**JStudyPlanner**

**(Spring MVC model with Integration of Hibernate ORM)**

**FINAL YEAR PROJECT REPORT**

*Submitted in partial fulfilment of the requirement for the award of the Degree* of

*Master of Computer Applications of BabaSaheb BhimRao Ambedkar University, Lucknow- 226025*

*By*

**SANTOSH KUMAR**

**Enroll. NO: 091/14**



Department of Computer Science

BabaSaheb Bhim Rao Ambedkar Universtiy

(A Central University)

Vidya Vihar , Raibariely Road, Lucknow - 22602

**BABA SAHEB BHIM RAO AMBEDKAR UNIVERSITY**

**( A CENTRAL UNIVERSITY )**

**LUCKNOW – 226025, UTTAR PRADESH**

**Candidate Declaration**

I hereby declare that the work which is being presented in the project work entitled

***“JStudyPlanner (Spring MVC model with Integration of Hibernate ORM)”*** in partial fulfillment of requirements for award of the degree of Master of Computer Application and submitted in the Department of Computer Science of BabaSaheb BhimRao Ambedkar University is an authentic record of my work carried out during the period of January 2017 to May 2017 under the supervision of Dr. Manoj Kumar, Assistant Professor, Department of Computer Science.

The Matter embodied in the project is a genuine work done by me and has not been submitted to any other university for the fulfillment of the requirement of any Degree/Diploma.

Date:  **/05/2017** Santosh Kumar

Place: Lucknow Roll No.-21055

The viva examination of Santosh Kumar has been held on

at

(Signature of Internal Guide)

Dr. Manoj Kumar

Assistant Professor (Signature of Examiner)

BBAU, Lucknow

**BABA SAHEB BHIM RAO AMBEDKAR UNIVERSITY**

**( A CENTRAL UNIVERSITY )**

**LUCKNOW – 226025, UTTAR PRADESH**



**CERTIFICATE**

This is to certify that this project report entitled ‘JStudyPlanner (Spring MVC model with Integration of Hibernate ORM)**’** is a confide record of the project work done by **Santosh Kumar, Roll no: 21055** in the year 2017 for the partial fulfilment of Master of Computer Applications Degree.

Project Guide Head of the Department

Internal Examiner External Exam

**ACKNOWLEDGEMENT**

It is with great pleasure and deep sense of gratitude that I acknowledge the contributions of the several individuals towards the successful completion of this project.

I am extremely grateful to **Dr. S.K. Dwivedi**, H.O.D, Department of Computer Science, BabaSaheb BhimRao Ambedkar University, Lucknow, whose encouragement and timely advice have helped me to great extent.

I express my heartfelt gratitude to my project guide **Dr. M. Kumar**, Department of Computer Science, BBAU, for his invaluable guidance and support throughout the course of my project as well as my academic career at BBAU (BabaSaheb BhimRao Ambedkar University, Lucknow-226025).

I express my immense pleasure and thankfulness to all the teachers and support staff of Department of MCA, BBAU for their cooperation and support.

I sincerely thank to \_\_\_\_\_\_\_\_\_\_\_\_\_, HR manager **Edureka\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**.,for providing me the necessary infrastructure for the project. I am deeply indebted to Mr.\_\_\_\_\_\_\_\_\_, Team Leader, for his guidance during the development of this project.

Words cannot express the immense gratitude I have for my parents who have been instrumental in shaping my career. Last but not the least; I acknowledge the contributions of my friends whose ideas and support have helped me a great extent.

Santosh Kr. Kushwaha

**PREFACE**

The Spring Framework is a collection of small, well-focused, loosely coupled Java frameworks that can be used independently or collectively to build industrial strength applications of many different types. The JStudyPlanner application is designed to show how the Spring application frameworks can be used to build simple, but powerful database-oriented applications. It will demonstrate the use of Spring's core functionality:

* JavaBeans based application configuration using Inversion-Of-Control
* Model-View-Controller Web Presentation Layer
* Practical database access through Hibernate
* Manage Database Connection pooling through Hibernate
* Unit Testing through JUnit framework
* Declarative Transaction Management using AOP
* Data Validation that supports but is not dependent on the Presentation Layer
* URL security & annotation based security through Spring Security frameworks

The Spring frameworks provide a great deal of useful infrastructure to simplify the tasks faced by application developers. This infrastructure helps developers to create applications that are:

* Concise: by handling a lot of the complex control flow that is needed to use the Java API's, such as JDBC, JNDI, JTA, RMI, and EJB.
* Flexible: by simplifying the process of external application configuration through the use of Reflection and JavaBeans. This allows the developer to achieve a clean separation of configuration data from application code. All application and web application objects, including validators, workflow controllers, and views, are JavaBeans that can be configured externally.
* Testable: by supplying an interface based design to maximize plug ability. This facilitates unit testing of Business Logic without requiring the presence of application or live database servers.
* Maintainable: by facilitating a clean separation of the application layers. It most importantly helps maintain the independence of the Business Layer from the Presentation layer. JStudyPlanner demonstrates the use of a Model-View-Controller based web presentation framework that can work seamlessly with many different types of view technologies. The Spring web application framework helps developers to implement their Presentation as a clean and thin layer focused on its main missions of translating user actions into application events and rendering model data.

**JStudyPlanner Application Requirements**

The application requirement is for an information system that is accessible through a web browser. The users of the application are university admins, staff & students of a university who in the course of their work need to view and manage information regarding the courses available at their various campuses. The sample application supports the following:

**Use Cases**

* + View a list of campuses and their available courses.
  + View information pertaining to a enrolled students.
  + Update the information pertaining to a enrolled students
  + Add a new student to the system
  + View information pertaining to a staff
  + Update the information pertaining to a staff
  + Add a new staff to the system
  + View information pertaining to a campus
  + Add information pertaining to a campus.
  + Update the information pertaining to a campus etc.

**Business Rules**

There should be a system admin of the application system who has authority to create new user admins.

**JStudyPlanner Application Design & Implementation**

**Server Technology**

The sample application should be usable with any Java EE web application container that is compatible with the Servlet 2.4 and JSP 2.0 specifications. Some of the deployment files provided are designed specifically for Apache Tomcat. These files specify container-supplied connection-pooled data sources. It is not necessary to use these files. The application has been configured by default to use a data source without connection pooling to simplify usage. Configuration details are provided in the Developer Instructions section. The view technologies that are to be used for rendering the application are Java Server Pages (JSP) along with the Java Standard Tag Library (JSTL).

**Database Technology**

The sample application uses a relational database for data storage. Support has been provided for a choice of 1 of 2 database selections, MySql or Oracle. MySql version 5.1.2 is the default choice. It is possible to easily configure the application to use either database. Configuration details will be provided in the Developer Instructions section.

**Development Environment**

A copy of the Spring runtime library jar file is provided with the application along with some of the other required jar files. The developer will need to obtain the following tools externally, all of which are freely available:

• Java SDK 1.5.x

• Maven 4.0.x

• Tomcat 8.x.x, or some other web application container

• JUnit 4.4 - needed to run the tests

• (Optional) Oracle11g.x with Oracle Connector/J 5.x

**NOTE:** The version numbers listed are those that were used in the development of the JStudyPlanner application. Other versions of the same tools may or may not work.

JStudy Planner

Spring MVC Model with Integration of Hibernate as ORM

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **1. INTRODUCTION** | | **1** |
| **2. ORGANIZATION PROFILE** | | **5** |
| **3. SYSTEM ANALYSIS** | | **10** |
| 3.1 | Existing System | 11 |
| 3.2 | Objective of system | 12 |
| 3.3 | Proposed System | 15 |
| 3.4 | Modules Description | 16 |
| 3.5 | Feasibility Study | 18 |

|  |  |  |
| --- | --- | --- |
| **4. SYSTEM REQUIREMENT & SPECIFICATIONS** | | **20** |
| 4.1 | Hardware Specification | 20 |
| 4.2 | Software Specification | 21 |
| 4.3 | Software Description | 28 |

|  |  |  |
| --- | --- | --- |
| **5. SYSTEM DESIGN** | | **30** |
| 5.1 | Design Overview | 31 |
| 5.2 | Database Design | 32 |
| 5.3 | Data Flow Diagrams | 38 |
| 5.4 UML Diagrams | | 46 |
| 5.5 ER Diagram | | 53 |

|  |  |  |
| --- | --- | --- |
| **6. SYSTEM TESTING** | | **54** |
| 6.1 | Unit Testing | 54 |
| 6.2 | Integration Testing | 54 |
| 6.3 Validation Testing | | 55 |
| 6.4 | Output Testing | 57 |
| **7. SCREENSHOTS** | | **59** |

|  |  |
| --- | --- |
| **8. USER/OPERATIONAL MANUAL** | **79** |
|  |  |
| **9. CONCLUSION** | **82** |
| **10. BIBLIOGRAPHY** | **84** |

1. **Introduction**

# **1. INTRODUCTION**

JStudyPlanner is a comprehensive tool providing information about the University. It is an online search engine to find list of courses available at different campuses of the university.

