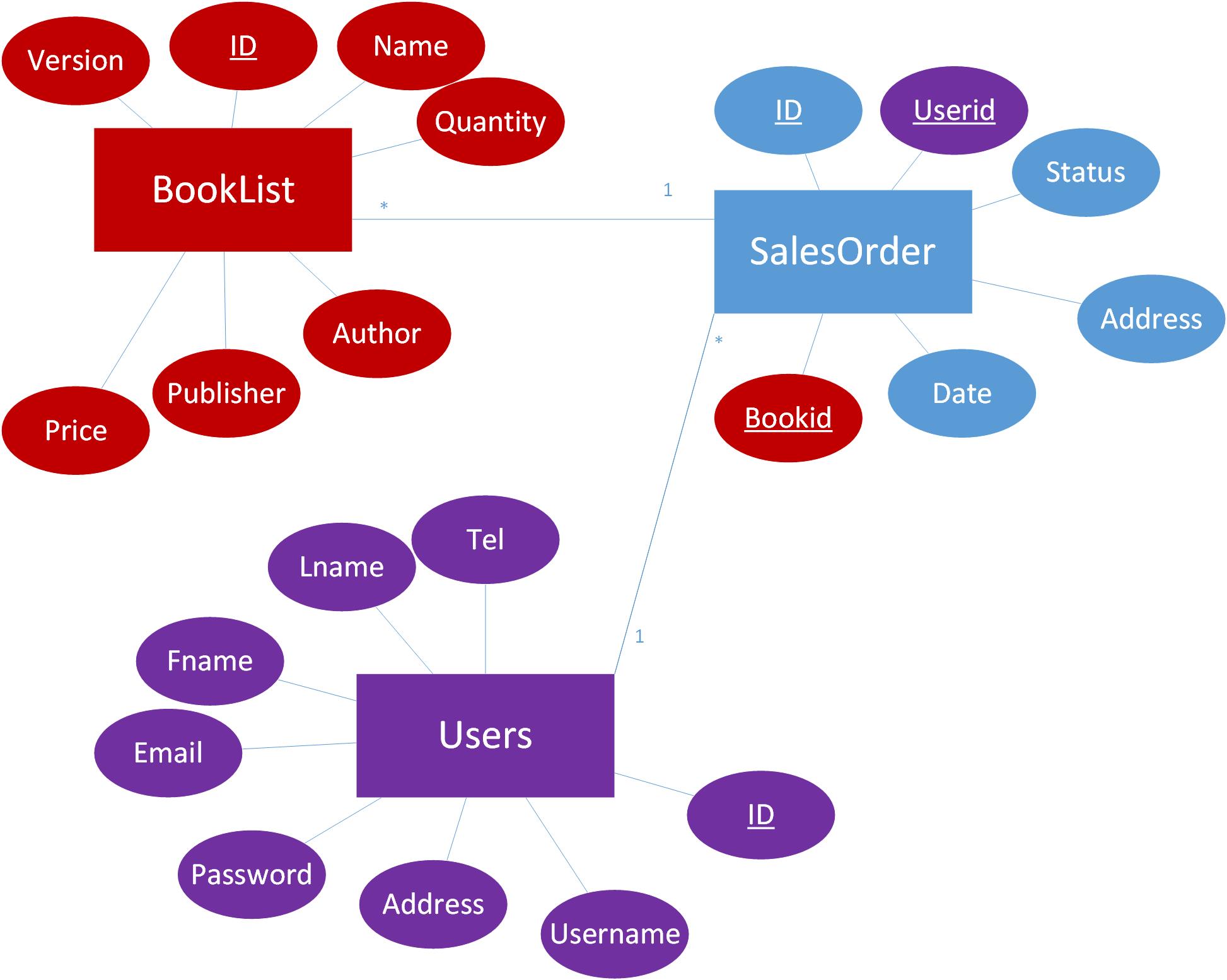
The login sequence diagram is shown as above, the users send message including username and password to the login GUI, and the GUI will return the login status.



The purchase book sequence diagram is shown as above, the users browse products(books), and then add book to cart When the user send message of checkout to the cart, the cart calculate the total price of all the items in cart and send place order message, then the user will get the order confirmation.

* 1. **Database E-R diagram**

The following is our database design, we have three entities: Books, Salesorders and Users. One user has one or more sales orders; one order can have one or more books; they connect to each other using two foreign keys: UserID and BookID.



