HACETTEPE UNIVERSITY DEPARTMENT OF COMPUTER ENGINEERING

BBM 487 SOFTWARE ENGINEERING LAB

ARCHITECTURE NOTEBOOK

Özdeş Öztürk	Ekin Kandemir	Kadir Bulut
21228635	**21228434**	**21228137**

<Library Book Loan System>
Architecture Notebook

1. Purpose

This document describes the philosophy, decisions, constraints, justifications, significant elements, and any other overarching aspects of the system that shape the design and implementation.

2. Architectural goals and philosophy

This section describes the software requirements and objectives that have some significant impact on the architecture. The architecture of the system was shaped in core subjects. This system will be desktop application. We need only database connection each other. Our application performance is changing and according to computer or connection between computer, it's response time will be low or high.

Technical platform : Library Book Loan System will be desktop application.

Transaction : The Library System application is transactional, leveraging the technical platform capabilities. Transaction management model of the PHP and Oracle platforms will be re-used intensively.

Security: The application must implement basic security behaviors:

- **Authentication:** Login using at least a user name and a password.
- **Authorization:** According to their profile, library authority must be granted or not to perform some specific actions.
- **Confidentiality:** Sensitive data must be encrypted.
- **Auditing:** Every sensitive action can be logged.

Persistence: Data persistence will be addressed using a relational database.

Reliability / Availability : High availability is required since there are monetary issues related to the systems availability. The availability of the system is a key requirement by nature, as it is a library system. The candidate architecture. Targeted availability is 16/7: 16 hours a day, 7 days a week. The time left (8 hours) is reserved for any maintenance activities.

Usability / Reliability : As the developers of this project, undoubtedly, usability is the most crucial point for this system since the main aim of constructing this software project is to provide a smooth and practical system to the users. The most crucial requirement for every single user in the system is a smooth and practical interface that will direct the user without leaving any question marks on their mind. The user will be directed to the operations that they are eligible to do, so that they will make the best use of such a system. In brief, the basic requirement for usability is building a user friendly system that is ready-to-use any time.

Reliability is also a crucial point considering the system will contain the data of numerous members and we have to maintain the data integrity. To hold the data in a proper way, as mentioned in previous documents, we will be using a database for the purpose of data integrity. To provide the requirements and maintain reliability, we will get help from database logging which is an important part for our database solution design. With database logging, we will make it possible to recover from a failure, therefore we will maintain the data integrity.

Performance: Search queries should return %70 of the time below 1 sec.

Internationalization : Initially the system will support English. But it should be easily modifiable to extend language support.

3. Assumptions and dependencies

Library Book Loan System application was developed availably for any changes about use-case, user interfaces etc. Our application devised as a desktop computer. They won't need another connection or arguments to do this.

New features can be added in future versions of the system. This is why the system has been designed accordingly. That software has been designed to develop a way. Because the system desktop application only works in Windows operating systems. In terms of project developers it consists of over Oracle database. This information is made by considering improvements.

4. Architecturally significant requirements

- System OS must be Windows.
- Library Book Loan System must be downloaded.
- Users must have desktop computer and its input devices (mouse, keyboard, monitor…).
- All computers, had to be installed before the system using and have to be connected each other.
- System is a desktop application. Because of that this program must be open.
- Users who want to login the system have to be filled forms in application completely.
- If any users want to reservation of books, they must be login with ID and password.
- If Librarians want to manipulate books or users, they must be login with ID and password.
- Librarian have to choose type of books or users when new books or users will add as a new at system.
- If any connection lost or any problem about this system, Library system will recover. System is recording them everyone hours.
- The system will run seven days a week, twenty-four hours per day.