The portal provides detailed information on staffs, campuses, students, programs running at various campuses, majors provided at these programs, core & elective courses available at these majors and details of each semester term. The portal has four different level of users – System Admin, Application Admins, Staffs & Students. The user has to register themselves and according to their role, they have different access privileges.

* 1. **System Users**
     1. **System Administrator**

System Administrator is a person who administers the portal. He updates the details of the educational services provided by the website. He adds edits and deletes application admins, campuses, staffs or students of the university.

* + 1. **Application Administrator**

Application Administrator is a person who needs to adds edits and deletes campuses, staffs or students of the university.

* + 1. **Staff**

Staff is a person who needs to adds edits and deletes term, program, major, course, student, schedule and view enrolments at various terms.

* + 1. **Student**

Student is a person who needs to adds edits and deletes course & schedule and view his details.

* 1. **Features**
* The website provides full-fledged details of various campuses, programs, courses offered by them and academic details of each of the registered students of the university.
* The user has to get registered if they would like to enjoy the facilities offered by the website according to the role.
* The administrator administers the website; he edits, adds or delete the required details.
  1. Purpose

The main objective of the system is that providing university information’s and their services to students so that they can opt their desired programs. This is a totally web based search engine, its main aim is to provide all education services details in one place, user does not need to search different sites to get information about the University. JStudyPlanner gives the full details of all programs available at various campuses information of the university.

## Advantages

* JStudyPlanner main advantage is all sorts of educational services information of the university will be available in one place
* It provides the details of affiliated campuses and respective course details at one system.
* In JStudyPlanner we can add the new campuses, courses to increase the selecting options for the students
* User should have a one desktop or system is enough for using all facilities staying anywhere in the world.

**2.Organization Profile**

1. **COMPANY PROFILE:**
   1. **Introduction**

**Edureka** is an interactive, online learning platform. It offers live, instructor-led courses that cater mainly to working professionals who want to enhance their skills. Founded in mid-2011 and headquartered in [Bangalore](https://en.wikipedia.org/wiki/Bangalore), India, the company is privately owned and operated by Brain4ce Education Ltd.

* 1. **History**

Lovleen Bhatia, an IIT-BHU alumnus exited his earlier venture to start Edureka in mid-2011. Lovleen partnered with Kapil Tyagi, an IIT Bombay alumni, to co-found this venture. Bhatia and Tyagi started off the company from a small office in Noida, with just 4 employees and an investment of Rs.1.5 crore.

The company launched its first pilot project to the final year students at ABES Engineering College in Ghaziabad, in mid-2011. A two-month training program was organized in partnership with a few Bangalore-based software companies and focused on mobile app development courses. Working professionals from the industry were its first instructors.

Over the next eight months, Edureka added 30 new instructors from the industry and trained around 1,200 students. Edureka shifted its base from Noida to Bengaluru in 2012 where they set up a small office in Indiranagar. Further in 2013, they expanded their office and relocated to Sigma Tech Park in Whitefield, Bengaluru.

* 1. **Courses**

Edureka has partnered with organizations like PMI as its registered education provider and Talend to provide certifications in project management and big data.

* 1. **Awards**

In November 2014, Edureka was ranked number 1 in Deloitte’s Technology Fast 50 India 2014.

In June 2015, Business Today named Edureka as one of the Coolest Startups of 2015.

**3. System Analysis**

* 1. **System Analysis**

The process of System Analysis is the application of the system approach to problem solving using computers. The ingredients are system elements, processes and the technology. It means that one has to understand the system concept and how organizations operate as system, and then design appropriate computer based system that will meet an organizations requirement. It is actually a customized approach to the use of the computer for problem solving. System Analysis works with users to identify goals and build systems to achieve them. System development revolves around a life cycle that begins with the recognition of the user needs. Following a feasibility study, the stages of the cycle are evolution of the present system, information gathering, cost-benefit analysis, detailed design and the implementation of the candidate system.

System Analyst is the person who does functions such as feasibility study, eventual design and implemented of a computer system. The role requires a combination of skills, experience, personality and common sense. System Analysis has a history dating to late 1890. The most critical phase of managing projects is planning. The initial investigation has the objective of determining whether the users request has potential merit. When the initial investigation is completed, the user receives a system proposal summarizing the findings and recommendations of the analyst. A key part of feasibility analysis is gathering information about the present system. Help analyst assess the effectiveness of the present system and provide the groundwork for recommending a candidate system.

System flowcharting based on the data gathered commit to a physical implementation of the candidate system before one has a complete understanding of the logical requirements.

The system specifications can be often redundant. The alternative ways of designing a candidate system are structured tools such as the dataflow diagrams, data dictionary and structured English.

For the purpose of feasibility study, we need to consider the economic, technical and behavioural factors in system development. The team develops system flow charts that identify the characteristics of candidate system, evaluate performance of each system, weigh system performance and cost data and select the best candidate system for the job. Data gathering, traditional or structured is only part of the system analysis. The next steps are examining the data gathered, assessing the situation, looking at the alternatives and recommending a solution. The best system is taken after considering the cost and benefits of the alternate system.

## 3.2 Scope

### 3.2.1 Existing System with Limitations

The existing system has lot of flaws, because here they have to work more on manually instead of working on system. In present system they have to go through lots of news papers and magazines to know the information about the details of Campuses and Programs. They won’t get the detailed information with out going the different sites /News papers / Magazines. In this system they have to spent lot of time, money etc to getting the exact information. JStudyPlanner helps students to plan their career by opting program at their opted campus. This system is developed to overcome the flaws and problems in existing system. They want replace with a system which gives the total information in a one tool, which is user friendly, time saving etc.

### 3.2.2 Objective of System

An information system is required for a university which will provide all information about the campuses, programs & courses provided, at one place. It will help students to plan their courses according to the career needs.

### 3.2.3 Proposed System Features

* JStudyPlanner is an educational portal providing reliable and comprehensive information on campuses & programs available at the university.
* Students and their parents can find information on campuses, programs, courses.
* Proposed system is developed with convenient modules, it takes the user where he wants to go.
  1. **Modules Description**
     1. **Open Access Module**
        1. View list of all Campuses.
        2. View list of all Programs available for enrolment.
* View Major available under the selected program.
* View Core Courses available under the selected program.
* View Elective Courses available under the selected program.
* View the details of all courses available under the selected program’s major.
* View the prerequisite for that selected course.
  + - 1. View list of all Majors available for enrolment
* View compulsory courses under the selected major
* View Elective courses under selected major.
* View Course details under selected major.
* View the prerequisite for the selected course.
  + - 1. View list of all Courses available for enrolment
* View details of each course.
* View prerequisite for the selected course.
  + 1. **User Login Module**

There are four types of user for this information system – System Admin, Application Admin, Staff & Student. According to their role they have different access authorizations.

* + 1. **System Admin Module**
       1. **Admin Sub Module**
* View list of existing Admins
* Create New Application Admins
* Update & Delete existing Admins
  + - 1. **Staff Sub Module**
* View list of existing Staffs
* Create New Staff
* Update & Delete existing Staffs
  + - 1. **Student Sub Module**
* View list of existing Students
* Create New Student
* Update & Delete existing Students
  + 1. **Application Admin Module**
       1. **Staff Sub Module**
* View list of existing Staffs
* Create New Staff
* Update & Delete existing Staffs
  + - 1. **Student Sub Module**
* View list of existing Students
* Create New Student
* Update & Delete existing Students
  + - 1. **Campus Sub Module**
* View list of existing Campuses.
* Create New Campus.
* Update & Delete existing Campuses.
  + 1. **Staff Module**
       1. Term Sub Module
* View list of existing Terms
* Create New Term
* Update & Delete existing Terms
  + - 1. Programs Sub Module
* View list of existing Programs
* Create New Program
* Update & Delete existing Programs
  + - 1. Major Sub Module
* View list of Majors available for enrolment
* Create New Majors
* Update & Delete existing Majors.
  + - * 1. Course Sub-Sub Module
  + View list of Course available at selected Major
  + Add & Remove Compulsory Courses available at that Major
    - 1. Course Sub Module
* View list of all available courses.
* Create New Course
* Update & Delete existing courses.
* Add & Remove prerequisite courses for the selected course.
  + - 1. Student Sub Module
* View list of all enrolled students.
* Create new Student
* Update & Delete enrolled students.
  + - 1. Course Schedule Sub Module
* View the courses available for enrolment in various campuses.
* Allocate course available for enrolment.
  + - 1. Enrolment Sub Module
* View list of all enrolments.
  + 1. **Student Module**
* View his own details
* View all the courses planned or enrolled.
* Update & Delete course-availability schedule of his/her opted courses.