1. **Apply the software management tools**
   1. **Introduction**

Our team is using the iceScrum as our major software management tool, and our project started from March 10 and ended at May 5, there are totally 4 Sprints in our first round of software development.

There are totally 5 major features included in our system:

1. Database management

This feature includes establishment and management of Mysql. And this functionality is completed during the sprint 1.

2. User information management

In this feature, there are 3 different functionalities should be achieved, the user register, user login in/out and user validation, all these 3 functionalities are finished in the sprint 2.

3. Books management

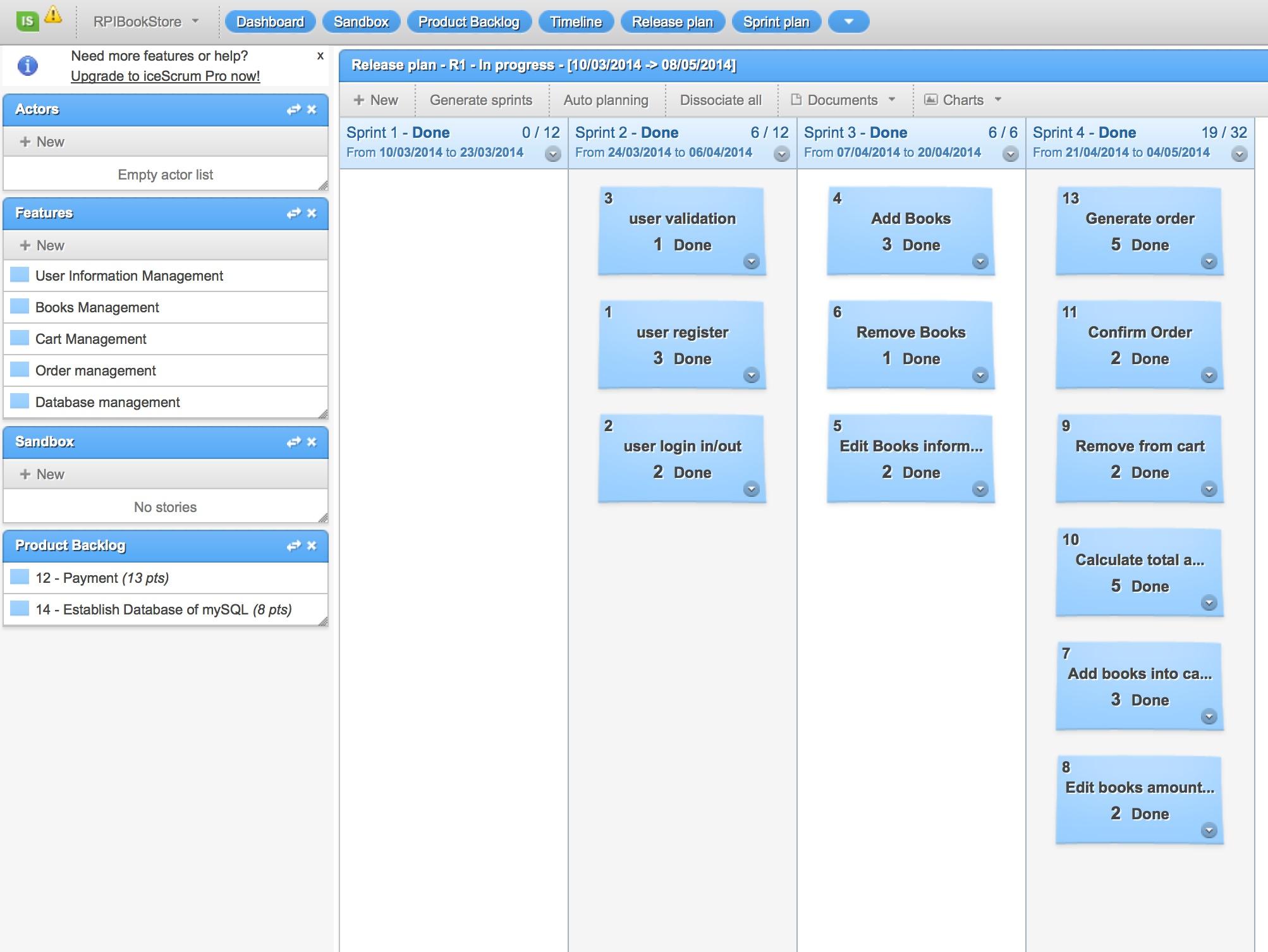
There are 3 different functionalities included in this feature: Adding books, editing books information and removing books. All these functionalities are completed in the sprint 3.