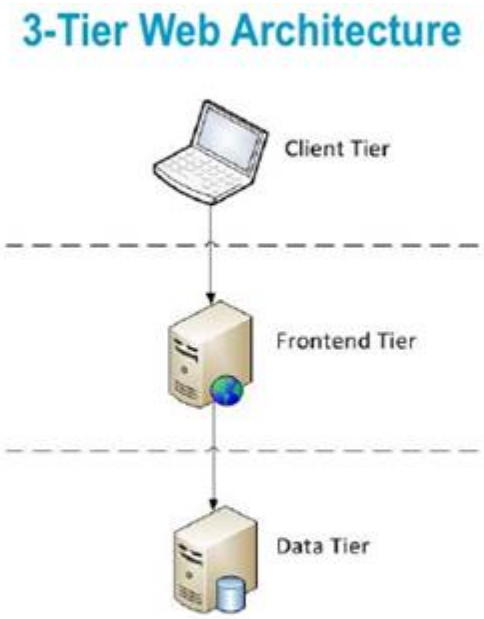
5. Decisions, constraints, and justifications

- In program architecture, understandable user interfaces is designed.
- Functionality is more important than visuality.
- All buttons is used to easier to use application.
- The error messages should be understandable for users.
- The application should be user friendly.
- The product is desktop application because of that Internet is not necessary.
- Librarian has to fill name, surname, birthday, priority of user to do adding user process.
- User has to fill username, surname, birthday…
- Users can be added by only admin.
- The visitors (not member in this system) can do only book searching.
- Book add, delete, update can be done by admin.
- Book reservations is done after login.

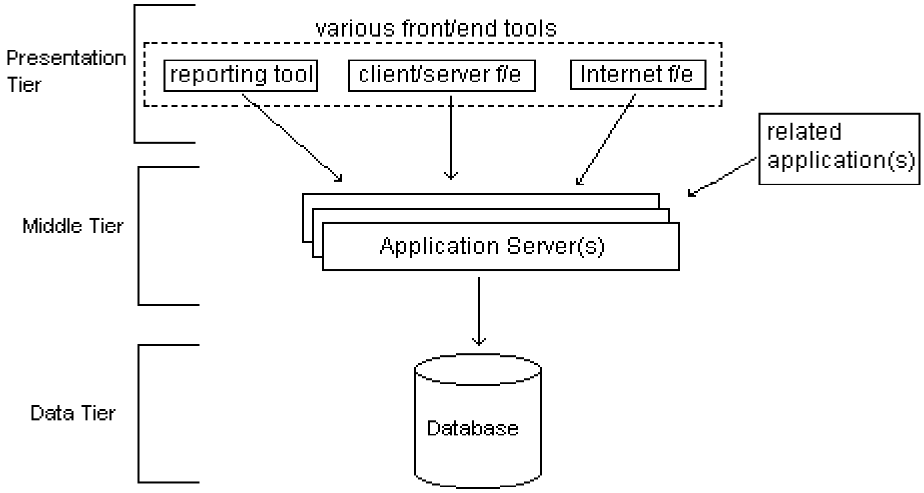
6. Architectural Mechanisms

Architectural mechanisms are common solutions to common problems that can be used during development to minimize complexity. Architecture mechanisms facilitate the evolution of architecturally significant aspects of the system.

Architectural Mechanism 1



We are using three-tier architecture model this program. The three-tier architecture model is the fundamental logical framework that divides an application's components into three tiers of service. These tiers do not correspond to physical locations on computers of the application or network, but to logical layers of the application. The three-tier architecture provides benefits such as re-usability, manageability, maintainability and scalability. The CTS implements the three-tier architecture as shown below.



Following are brief descriptions of each of the three tiers.

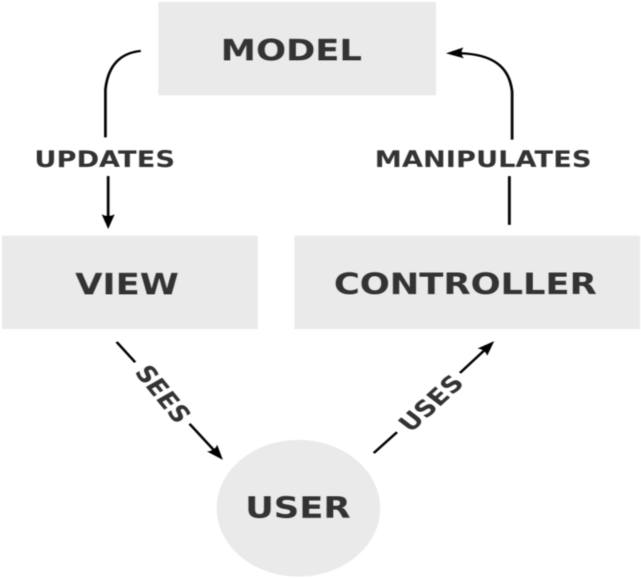
● **Presentation Tier:** This tier gives the user access to the application. It presents data to the user and may permit data manipulation and data entry.