* 1. **Feasibility Study**

During system analysis, a feasibility study of the proposed system was carried out to see whether it was beneficial to the organization. Two key considerations that are involved in the feasibility study and the result of the feasibility study are given below:

* + 1. **Technical Feasibility**

Technical Feasibility is done around the existing system environment and to what extent it can support the proposed system. While considering the technical factors of the organization that it presently have, it is sufficient to implement the new system. The new system can use the existing LAN inside the company and no other communication channel is required.

* + 1. **Economic Feasibility**

The system was developed is user friendly, needless training will improve the working environment. Justification for any capital outlay is that it will increase profit, reduce expenditure or improve the quality of service or goods, which in turn may be expected to provide increased profits. Economic feasibility is the most frequently used for evaluating the effectiveness of the candidate system more commonly known as cost/benefit analysis, the procedure is to determine the benefits and savings that are expected from a candidate system and compare them with the existing system. If the benefits of the candidate system out weight the existing, the decision is made to design and implement the system. In the case of "Intra-Mailing", the system reduces additional expenses for conventional network system and its installation.

* + 1. **Behavioural Feasibility**

People are inherently resistant to change, and computers have been known to facilitate change. An estimate should be made about the reaction of the user staff towards the development of a computerized system. Computer installations have something to do with turnover, transfers and changes in job status. The introduction of a candidate system requires special effort to educate, sell and train the staff for conducting the business. Behavioural feasibility shows up to what extend the user accept the system. Since "Intra-Mailing” is designed to be fast and easy to use, and it is a much more efficient communication system than dialling telephone or conducting a meeting, the system is feasible.

The proposed system was found to be technically, economically, and behaviourally feasible. The system was developed user friendly, needless training and improves the working environment. Justification for any capital outlay is that it will increase profit, reduce expenditure or improve the quality of service or goods, which in turn may be expected to provide increased profits.

**4. System Specification Requirement**

# 4. SPECIFICATION REQUIREMENT

## 4.1. Software requirement Specification

* HTML
* Bootstrap (A Framework for HTML, CSS & JavaScript)
* Spring Framework
* Hibernate Framework
* Apache Tiles Framework,
* SLF4J Logging Framework
* Junit Testing Framework
* Java Server Pages and Servlets
* Java Script, JDBC
* My SQL, Oracle 11g SE
* Apache Tomcat Server
* Win NT/2000

## 4.2. Hardware Requirement Specification

* Pentium III/IV
* 20 GB Hard Disk, 64 -128 MB RAM or Above
* 10 Mbps Network Interface Card.

## 4.3. Communication Protocols

|  |  |
| --- | --- |
| TCP/IP Protocol | Transmission Control Protocol/Internet Protocol |
| FTP Protocol | File Transfer Protocol |
| HTTP Protocol | Hyper Text Transfer Protocol |
| HTML | Hyper Text Markup Language |

**4.3.1. TCP /IP**

* TCP/IP stands for Transmission Control Protocol / Internet Protocol
* TCP/IP was developed prior to the OSI model. Therefore Layers of TCP/IP protocol suite do not match exactly with OSI model
* TCP/IP consists of 5 Layer
  + - Application Layer
    - Transport Layer
    - Internet Layer
    - Network Access Layer
    - Physical Layer

**4.3.1.1. Application Layer**

Provides access to the OSI environment for users and also provides distributed information service.

**4.3.1.2. Transport Layer**

Provides reliable transport, Transfer of data between end points

**4.3.1.3. Internet Layer**

It is used for establishing, maintaining and terminating connection.

**4.3.1.4. Network Layer**

Provides a reliable transfer of data across the physical link **s**ends blocks of data with necessary synchronization

**4.3.1.5. Physical Layer**

            Concerned with transmission of unstructured bit stream over physical    medium

deals with mechanical, electrical, functional and procedural characteristics to

access the physical medium

**4.3.2. FTP**

* Ftp Stands for File Transfer Protocol
* Ftp is the standard mechanism provided by TCP/IP. For copying a file    from one host to another.
* Transferring files may be simple and straight forward, But some problems    must be delt as follows
* Two systems may have different ways to represent text
  + - Two systems have different directory structure
    - All of these have been solved by Ftp in every simple and elegant approach
* Ftp differs from other client server applications in that it establishes two connections between the hosts, one connection is used for data transfer, the other for control information separation of commands and data transfer makes Ftp more efficient
* Ftp uses two well known TCP ports Port 21 is used for control connection,    and port 20 is used for data connection

**4.3.3. HTTP**

Http stands for Hypertext Transfer Protocol

* Http is a protocol used in the client side web browser used mainly in the    access of data on the World Wide Web
* To facilitate the access of documents distributed throughout the world,    Http uses Locators
* The URL stands for uniform resource locator for specifying any kind of     information on the internet
* The URL defines 4 things Protocol, Host computer, ports and path
* The protocol is the Client-Server program used to retrieve the document    .Many different protocols can retrieve a document among them are    Gopher,     Ftp ,Http, News And Telnet the Most common is HTTP
* The host is the computer on which the information is located although the    name of the computer can be alias which begin with the characters “WWW”    this is not mandatory

# 4.4. TECHNOLOGIES USED

## 4.4.1. SPRING FRAMEWORK

Spring framework is an open source Java platform that provides comprehensive infrastructure support for developing robust Java applications very easily and very rapidly. Spring framework was initially written by Rod Johnson and was first released under the Apache 2.0 license in June 2003.

Spring is lightweight when it comes to size and transparency. The basic version of Spring framework is around 2MB.

The core features of the Spring Framework can be used in developing any Java application, but there are extensions for building web applications on top of the Java EE platform. Spring framework targets to make J2EE development easier to use and promotes good programming practices by enabling a POJO-based programming model.

## 4.4.1.1. Benefits of Using the Spring Framework

Following is the list of few of the great benefits of using Spring Framework −

* Spring enables developers to develop enterprise-class applications using POJOs. The benefit of using only POJOs is that you do not need an EJB container product such as an application server but you have the option of using only a robust servlet container such as Tomcat or some commercial product.
* Spring is organized in a modular fashion. Even though the number of packages and classes are substantial, you have to worry only about the ones you need and ignore the rest.
* Spring does not reinvent the wheel, instead it truly makes use of some of the existing technologies like several ORM frameworks, logging frameworks, JEE, Quartz and JDK timers, and other view technologies.
* Testing an application written with Spring is simple because environment-dependent code is moved into this framework. Furthermore, by using JavaBeanstyle POJOs, it becomes easier to use dependency injection for injecting test data.
* Spring's web framework is a well-designed web MVC framework, which provides a great alternative to web frameworks such as Struts or other over-engine ered or less popular web frameworks.
* Spring provides a convenient API to translate technology-specific exceptions (thrown by JDBC, Hibernate, or JDO, for example) into consistent, unchecked exceptions.
* Lightweight IoC containers tend to be lightweight, especially when compared to EJB containers, for example. This is beneficial for developing and deploying applications on computers with limited memory and CPU resources.
* Spring provides a consistent transaction management interface that can scale down to a local transaction (using a single database, for example) and scale up to global transactions (using JTA, for example).

**4.4.1.2. Dependency Injection (DI)**

The technology that Spring is most identified with is the Dependency Injection (DI) flavor of Inversion of Control. The Inversion of Control (IoC) is a general concept, and it can be expressed in many different ways. Dependency Injection is merely one concrete example of Inversion of Control.

When writing a complex Java application, application classes should be as independent as possible of other Java classes to increase the possibility to reuse these classes and to test them independently of other classes while unit testing. Dependency Injection helps in gluing these classes together and at the same time keeping them independent.

What is dependency injection exactly? Let's look at these two words separately. Here the dependency part translates into an association between two classes. For example, class A is dependent of class B. Now, let's look at the second part, injection. All this means is, class B will get injected into class A by the IoC.

Dependency injection can happen in the way of passing parameters to the constructor or by post-construction using setter methods. As Dependency Injection is the heart of Spring Framework, we will explain this concept in a separate chapter with relevant example.

**4.4.1.3. Aspect Oriented Programming (AOP)**

One of the key components of Spring is the Aspect Oriented Programming (AOP) framework. The functions that span multiple points of an application are called cross-cutting concerns and these cross-cutting concerns are conceptually separate from the application's business logic. There are various common good examples of aspects including logging, declarative transactions, security, caching, etc.

The key unit of modularity in OOP is the class, whereas in AOP the unit of modularity is the aspect. DI helps you decouple your application objects from each other, while AOP helps you decouple cross-cutting concerns from the objects that they affect.

The AOP module of Spring Framework provides an aspect-oriented programming implementation allowing you to define method-interceptors and pointcuts to cleanly decouple code that implements functionality that should be separated.