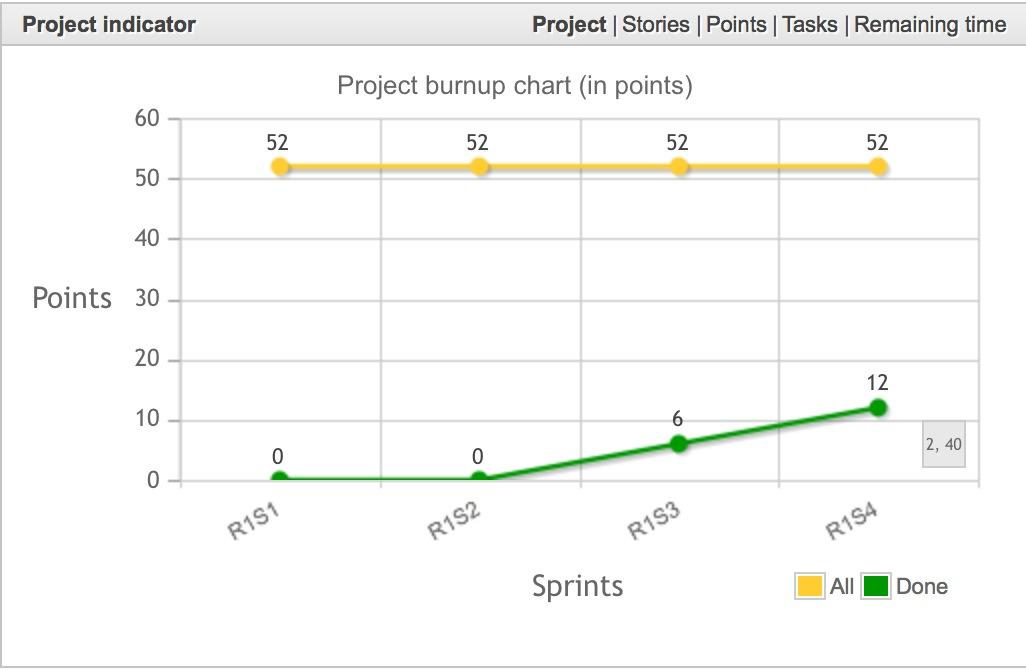
4. Cart management

There are 4 major functionalities concluded in this feature, they are adding books to cart, editing Amount, removing from cart and calculation of total amount and value; All these functionalities are completed during the sprint 4.

5. Order management

Order management including totally 3 major functionalities: confirmation of order, payment method and generation of order. Except payment method, all the other functionalities are finished during the sprint 4.





* 1. **Web application**

a. Web pages for users to login/logout/registration, and login validation

For user registration, the web application require customer to enter the value of username, password, telephone, real name, address and email address to finish the registration. And during typing information into the web application, the system will verify whether each information match the format required. After all the information check finished, the web application will link to the Mysql database and write down all the user register information.

For user login web page, the web application will link to the Mysql database and search all the user information recorded in the user table, and if the web application found the record that match the request, the user will pass the login validation and login successfully, at the same time, the website will also record the username and use rid in the session, so that it could be used through the whole website.

For user logout, when the user choose to logout the system, the website will automatically clear the information in the session and then the user will logout the system. And next time when the user would like to use the system, the user is required to login again.

b. Web pages with functions of record creation/update/delete/search (This portion will be specific to your project.)

In our web application, we have link it with database to achieve the creation/update/delete/search function of books, user, order information and so on.

c. Database access is preferred over using a file system

We had chosen the Mysql as our database management software over using a file system.

d. Help link for some of your web pages

There is not help link currently in our project, but for further work, we plan to add help link in the future.

e. Web page navigation

When each website had finish it function, it will automatically jump into the next navigation website or jump into a page reporting error. For example, when the user had finish the registration process, the website will automatically jump into the login page, so that the users can directly use the information they just register to login.