● **Middle Tier:** This tier consists of business and data rules. It contains the application and business logic of the system. The components that exist on this tier can be used to enforce business rules, such as business algorithms or governmental regulations, and data rules, to keep the data structures consistent within the database.

● **Data Tier:** This tier interacts with persistent data usually stored in a database or in permanent storage. This is the layer that provides access to the DMBS and it can be accessed by the middle and sometimes the presentation tier.

Architectural Mechanism 2

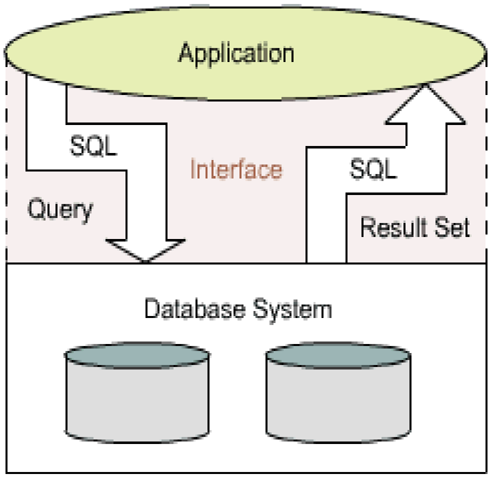
We are using MVC(Model-View-Controller) architecture to design this program. This is really useful for database systems. Because this control the system with 3-component. A brief description of the model is displayed in the diagram below.



MVC General Information:

- A model stores data that is retrieved according to commands from the controller and displayed in the view.
- A view generates an output presentation to the user based on changes in the model.
- A controller can send commands to the model to update the model's state.