Spring could potentially be a one-stop shop for all your enterprise applications. However, Spring is modular, allowing you to pick and choose which modules are applicable to you, without having to bring in the rest. The following section provides details about all the modules available in Spring Framework.

The Spring Framework provides about 20 modules which can be used based on an application requirement.



## 4.4.1.3. Spring Architecture

## 4.4.1.3.1. Core Container

The Core Container consists of the Core, Beans, Context, and Expression Language modules the details of which are as follows −

* The **Core** module provides the fundamental parts of the framework, including the IoC and Dependency Injection features.
* The **Bean** module provides BeanFactory, which is a sophisticated implementation of the factory pattern.
* The **Context** module builds on the solid base provided by the Core and Beans modules and it is a medium to access any objects defined and configured. The ApplicationContext interface is the focal point of the Context module.
* The **SpEL** module provides a powerful expression language for querying and manipulating an object graph at runtime.

## 4.4.1.3.2. Data Access/Integration

The Data Access/Integration layer consists of the JDBC, ORM, OXM, JMS and Transaction modules whose detail is as follows −

* The **JDBC** module provides a JDBC-abstraction layer that removes the need for tedious JDBC related coding.
* The **ORM** module provides integration layers for popular object-relational mapping APIs, including JPA, JDO, Hibernate, and iBatis.
* The **OXM** module provides an abstraction layer that supports Object/XML mapping implementations for JAXB, Castor, XMLBeans, JiBX and XStream.
* The Java Messaging Service **JMS** module contains features for producing and consuming messages.
* The **Transaction** module supports programmatic and declarative transaction management for classes that implement special interfaces and for all your POJOs.

## 4.4.1.3.3. Web

The Web layer consists of the Web, Web-MVC, Web-Socket, and Web-Portlet modules the details of which are as follows −

* The **Web** module provides basic web-oriented integration features such as multipart file-upload functionality and the initialization of the IoC container using servlet listeners and a web-oriented application context.
* The **Web-MVC** module contains Spring's Model-View-Controller (MVC) implementation for web applications.
* The **Web-Socket** module provides support for WebSocket-based, two-way communication between the client and the server in web applications.
* The **Web-Portlet** module provides the MVC implementation to be used in a portlet environment and mirrors the functionality of Web-Servlet module.

## 4.4.1.3.4. Miscellaneous

There are few other important modules like AOP, Aspects, Instrumentation, Web and Test modules the details of which are as follows −

* The **AOP** module provides an aspect-oriented programming implementation allowing you to define method-interceptors and pointcuts to cleanly decouple code that implements functionality that should be separated.
* The **Aspects** module provides integration with AspectJ, which is again a powerful and mature AOP framework.
* The **Instrumentation** module provides class instrumentation support and class loader implementations to be used in certain application servers.
* The **Messaging** module provides support for STOMP as the WebSocket sub-protocol to use in applications. It also supports an annotation programming model for routing and processing STOMP messages from WebSocket clients.
* The **Test** module supports the testing of Spring components with JUnit or TestNG frameworks.

## 4.4.2. Hiberante framework

Hibernate is a high-performance Object/Relational persistence and query service which is licensed under the open source GNU Lesser General Public License (LGPL) and is free to download. Hibernate not only takes care of the mapping from Java classes to database tables (and from Java data types to SQL data types), but also provides data query and retrieval facilities.

## 4.4.2.1. What is JDBC?

JDBC stands for **Java Database Connectivity** and provides a set of Java API for accessing the relational databases from Java program. These Java APIs enables Java programs to execute SQL statements and interact with any SQL compliant database.

JDBC provides a flexible architecture to write a database independent application that can run on different platforms and interact with different DBMS without any modification.

## Pros and Cons of JDBC

|  |  |
| --- | --- |
| **Pros of JDBC** | **Cons of JDBC** |
| * Clean and simple SQL processing * Good performance with large data * Very good for small applications * Simple syntax so easy to learn | * Complex if it is used in large projects * Large programming overhead * No encapsulation * Hard to implement MVC concept * Query is DBMS specific |

**4.4.2.2. Why Object Relational Mapping (ORM)?**

When we work with an object-oriented system, there's a mismatch between the object model and the relational database. RDBMSs represent data in a tabular format whereas object-oriented languages, such as Java or C# represent it as an interconnected graph of objects. Consider the following Java Class with proper constructors and associated public function:

public class Employee {

private int id;

private String first\_name;

private String last\_name;

private int salary;

public Employee() {}

public Employee(String fname, String lname, int salary) {

this.first\_name = fname;

this.last\_name = lname;

this.salary = salary;

}

public int getId() {

return id;

}

public String getFirstName() {

return first\_name;

}

public String getLastName() {

return last\_name;

}

public int getSalary() {

return salary;

}

}

Consider above objects need to be stored and retrieved into the following RDBMS table:

create table EMPLOYEE (

id INT NOT NULL auto\_increment,

first\_name VARCHAR(20) default NULL,

last\_name VARCHAR(20) default NULL,

salary INT default NULL,

PRIMARY KEY (id)

);

First problem, what if we need to modify the design of our database after having developed few pages or our application? Second, Loading and storing objects in a relational database exposes us to the following five mismatch problems.

|  |  |
| --- | --- |
| **Mismatch** | **Description** |
| Granularity | Sometimes you will have an object model which has more classes than the number of corresponding tables in the database. |
| Inheritance | RDBMSs do not define anything similar to Inheritance which is a natural paradigm in object-oriented programming languages. |
| Identity | A RDBMS defines exactly one notion of 'sameness': the primary key. Java, however, defines both object identity (a==b) and object equality (a.equals(b)). |
| Associations | Object-oriented languages represent associations using object references where as am RDBMS represents an association as a foreign key column. |
| Navigation | The ways you access objects in Java and in a RDBMS are fundamentally different. |

The **O**bject-**R**elational **M**apping (ORM) is the solution to handle all the above impedance mismatches.

## 4.4.What is ORM?

ORM stands for **O**bject-**R**elational **M**apping (ORM) is a programming technique for converting data between relational databases and object oriented programming languages such as Java, C# etc. An ORM system has following advantages over plain JDBC

|  |  |
| --- | --- |
| **S.N.** | **Advantages** |
| 1 | Lets business code access objects rather than DB tables. |
| 2 | Hides details of SQL queries from OO logic. |
| 3 | Based on JDBC 'under the hood' |
| 4 | No need to deal with the database implementation. |
| 5 | Entities based on business concepts rather than database structure. |
| 6 | Transaction management and automatic key generation. |
| 7 | Fast development of application. |

An ORM solution consists of the following four entities:

|  |  |
| --- | --- |
| **S.N.** | **Solutions** |
| 1 | An API to perform basic CRUD operations on objects of persistent classes. |
| 2 | A language or API to specify queries that refer to classes and properties of classes. |
| 3 | A configurable facility for specifying mapping metadata. |
| 4 | A technique to interact with transactional objects to perform dirty checking, lazy association fetching, and other optimization functions. |

## Java ORM Frameworks:

There are several persistent frameworks and ORM options in Java. A persistent framework is an ORM service that stores and retrieves objects into a relational database.

* Enterprise JavaBeans Entity Beans
* Java Data Objects
* Castor
* TopLink
* Spring DAO
* Hibernate

Hibernate is an Object-Relational Mapping(ORM) solution for JAVA and it raised as an open source persistent framework created by Gavin King in 2001. It is a powerful, high performance Object-Relational Persistence and Query service for any Java Application.

Hibernate maps Java classes to database tables and from Java data types to SQL data types and relieve the developer from 95% of common data persistence related programming tasks.

Hibernate sits between traditional Java objects and database server to handle all the work in persisting those objects based on the appropriate O/R mechanisms and patterns.



## Hibernate Advantages:

* Hibernate takes care of mapping Java classes to database tables using XML files and without writing any line of code.
* Provides simple APIs for storing and retrieving Java objects directly to and from the database.
* If there is change in Database or in any table then the only need to change XML file properties.
* Abstract away the unfamiliar SQL types and provide us to work around familiar Java Objects.
* Hibernate does not require an application server to operate.
* Manipulates Complex associations of objects of your database.
* Minimize database access with smart fetching strategies.
* Provides Simple querying of data.

## 4.4.2.4. Supported Databases:

Hibernate supports almost all the major RDBMS. Following is list of few of the database engines supported by Hibernate.

* HSQL Database Engine
* DB2/NT
* MySQL
* PostgreSQL
* FrontBase
* Oracle
* Microsoft SQL Server Database
* Sybase SQL Server
* Informix Dynamic Server

## 4.4.2.5. Supported Technologies:

Hibernate supports a variety of other technologies, including the following:

* XDoclet Spring
* J2EE
* Eclipse plug-ins
* Maven

**4.4.2.6. Hibernate Architecture Details**

The Hibernate architecture is layered to keep you isolated from having to know the underlying APIs. Hibernate makes use of the database and configuration data to provide persistence services (and persistent objects) to the application.