Besides this, there are several different navigation hyperlinks in nearly each page, so that you can directly link to the page you would like to visit. For example, when the users are browsing the list of books, they could choose the book they prefer and directly jump into this page and then finish further operation.

f. Validation of the input data

When the user are typing the information into the website, the web application will automatically have a check of format and content of each information, so that all the information match the required format, and can be used correctly.

g. You may need to create a war file for the deployment

We had compressed the war file for you for the deployment.

h. You may need to write xml files for the deployment

We had finished the xml in ant for deployment.

i. Deploy the application to the web server on your local host

We have already used the ant to finish part of deployment, more details please see the ant deployment part.

1. **Unit testing and Integrated testing**

In this project, our team is using JUnit for unit testing and then integrated testing.  A unit test targets a small unit of code, such as a method or a class. An integration test has the target to test the behavior of a component or the integration between a set of components.

For example, in our book.java class, part of the member functions is as follow:

*package hyg.book;*

*import java.io.Serializable;*

*import java.text.NumberFormat;*

*public class book implements Serializable*

*{*

*private String bookid;*

*private String bookname;*

*private String publisher;*

*private String author;*

*private String version;*

*private int quantity;*

*private double price;*

*public book()*

*{*

*bookname = "";*

*publisher = "";*

*author="";*

*version="";*

*quantity=0;*

*price = 0;*

*}*

*public void setbookname(String bookname)*

*{*

*this.bookname = bookname;*

*}*

*public String getbookname()*

*{*

*return bookname;*

*}*

*}*

And then we create a JUnit test case to test member function of setbookname:

*package hyg.book;*

*import static org.junit.Assert.\*;*

*import org.junit.Test;*

*public class bookTest {*

*@Test*

*public void testSetbookname() {*

*// book is tested*

*book tester = new book();*

*tester.Setbookname("Software Development");*

*// check if Setbookname("Software Development") reset the value of bookname*

*assertEquals("10 x 5 must be 50", 50, tester.multiply(10, 5));*

*assertEquals("Setbookname("Software Development") change the book name",*

*"Software Development", tester.Getbookname());*

*}*

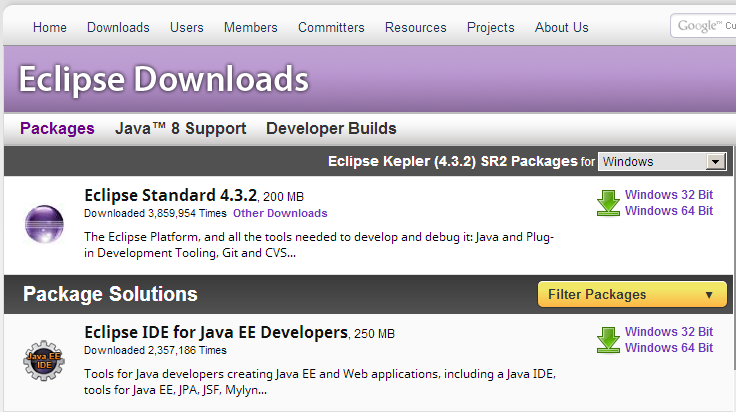
*}*

1. **Configuration and Deployment**
   1. **Configuration**

1. Download eclipse

First, download Eclipse IDE for Java EE Developer from http://www.eclipse.org/downloads/.

This version IDE is integrated with Tomcat server which our web project will use.



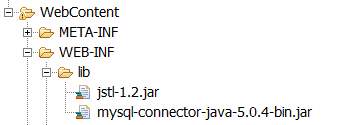
1. Download MySQL java connector

MySQL java connector is a plug-in which we use to connect IDE and database. Download MySQL connector jar file from http://dev.mysql.com/downloads/connector/j/. And unzip it and find the bin file. The version we use is ‘mysql-connector-java-5.0.4-bin.jar’. The jar file looks like this.

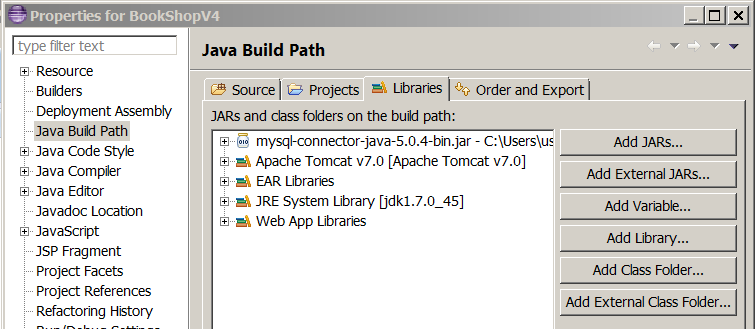


1. Build path for MySQL java connector

First, copy and paste the jar file the folder of lib which is under the directory of WEB-INF which is the sub-directory of WebContent as the diagram shows.