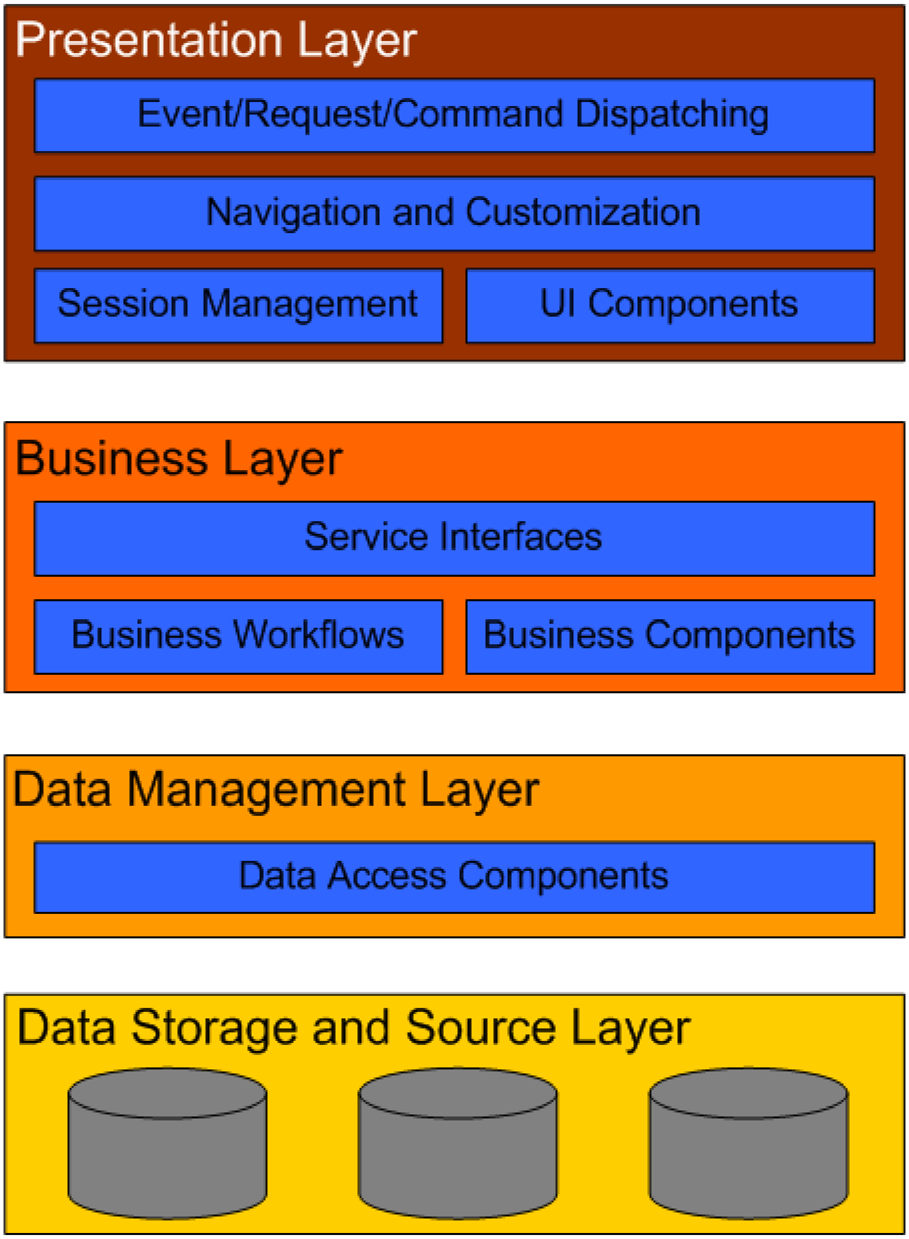
7. Key abstractions



Interface : User and librarian use interface.
Application: Application is used from users and librarians
Database: Database is used from applications.
These are the main components of the system. That realize what includes the system and who use this application. Persons and System elements are component in this system. We abstract the system elements with using software.

8. Layers or architectural framework

Layers:



Presentation Layer: The presentation layer contains components needed to interact with the user of the application. Examples of such components are web pages, rich-client forms, user interaction process components etc.

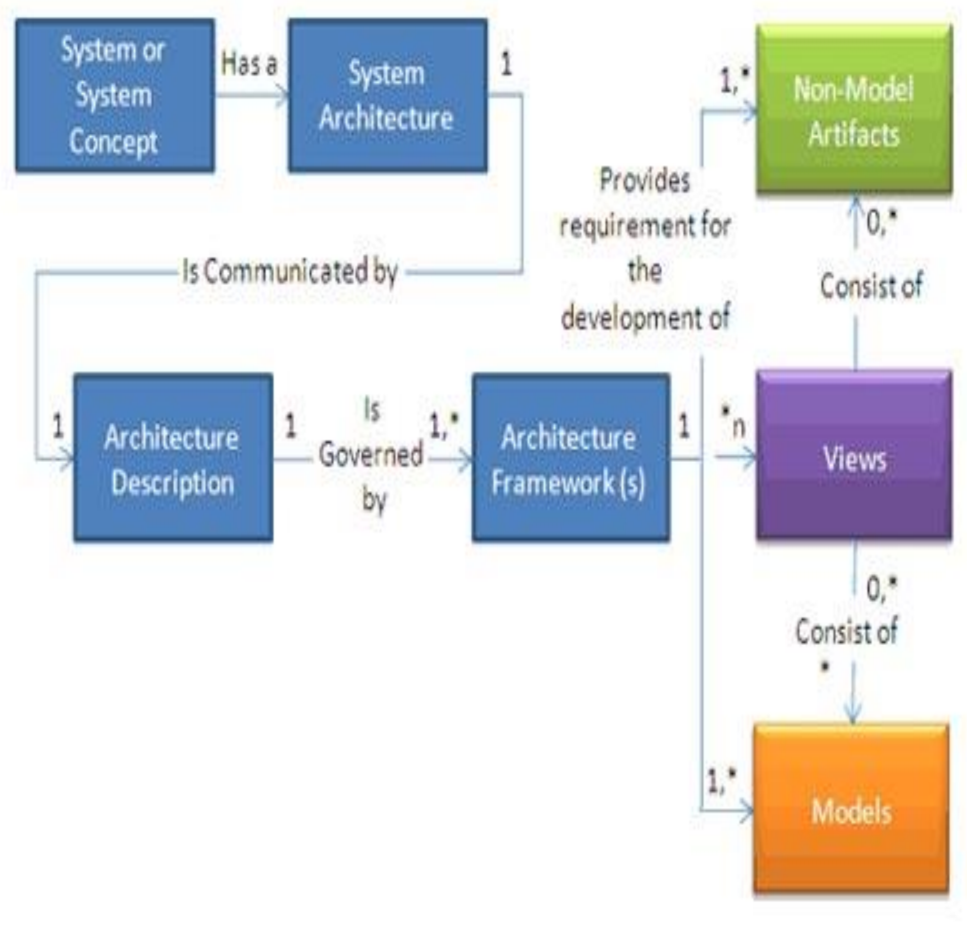
Business Layer: The business layer encapsulates the core business functionality of the application. Simple business functions can be implemented using stateless components, whereas complex, long-running transactions can be