Following is a very high level view of the Hibernate Application Architecture.



Following is a detailed view of the Hibernate Application Architecture with few important core classes.



Hibernate uses various existing Java APIs, like JDBC, Java Transaction API(JTA), and Java Naming and Directory Interface (JNDI). JDBC provides a rudimentary level of abstraction of functionality common to relational databases, allowing almost any database with a JDBC driver to be supported by Hibernate. JNDI and JTA allow Hibernate to be integrated with J2EE application servers.

Following section gives brief description of each of the class objects involved in Hibernate Application Architecture.

## 4.4.2.6.1. Configuration Object

The Configuration object is the first Hibernate object you create in any Hibernate application and usually created only once during application initialization. It represents a configuration or properties file required by the Hibernate. The Configuration object provides two keys components:

* **Database Connection:** This is handled through one or more configuration files supported by Hibernate. These files are **hibernate.properties** and **hibernate.cfg.xml**.
* **Class Mapping Setup**

This component creates the connection between the Java classes and database tables..

## 4.4.2.6.2. Session Factory Object

Configuration object is used to create a SessionFactory object which inturn configures Hibernate for the application using the supplied configuration file and allows for a Session object to be instantiated. The SessionFactory is a thread safe object and used by all the threads of an application.

The SessionFactory is heavyweight object so usually it is created during application start up and kept for later use. You would need one SessionFactory object per database using a separate configuration file. So if you are using multiple databases then you would have to create multiple SessionFactory objects.

## 4.4.2.6.3. Session Object:

A Session is used to get a physical connection with a database. The Session object is lightweight and designed to be instantiated each time an interaction is needed with the database. Persistent objects are saved and retrieved through a Session object.

The session objects should not be kept open for a long time because they are not usually thread safe and they should be created and destroyed them as needed.

## 4.4.2.6.4. Transaction Object:

A Transaction represents a unit of work with the database and most of the RDBMS supports transaction functionality. Transactions in Hibernate are handled by an underlying transaction manager and transaction (from JDBC or JTA).

This is an optional object and Hibernate applications may choose not to use this interface, instead managing transactions in their own application code.

## 4.4.2.6.5. Query Object:

Query objects use SQL or Hibernate Query Language (HQL) string to retrieve data from the database and create objects. A Query instance is used to bind query parameters, limit the number of results returned by the query, and finally to execute the query.

## 4.4.2.6.6. Criteria Object:

Criteria object are used to create and execute object oriented criteria queries to retrieve objects.

## 4.4.3. JAVA SERVER PAGES

JSP not only enjoys cross-platform and cross-Web-server support, but effectively melds the power of server-side Java technology with features of static HTML pages.

## JSP Advantages

**Separation of static from dynamic content**: In JSP, the logic to generate the dynamic content is kept separate from the static presentation templates by encapsulating it within external Java beans components. When a page designer makes any changes to the presentation template, the JSP page is automatically recompiled and reloaded into the web server by the JSP engine.

**Write Once Run Anywhere:** JSP technology brings the “Write Once, Run anywhere” paradigm to interactive Web pages.

**Dynamic content can be served in a variety of formats:** There is nothing that mandates the static template data within a JSP page to be of a certain format.

## JSP Architecture

The purpose of JSP is to provide a declarative, presentation-centric method of developing servlets. JSP pages are subject to a translation phase and a request-processing phase. The translation phase is carried phase is carried out only once, unless the JSP page changes, in which case it is repeated. The JSP engine itself typically carries out the translation phase, when it receives a request for the JSP page for the first time.

**Life Cycle of A JSP:** Life cycle of a JSP consists of the following three methods:

\_jspInit

\_jspService

\_jspDestroy

### JSP Syntax

### Directives

JSPs can define information for the container with directives. Here is what directives look like in a general form:

   <%@ directive attribute="some Value" attribute="another Value" ... %>

There are three directives:

<%@ page ... %> specifies information that affects the page

* <%@ include ... %> includes a file at the location of the include directive (parsed)
* <%@ taglib ... %> allows the use of custom tags in the page

<%@ page language="java" %> will always be the first line of every JSP file.

#### Declarations

Declarations are used to specify supplemental methods and variables. You can think of these are the page's private functions; they can only be called by the JSP where they are defined, or by another JSP that includes it (using the <@ include > directive).

Here is a sample declaration:

<%! // this integer can be used anywhere in this JSP page

private int myVariable = -1; // this function can be called from anywhere in this JSP page

public boolean isPositive() {

return ( myVariable > 0 );

}

%>

#### Scriptlets

Scriptlets are bits of Java code. They can do anything but they will most likely concentrate on generating HTML code or setting up variables to be part of later expressions.

#### Expressions

Expressions are special-purpose mini-Scriptlets used for evaluating expressions. This could be something as simple as outputting the value of a variable, or a more complicated Java expression, like calling a function and outputting the result.

## 

## 4.4.4. Servlets

A servlet is a java programming language class that is used to extend the capabilities of servers that host applications access via a request-response programming mode. Servlets are Java technology’s answer to Common Gateway Interface (CGI) Programming. They are programs that run on a Web server, acting as middle layer between request coming from a Web browser or other HTTP client and databases of applications on the HTTP server.

**Read any data sent by the user**: This data usually entered in a form on a Web page, but could also come from a java applet or a custom HTTP client program.

**Look up any other information about the request that is embedded in the HTTP request**: This information includes details about browser capabilities, cookies, the host name of the requesting client, and so froth.

**Generate the results**: This process may require talking to a database, executing an RMI or CORBA call, invoking a legacy application, or computing the response directly.

**Format the results inside a document**: In most cases, this involves embedding the information inside an HTML page.

**Set the appropriate HTTP response parameters**: This means telling the browser what type of document is being returned (e.g.HTML), setting cookies and caching parameters, and other such tasks.

**Send the document back to the client:** This document may be sent in text format (HTML), binary format (GIF images), or even in a compressed format like gzip that is layered on top of some other underlying format.

The Javax.servlet and javax.servlet.http packages provide interfaces and classes for writing servlets. All servlets must implement the Servlet interface, which defines life-cycle methods. When implementing a generic service, you can use or extend the GenericServlet class provided with the java Servlet API. The HttpServlet classes provide methods, such as doGet and do Post, for handling HTTP-specific services.

To be a servlet, a class should extend HTTPServlet and override doGet or do Post (or both), depending on whether the data is being sent by GET or by POST. These methods take two arguments: An HttpServletRequest and an HttpServletResponse.The HttpServletRequest have methods that let you find out about incoming information such as FORM data, HTTP request headers, and the like. Finally, note that doGet and do Post are called by the service method, and sometimes you may want to override service directly.

**Servlet Life Cycle: The** life cycle of a servlet is controlled by the container in which the servlet has been deployed. When a request is mapped to a servlet, the container performs the following steps.

1. If an instance of the servlet does not exist, the Web container:

Loads the servlet class.

Creates an instance of the Servlet class.

Initializes the servlet instance by calling the init method.

2. Invokes the service method, passing request and response objects.

If the container needs to remove the servlet, it finalizes the servlet by calling the servlet’s destroy method.

## 4.4.5 . MySQL

We started out with the intention of using the MySQL database system to connect to our tables using our own fast low-level (ISAM) routines.This resulted in a new SQL interface to our database but with almost the same API interface as MySQL.

The following list describes some of the important characteristics of the MySQL Database Software. See also Section 1.6, “MySQL Development Roadmap”, for more information about current and upcoming features.

**Internals and Portability**

* Written in C and C++.
* Tested with a broad range of different compilers.
* Works on many different platforms.
* Uses GNU Automake, Autoconf, and Libtool for portability.
* APIs for C, C++, Eiffel, Java, Perl, PHP, Python, Ruby, and Tcl are available.

**Data Types**

* Many data types: signed/unsigned integers 1, 2, 3, 4, and 8 bytes long, FLOAT, DOUBLE, CHAR, VARCHAR, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP, YEAR, SET, ENUM, and OpenGIS spatial types.
* Fixed-length and variable-length records.

**Statements and Functions**

* Full operator and function support in the SELECT and WHERE clauses of queries. For example:
* mysql> SELECT CONCAT(first\_name, ' ', last\_name)
* FROM citizen

**Security**

A privilege and password system that is very flexible and secure, and that allows host-based verification. Passwords are secure because all password traffic is encrypted when you connect to a server.

**Scalability and Limits**

Handles large databases. We use MySQL Server with databases that contain 50 million records. We also know of users who use MySQL Server with 60,000 tables and about 5,000,000,000 rows.

**Connectivity**

Clients can connect to the MySQL server using TCP/IP sockets on any platform. On Windows systems in the NT family (NT, 2000, XP, or 2003), clients can connect using named pipes. On Unix systems, clients can connect using Unix domain socket files.