Then build path for connector, right click on the project, click build path->configure build path. Then show the tool box as following. Click libraries and then click ‘Add External JARs’ from the right bars. Go to the directory where your jar file is and select it. Click ‘open’, then return to the tool box, then click ‘OK’. You will finish build path.



Finally, the connector is in the same level of libraries as shows; make sure you configure it right.

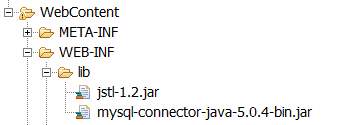


1. Download JSTL

JSTL is short for JSP standard tag library. We used it in the JSP files. Download it at the following url: <https://jstl.java.net/download.html>.

1. Put JSTL under

Copy and paste the jar file the folder of lib which is under the directory of WEB-INF which is the sub-directory of WebContent as the diagram shows.



1. XML Deployment

In the shopping cart part, we use the servlet rather than JSP, the following is the XML document deployment

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<web-app xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns=*"http://java.sun.com/xml/ns/javaee"* xsi:schemaLocation=*"http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd"* id=*"WebApp\_ID"* version=*"3.0"*>

<display-name>BookShopV4</display-name>

<welcome-file-list>

<welcome-file>index.html</welcome-file>

<welcome-file>index.htm</welcome-file>

<welcome-file>index.jsp</welcome-file>

<welcome-file>default.html</welcome-file>

<welcome-file>default.htm</welcome-file>

<welcome-file>default.jsp</welcome-file>

</welcome-file-list>

<servlet>

<servlet-name>CartServlet1</servlet-name>

<servlet-class>hyg.cart.CartServlet1</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>CartServlet1</servlet-name>

<url-pattern>/cart</url-pattern>

</servlet-mapping>

<session-config>

<session-timeout>30</session-timeout>

</session-config>

<welcome-file-list>

<welcome-file>

index.jsp

</welcome-file>

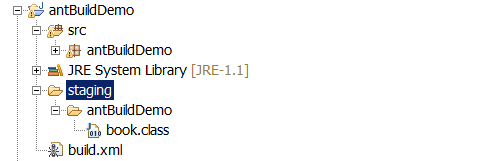
</welcome-file-list>

</web-app>

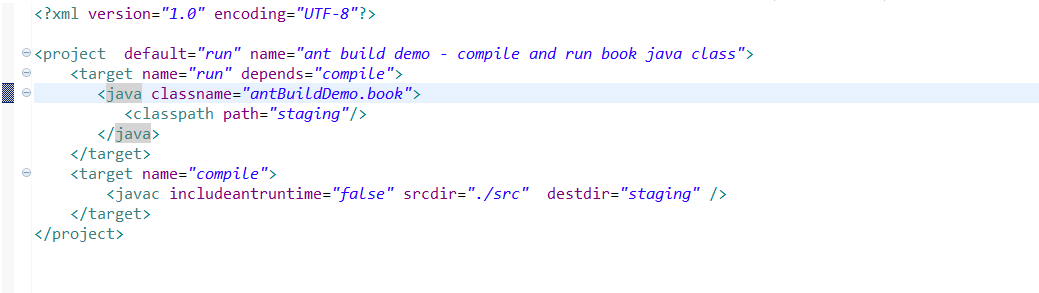
* 1. **Deployment with Ant**

This part is created to show how we deploy our project with Apache Ant.