implemented using stateful workflows. The business components are generally front-ended by a service interface that acts as a facade to hide the complexity of the business logic.
Data Access Layer: The data access layer provides a simple API for accessing and manipulating data. The components in this layer abstract the semantics of the underlying data access technology thus allowing the business layer to focus on business logic. Each component typically provides methods to perform Create, Read, Update, and Delete (CRUD) operations for a specific business entity.

Data Storage and Source Layer: Enterprise applications store their data in one or more data stores. Databases and file systems are two very common types of data stores. In addition, enterprise applications deal with external data providers, such as RSS feeds and real-time messages.

Architectural framework :

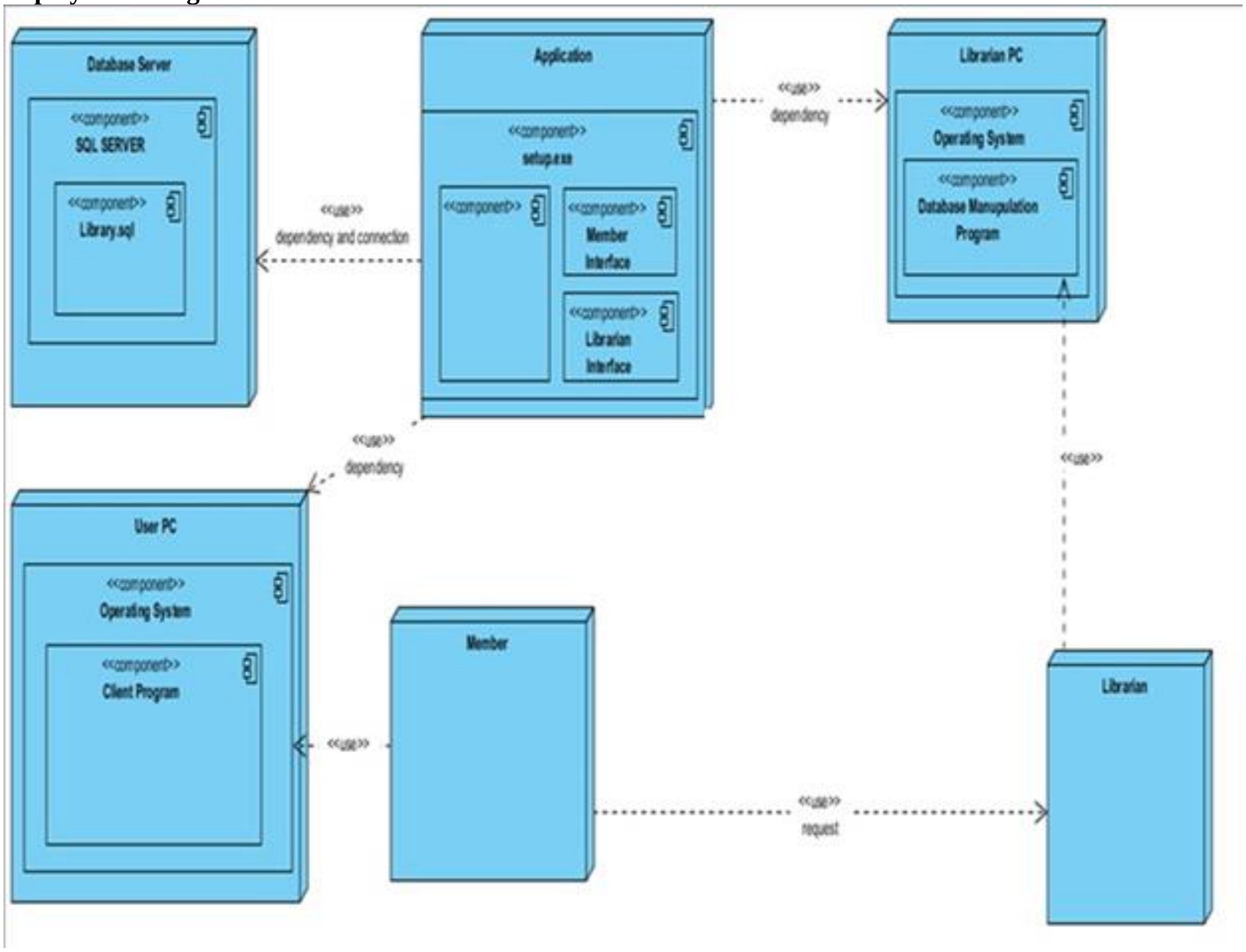
Definition: An architecture framework is an encapsulation of a minimum set of practices and requirements for artifacts that describe a system's architecture. Models are representations of how objects in a system fit structurally in and behave as part of the system. Views are a partial expression of the system from a particular perspective. A viewpoint is a set of representations (views and models) of an architecture that covers a stakeholder's issues.



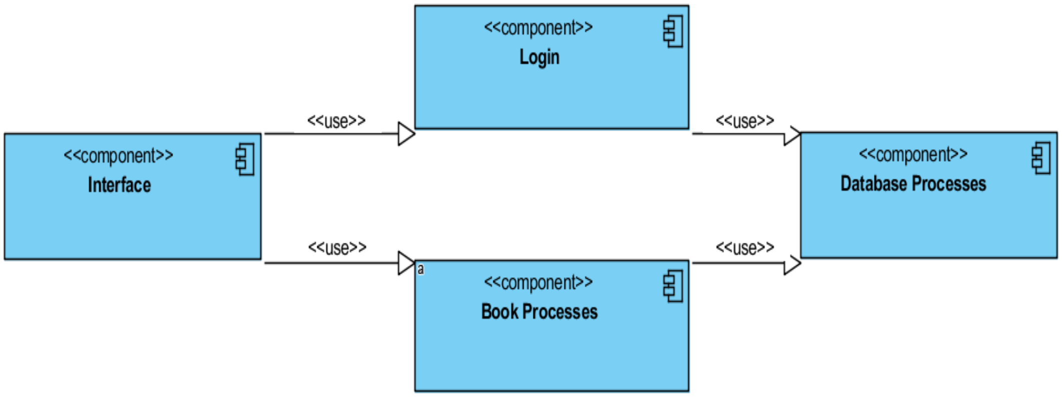
4+1 architectural view model is proper for designing our system. Because the views are used to describe the system from the viewpoint of different stakeholders, such as end-users, developers and project managers. So much more proper for our system. So we select this framework for our project