* In MySQL 4.1 and higher, Windows servers also support shared-memory connections if started with the --shared-memory option. Clients can connect through shared memory by using the --protocol=memory option.

**Localization**

* The server can provide error messages to clients in many languages. See Section 5.11.2, “Setting the Error Message Language”.
* Full support for several different character sets, including latin1 (cp1252), german, big5, ujis, and more. For example, the Scandinavian characters ‘å’, ‘ä’ and ‘ö’ are allowed in table and column names. Unicode support is available as of MySQL 4.1.

**Clients and Tools**

* MySQL Server has built-in support for SQL statements to check, optimize, and repair tables. These statements are available from the command line through the **mysqlcheck** client. MySQL also includes **myisamchk**, a very fast command-line utility for performing these operations on MyISAM tables. See Chapter 5, *Database Administration*.
* All MySQL programs can be invoked with the --help or -? options to obtain online assistance.

## 4.4.6. TOMCAT

Apache Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and Java Server Pages technologies. The Java Servlet and Java Server Pages specifications are developed by Sun under the Java Community Process.

Tomcat 5 implements the Servlet 2.4 and Java Server Pages 2.0 specifications and includes many additional features that make it a useful platform for developing and deploying web applications and web services.

Directories and files: **Directories and Files**

**$CATALINA\_HOME** represents the root of your Tomcat installation. When we say, "This information can be found in your $CATALINA\_HOME/README.txt file" we mean to look at the README.txt file at the root of your Tomcat install.

These are some of the key tomcat directories, all relative to **$CATALINA\_HOME**:

* **/bin** - Startup, shutdown, and other scripts. The \*.sh files (for Unix systems) are functional duplicates of the \*.bat files (for Windows systems). Since the Win32 command-line lacks certain functionality, there are some additional files in here.
* **/conf** - Configuration files and related DTDs. The most important file in here is server.xml. It is the main configuration file for the container.
* **/logs** - Log files are here by default.
* **/webapps** - This is where your webapps go.

**INSTALLING TOMCAT**

Installing Tomcat on Windows can be done easily using the Windows installer. **Installation as a service**: Tomcat will be installed as a Windows NT/2k/XP service no matter what setting is selected. Using the checkbox on the component page sets the service as "auto" startup, so that Tomcat is automatically startup when Windows starts. For optimal security, the service should be affected a separate user, with reduced permissions (see the Windows Services administration tool and its documentation).

* **Java location**: The installer will use the registry or the JAVA\_HOME environment variable to determine the base path of the JDK or a JRE. If only a JRE (or an incorrect path) is specified, Tomcat will run but will be unable to compile JSP pages at runtime. Either all webapps will need to be precompiled (this can be easily done using the Tomcat deployed), or the lib\tools. Jar file from a JDK installation must be copied to the common\lib path of the Tomcat installation.

## Architecture Overview

**Server:** In the Tomcat world, a Server represents the whole container. A **Server** element represents the entire Catalina servlet container. Therefore, it must be the single outermost element in the conf/server.xml configuration file. Its attributes represent the characteristics of the servlet container as a whole. Tomcat provides a default implementation of the Server interface, and this is rarely customized by users.

**4.4.7. Maven**

Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information.

What is Maven?

Maven is a project management and comprehension tool. Maven provides developers a complete build lifecycle framework. Development team can automate the project's build infrastructure in almost no time as Maven uses a standard directory layout and a default build lifecycle.

In case of multiple development teams environment, Maven can set-up the way to work as per standards in a very short time. As most of the project setups are simple and reusable, Maven makes life of developer easy while creating reports, checks, build and testing automation setups.

Maven provides developers ways to manage following:

* Builds
* Documentation
* Reporting
* Dependencies
* SCMs
* Releases
* Distribution
* mailing list

To summarize, Maven simplifies and standardizes the project build process. It handles compilation, distribution, documentation, team collaboration and other tasks seamlessly. Maven increases reusability and takes care of most of build related tasks.

**4.4.7.1. Maven History**

Maven was originally designed to simplify building processes in Jakarta Turbine project. There were several projects and each project contained slightly different ANT build files. JARs were checked into CVS.

Apache group then developed Maven which can build multiple projects together, publish projects information, deploy projects, share JARs across several projects and help in collaboration of teams.

**4.4.4.2. Maven Objective**

Maven primary goal is to provide developer

A comprehensive model for projects which is reusable, maintainable, and easier to comprehend.

plugins or tools that interact with this declarative model.

Maven project structure and contents are declared in an xml file, pom.xml referred as Project Object Model (POM), which is the fundamental unit of the entire Maven system. Refer to Maven POM section for more detail.

Convention over Configuration

Maven uses Convention over Configuration which means developers are not required to create build process themselves.

Developers do not have to mention each and every configuration detail. Maven provides sensible default behavior for projects. When a Maven project is created, Maven creates default project structure. Developer is only required to place files accordingly and he/she need not to define any configuration in pom.xml.

As an example, following table shows the default values for project source code files, resource files and other configurations. Assuming, ${basedir} denotes the project location:

|  |  |
| --- | --- |
| **Item** | **Default** |
| source code | ${basedir}/src/main/java |
| Resources | ${basedir}/src/main/resources |
| Tests | ${basedir}/src/test |
| distributable JAR | ${basedir}/target |
| Complied byte code | ${basedir}/target/classes |

In order to build the project, Maven provides developers options to mention life-cycle goals and project dependencies (that rely on Maven pluging capabilities and on its default conventions). Much of the project management and build related tasks are maintained by Maven plugins.

Developers can build any given Maven project without need to understand how the individual plugins work. Refer to Maven Plug-ins section for more detail.

POM stands for *Project Object Model*. It is fundamental Unit of Work in Maven. It is an XML file. It always resides in the base directory of the project as pom.xml.

The POM contains information about the project and various configuration detail used by Maven to build the project(s).

POM also contains the goals and plugins. While executing a task or goal, Maven looks for the POM in the current directory. It reads the POM, gets the needed configuration information, then executes the goal. Some of the configuration that can be specified in the POM are following:

* project dependencies
* plugins
* goals
* build profiles
* project version
* developers
* mailing list

Before creating a POM, we should first decide the project **group** (groupId), its **name**(artifactId) and its version as these attributes help in uniquely identifying the project in repository.

## Example POM

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.companyname.project-group</groupId>

<artifactId>project</artifactId>

<version>1.0</version>

</project>

It should be noted that there should be a single POM file for each project.

* All POM files require the **project** element and three mandatory fields: **groupId, artifactId,version.**
* Projects notation in repository is **groupId:artifactId:version.**
* Root element of POM.xml is **project** and it has three major sub-nodes :

|  |  |
| --- | --- |
| **Node** | **Description** |
| groupId | This is an Id of project's group. This is generally unique amongst an organization or a project. For example, a banking group com.company.bank has all bank related projects. |
| artifactId | This is an Id of the project.This is generally name of the project. For example, consumer-banking. Along with the groupId, the artifactId defines the artifact's location within the repository. |
| Version | This is the version of the project.Along with the groupId, It is used within an artifact's repository to separate versions from each other. For example:   *com.company.bank:consumer-banking:1.0*  *com.company.bank:consumer-banking:1.1.* |

**4.4.8. Git**

Git is a distributed revision control and source code management system with an emphasis on speed. Git was initially designed and developed by Linus Torvalds for Linux kernel development. Git is a free software distributed under the terms of the GNU General Public License version 2.

**Version Control System**

Version Control System (VCS) is a software that helps software developers to work together and maintain a complete history of their work.

Listed below are the functions of a VCS:

* Allows developers to work simultaneously.
* Does not allow overwriting each other’s changes.
* Maintains a history of every version.

Following are the types of VCS:

* Centralized version control system (CVCS).
* Distributed/Decentralized version control system (DVCS).
* In this chapter, we will concentrate only on distributed version control system and especially on Git. Git falls under distributed version control system.

**Distributed Version Control System**

Centralized version control system (CVCS) uses a central server to store all files and enables team collaboration. But the major drawback of CVCS is its single point of failure, i.e., failure of the central server. Unfortunately, if the central server goes down for an hour, then during that hour, no one can collaborate at all. And even in a worst case, if the disk of the central server gets corrupted and proper backup has not been taken, then you will lose the entire history of the project. Here, distributed version control system (DVCS) comes into picture.

DVCS clients not only check out the latest snapshot of the directory but they also fully mirror the repository. If the server goes down, then the repository from any client can be copied back to the server to restore it. Every checkout is a full backup of the repository. Git does not rely on the central server and that is why you can perform many operations when you are offline. You can commit changes, create branches, view logs, and perform other operations when you are offline. You require network connection only to publish your changes and take the latest changes.

**Advantages of Git**

Free and open source

Git is released under GPL’s open source license. It is available freely over the internet. You can use Git to manage property projects without paying a single penny. As it is an open source, you can download its source code and also perform changes according to your requirements.