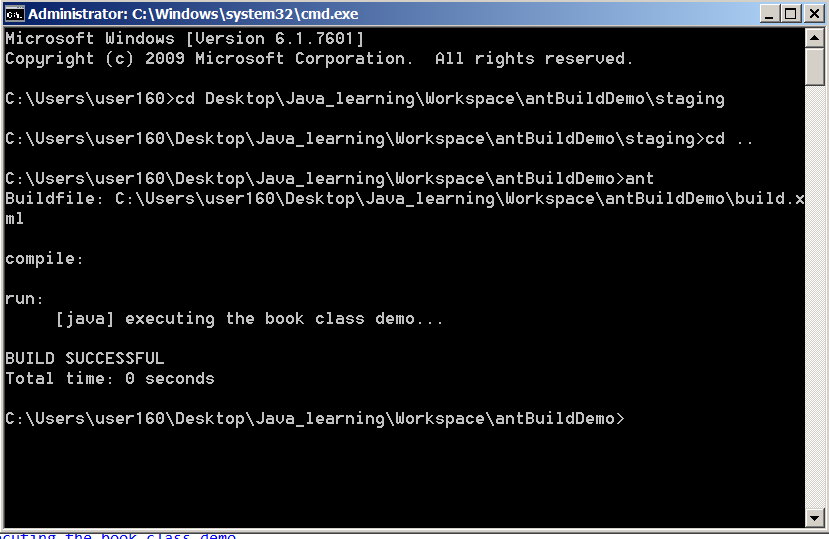
In this first stage, we create a folder named antBuidDemo to storage one of our class: book.java. And then we go through the process to establish the ant and write the build.xml file for deployment.



As below shows, our example build.xml has the function of compile and run.



Then we use the cmd run the ant, the result shows that we have successfully finish the ant function.



# **Appendix: Use Case Documentation**

|  |
| --- |
| Use Case Name: RPI Used Book Exchange System |
| Point of Contact Name: Hanyang Guan, Xueyang Guan, Huiyu Ma |

|  |
| --- |
| **Use Case Name**  RPI used book exchange system. |
| **Goal**  The goal of this use case is to provide the graduate and undergraduate students a platform to buy used text books. |
| **Summary**  The users of this use case would be able to register, log-in, view books, add book to cart, delete book from cart and checkout. The administrators of this use case would be able to add, delete and modify the book list. |
| **Actors**  Primary: Users  Secondary: Administrators |
| **Preconditions**  When students need a text book, they will buy a used one.  Students prefer buying books online to going to bookstore. |
| **Triggers**  Any time a student needs a text book. |
| **Basic Flow**   1. Administrator adds book into book list. 2. User registers. 3. User logs in. 4. User browses the book list. 5. User adds selected books into cart. 6. User views shopping cart. 7. User checks out. |
| ***Alternate Flow***  If user don’t need a book in the cart   1. User views shopping cart. 2. User deletes a book from cart. 3. User views shopping cart. 4. User checks out. |
| **Post Conditions**  The new book lists are correctly updated |
| ***Activity Diagram***  C:\Users\apple\Desktop\sd pic\use case3.jpg  **Register:** This use case allows the user to register a new user account by inputting information including user name, password, real name, e-mail address, etc. This use case will create a new tuple in the database.  **Login:** This use case describes how guests log into the system. The guests include users and administrators. This use case will connect the database to verify the user name and password.  **Add Books:** This use case is ONLY for administrators to add a new book into the database.  **Edit Books:** This use case is ONLY for administrators to edit the existing book information in the database.  **Delete Book:** This use case is ONLY for administrators to delete an existing book in the database.  **View Book List:** This use case allows the user to browse all the books existing in the database.  **Add Book to Cart:** This use case allows the user to choose one book form the book list and add to shopping cart.  **View Cart:** This use case allows the user to browse the current items in the shopping cart. This use case includes the use case Delete Book from Cart.  **Delete Book from Cart:** This use case allow the user to delete one item in the shopping cart.  **Finalize:** This use case allow the user to check again for missing or unwanted items in the shopping cart.  **Checkout:** This use case allow the user to pay for the existing items in the shopping cart. |
| **Notes**  RPI –Rensselaer Polytechnic Institute |

# **Reference:**

1. Ralph, P., and Wand, Y. A Proposal for a Formal Definition of the Design Concept. In, Lyytinen, K., Loucopoulos, P., Mylopoulos, J., and Robinson, W., (eds.), Design Requirements Engineering: A Ten-Year Perspective: Springer-Verlag, 2009, pp. 103-136

2. McConnell, Steve. "7: Lifecycle Planning". Rapid Development. Redmond, Washington: Microsoft Press. p. 140.

3. "ISO/IEC 14764:2006 Software Engineering — Software Life Cycle Processes — Maintenance". Iso.org. 2011-12-17. Retrieved 2013-12-02.

4. Lientz B., Swanson E., 1980: Software Maintenance Management. Addison Wesley, Reading, MA

5. <http://en.wikipedia.org/wiki/Software_maintenance>