9. Architectural views

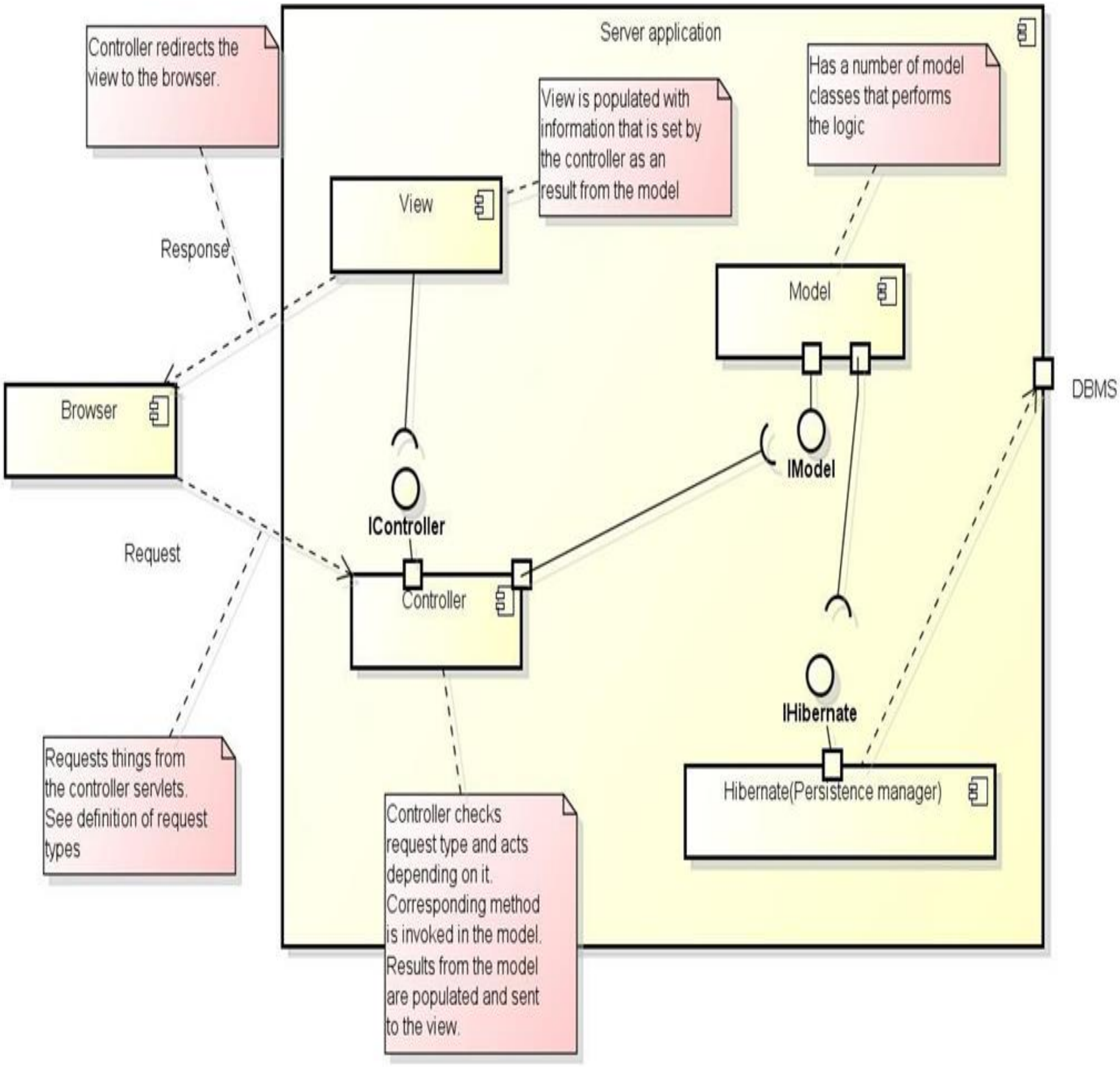
- **Deployment Diagram:**



- **Component Diagram:**

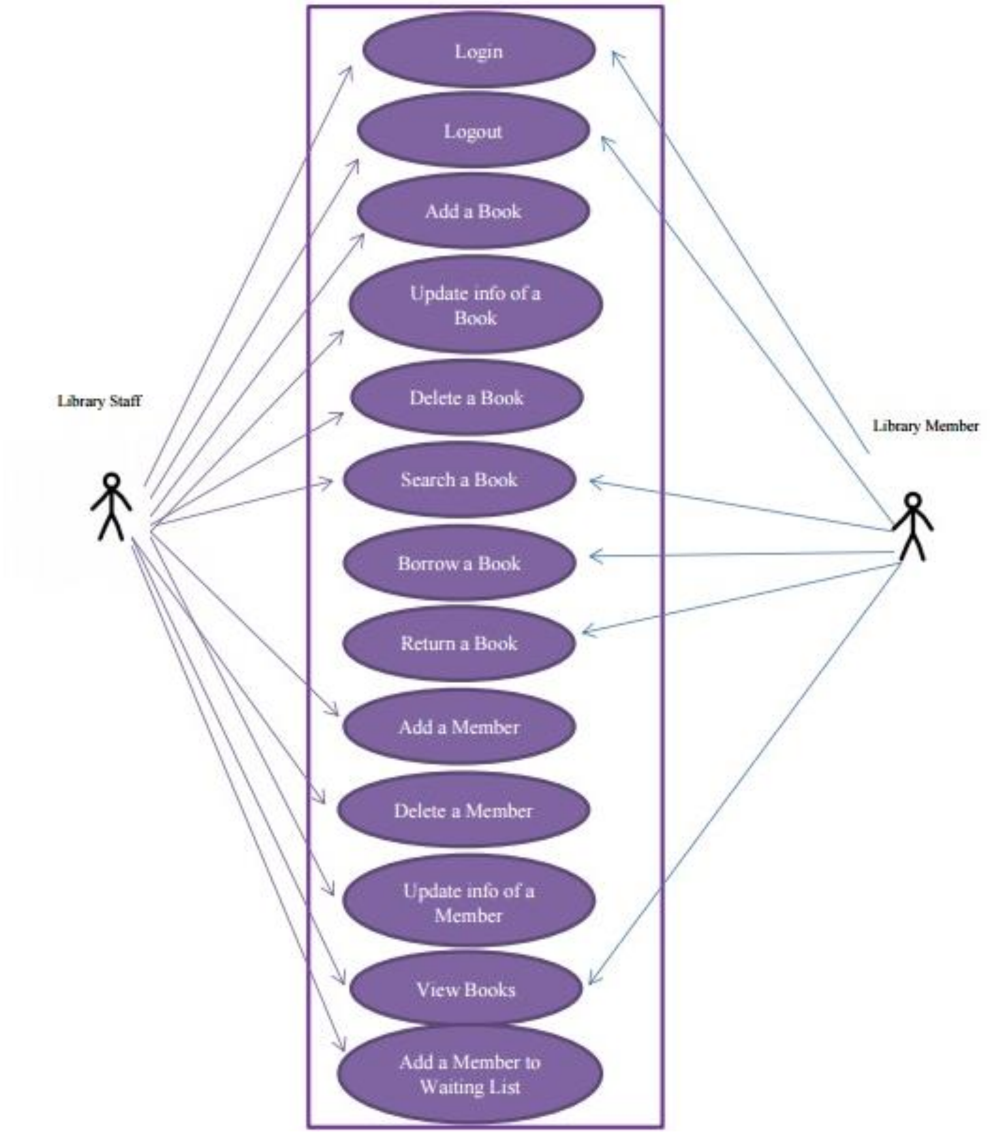


- **Package Diagram:**



Server application
Controller redirects the view to the browser.
View is populated with information that is set by the controller as an result from the model
Has a number of model classes that performs the logic
View
Response
Model
Browser
DBMS
IModel
IController
Request
Controller
IHibernate
Requests things from the controller servlets. See definition of request types
Hibernate(Persistence manager)
Controller checks request type and acts depending on it. Corresponding method is invoked in the model. Results from the model are populated and sent to the view.

- **Use case:**


Library Staff
Library Member
Login
Logout
Add a Book
Update info of a Book
Delete a Book
Search a Book
Borrow a Book
Return a Book
Add a Member
Delete a Member
Update info of a Member
View Books
Add a Member to Waiting List