Fast and small

As most of the operations are performed locally, it gives a huge benefit in terms of speed. Git does not rely on the central server; that is why, there is no need to interact with the remote server for every operation. The core part of Git is written in C, which avoids runtime overheads associated with other high-level languages. Though Git mirrors entire repository, the size of the data on the client side is small. This illustrates the efficiency of Git at compressing and storing data on the client side.

Implicit backup

The chances of losing data are very rare when there are multiple copies of it. Data present on any client side mirrors the repository, hence it can be used in the event of a crash or disk corruption.

Security

Git uses a common cryptographic hash function called secure hash function (SHA1), to name and identify objects within its database. Every file and commit is check-summed and retrieved by its checksum at the time of checkout. It implies that, it is impossible to change file, date, and commit message and any other data from the Git database without knowing Git.

No need of powerful hardware

In case of CVCS, the central server needs to be powerful enough to serve requests of the entire team. For smaller teams, it is not an issue, but as the team size grows, the hardware limitations of the server can be a performance bottleneck. In case of DVCS, developers don’t interact with the server unless they need to push or pull changes. All the heavy lifting happens on the client side, so the server hardware can be very simple indeed.

Easier branching

CVCS uses cheap copy mechanism, If we create a new branch, it will copy all the codes to the new branch, so it is time-consuming and not efficient. Also, deletion and merging of branches in CVCS is complicated and time-consuming. But branch management with Git is very simple. It takes only a few seconds to create, delete, and merge branches.

**DVCS Terminologies**

Local Repository

Every VCS tool provides a private workplace as a working copy. Developers make changes in their private workplace and after commit, these changes become a part of the repository. Git takes it one step further by providing them a private copy of the whole repository. Users can perform many operations with this repository such as add file, remove file, rename file, move file, commit changes, and many more.

Working Directory and Staging Area or Index

The working directory is the place where files are checked out. In other CVCS, developers generally make modifications and commit their changes directly to the repository. But Git uses a different strategy. Git doesn’t track each and every modified file. Whenever you do commit an operation, Git looks for the files present in the staging area. Only those files present in the staging area are considered for commit and not all the modified files.

Let us see the basic workflow of Git.

Step 1 : You modify a file from the working directory.

Step 2 : You add these files to the staging area.

Step 3 : You perform commit operation that moves the files from the staging area. After push operation, it stores the changes permanently to the Git repository.

Suppose you modified two files, namely “sort.c” and “search.c” and you want two different commits for each operation. You can add one file in the staging area and do commit. After the first commit, repeat the same procedure for another file.

# First commit

[bash]$ git add sort.c

# adds file to the staging area

[bash]$ git commit –m “Added sort operation”

# Second commit

[bash]$ git add search.c

# adds file to the staging area

[bash]$ git commit –m “Added search operation”

### Blobs

Blob stands for **B**inary **L**arge **Ob**ject. Each version of a file is represented by blob. A blob holds the file data but doesn’t contain any metadata about the file. It is a binary file, and in Git database, it is named as SHA1 hash of that file. In Git, files are not addressed by names. Everything is content-addressed.

### Trees

Tree is an object, which represents a directory. It holds blobs as well as other sub-directories. A tree is a binary file that stores references to blobs and trees which are also named as **SHA1** hash of the tree object.

### Commits

Commit holds the current state of the repository. A commit is also named by **SHA1** hash. You can consider a commit object as a node of the linked list. Every commit object has a pointer to the parent commit object. From a given commit, you can traverse back by looking at the parent pointer to view the history of the commit. If a commit has multiple parent commits, then that particular commit has been created by merging two branches.

### Branches

Branches are used to create another line of development. By default, Git has a master branch, which is same as trunk in Subversion. Usually, a branch is created to work on a new feature. Once the feature is completed, it is merged back with the master branch and we delete the branch. Every branch is referenced by HEAD, which points to the latest commit in the branch. Whenever you make a commit, HEAD is updated with the latest commit.

### Tags

Tag assigns a meaningful name with a specific version in the repository. Tags are very similar to branches, but the difference is that tags are immutable. It means, tag is a branch, which nobody intends to modify. Once a tag is created for a particular commit, even if you create a new commit, it will not be updated. Usually, developers create tags for product releases.

### Clone

Clone operation creates the instance of the repository. Clone operation not only checks out the working copy, but it also mirrors the complete repository. Users can perform many operations with this local repository. The only time networking gets involved is when the repository instances are being synchronized.

### Pull

Pull operation copies the changes from a remote repository instance to a local one. The pull operation is used for synchronization between two repository instances. This is same as the update operation in Subversion.

### Push

Push operation copies changes from a local repository instance to a remote one. This is used to store the changes permanently into the Git repository. This is same as the commit operation in Subversion.

### HEAD

HEAD is a pointer, which always points to the latest commit in the branch. Whenever you make a commit, HEAD is updated with the latest commit. The heads of the branches are stored in **.git/refs/heads/** directory.

[CentOS]$ ls -1 .git/refs/heads/

master

[CentOS]$ cat .git/refs/heads/master

570837e7d58fa4bccd86cb575d884502188b0c49

### Revision

Revision represents the version of the source code. Revisions in Git are represented by commits. These commits are identified by **SHA1** secure hashes.

### URL

URL represents the location of the Git repository. Git URL is stored in config file.

[tom@CentOS tom\_repo]$ pwd

/home/tom/tom\_repo

[tom@CentOS tom\_repo]$ cat .git/config

[core]

repositoryformatversion = 0

filemode = true

bare = false

logallrefupdates = true

[remote "origin"]

url = gituser@git.server.com:project.git

fetch = +refs/heads/\*:refs/remotes/origin/\*

**5. System Design**

**5. System Design**

## 

## 5.1 Data Flow

**Data Flow Diagrams**

There are three types of DFD’s they are

1. Context Level DFD
2. Top Level DFD and
3. Detailed Level DFD

**Context Level DFD**

In the Context Level the whole system is shown as a single process.

* No data stores are shown.
* Inputs to the overall system are shown together with data sources (as External entities).
* Outputs from the overall system are shown together with their destinations (as External entities).

### 

### 5.1.1 DFD



**1.Level DFD**

**System Admin User DFD**



**Application Admin User DFD**



**Staff User DFD**



**Student User DFD**



**5.2. USE CASE DIAGRAM**

* A use case diagram is a diagram that shows a set of use cases and actors and relationships.

**Contents**

* Use case commonly contain
* Use cases
* Actors
* Dependency, generalization and association relationships

**5.2.1. System Administrator Use Case**



**5.2.2. Application Administrator Use Case**



**5.2.3. Staff Use Case**



**5.2.3. Student Use Case**



## 

## 5.3 Process Flow

### 5.3.1 Activity Diagrams

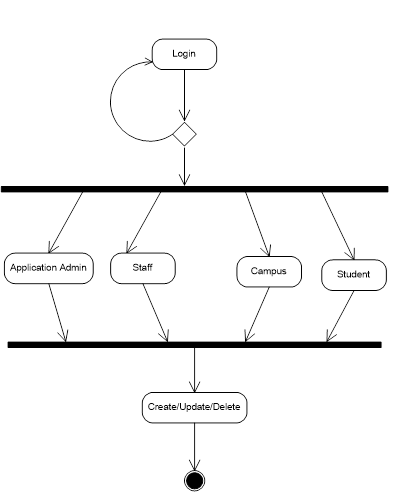
* An activity diagram shows the flow from activity to activity. An activity is an ongoing non-atomic execution within a state machine.
* Activities ultimately result in some action, which is made up of executable atomic computations that result in a change in state of the system or the return of a value.

Activity diagrams commonly contain

* Activity states and action states
* Transitions
* Objects

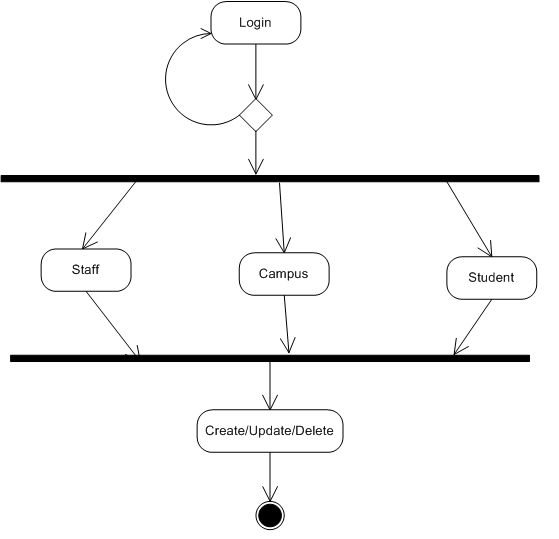
Like all other diagrams, activity diagrams may contain notes and constrains

**System Administrator Process**

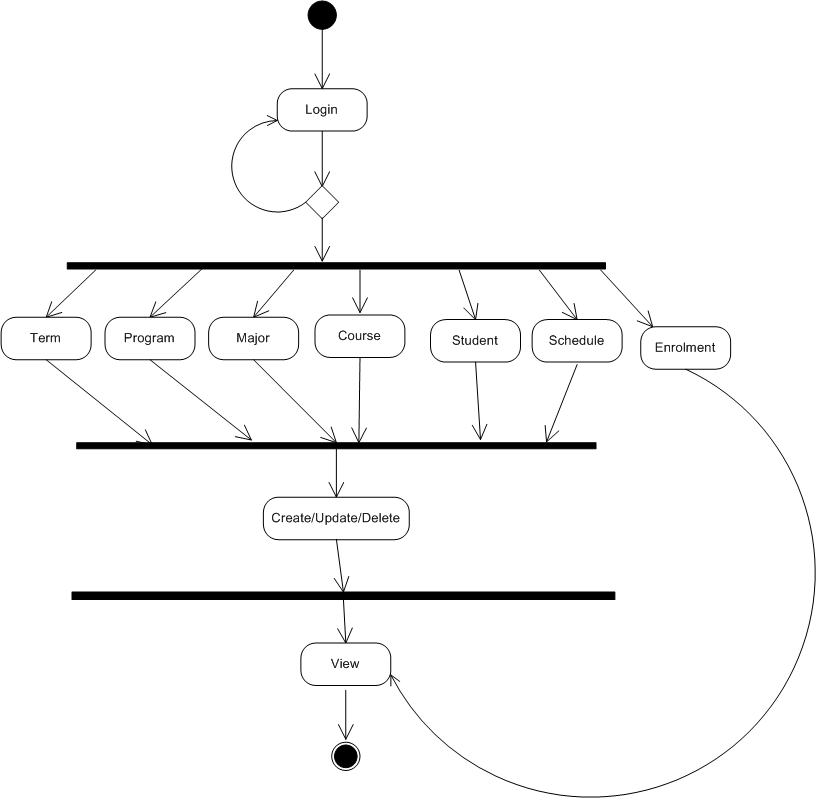


**Application Admin Process**

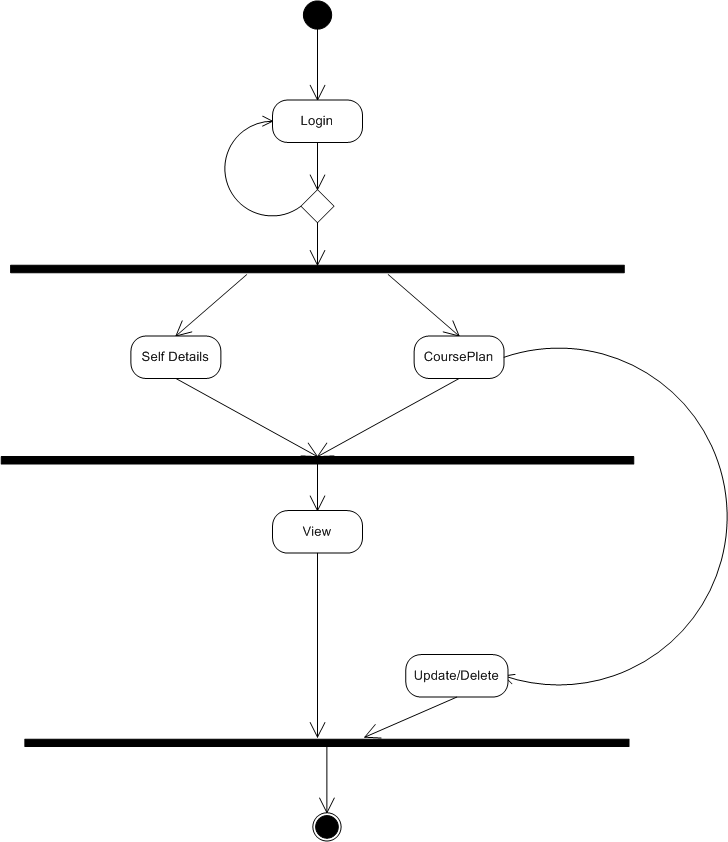
**Application Administator Process**



**Staff Process**

****

**Student Process**

****

## 

## 5.4. JPA Diagrams

**6. Testing**

**&**

**Implementation**

# 6. Testing and Implementation

**Testing**

* **Testing Methodologies**
* Black box Testing:
* White box Testing.
* Gray Box Testing.
* **Levels of Testing**
  + Unit Testing.
  + Module Testing.
  + Integration Testing.
  + System Testing.
  + User Acceptance Testing.
* **Types Of Testing**
  + Smoke Testing.
  + Sanitary Testing.
  + Regression Testing.
  + Re-Testing.
  + Static Testing.
  + Dynamic Testing.
  + Alpha-Testing.
  + Beta-Testing.
  + Monkey Testing.
  + Compatibility Testing.
  + Installation Testing.
  + Adhoc Testing.
  + Ext….
* **TCD (Test Case Documentation)**
* **STLC**
  + Test Planning.
  + Test Development.
  + Test Execution.
  + Result Analysis.
  + Bug-Tracing.
  + Reporting.
* **Microsoft Windows – Standards**
* **Manual Testing**
* **Automation Testing (Tools)**
  + Win Runner.
  + Test Director

**Testing**

* The process of executing a system with the intent of finding an error.
* Testing is defined as the process in which defects are identified, isolated, subjected for rectification and ensured that product is defect free in order to produce the quality product and hence customer satisfaction.
* Quality is defined as justification of the requirements
* Defect is nothing but deviation from the requirements
* Defect is nothing but bug.
* Testing --- The presence of bugs
* Testing can demonstrate the presence of bugs, but not their absence
* Debugging and Testing are not the same thing!
* Testing is a systematic attempt to break a program or the AUT
* Debugging is the art or method of uncovering why the script /program did not execute properly.

**Testing Methodologies**

* **Black box Testing**: is the testing process in which tester can perform testing on an application without having any internal structural knowledge of application.

Usually Test Engineers are involved in the black box testing.

* **White box Testing**: is the testing process in which tester can perform testing on an application with having internal structural knowledge.

Usually The Developers are involved in white box testing.

* **Gray Box Testing**: is the process in which the combination of black box and white box techniques are used.

**Levels of Testing**

* + Unit Testing.
  + Module Testing.
  + Integration Testing.
  + System Testing.
  + User Acceptance Testing.
* **Types Of Testing**
  + Smoke Testing.
  + Sanitary Testing.
  + Regression Testing.
  + Re-Testing.
  + Static Testing.
  + Dynamic Testing.
  + Alpha-Testing.
  + Beta-Testing.
  + Monkey Testing.
  + Compatibility Testing.
  + Installation Testing.
  + Adhoc Testing.
  + Ext….
* **TCD (Test Case Documentation)**
* **STLC**
  + Test Planning.
  + Test Development.
  + Test Execution.
  + Result Analysis.
  + Bug-Tracing.
  + Reporting.
* **Microsoft Windows – Standards**
* **Manual Testing**
* **Automation Testing (Tools)**
  + Win Runner.
  + Test Director

**Testing**

* The process of executing a system with the intent of finding an error.
* Testing is defined as the process in which defects are identified, isolated, subjected for rectification and ensured that product is defect free in order to produce the quality product and hence customer satisfaction.
* Quality is defined as justification of the requirements
* Defect is nothing but deviation from the requirements
* Defect is nothing but bug.
* Testing --- The presence of bugs
* Testing can demonstrate the presence of bugs, but not their absence
* Debugging and Testing are not the same thing!
* Testing is a systematic attempt to break a program or the AUT
* Debugging is the art or method of uncovering why the script /program did not execute properly.

**Testing Methodologies**

* **Black box Testing**: is the testing process in which tester can perform testing on an application without having any internal structural knowledge of application.

Usually Test Engineers are involved in the black box testing.

* **White box Testing**: is the testing process in which tester can perform testing on an application with having internal structural knowledge.

Usually The Developers are involved in white box testing.

* **Gray Box Testing**: is the process in which the combination of black box and white box techniques are used.

**Levels of Testing**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Module1*** ***Module2*** ***Module3***   |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | ***Units***   |  |  |  | | --- | --- | --- | |  |  |  | |  | ***Units***   |  |  |  | | --- | --- | --- | |  |  |  | |  | ***Units***   |  |  |  | | --- | --- | --- | |  |  |  | |   i/p ***Integration***  o/p i/p ***Integration o/p*** |

**Types of Testing**

* **Smoke Testing**: is the process of initial testing in which tester looks for the availability of all the functionality of the application in order to perform detailed testing on them. (Main check is for available forms)
* **Sanity Testing:** is a type of testing that is conducted on an application initially to check for the proper behavior of an application that is to check all the functionality are available before the detailed testing is conducted by on them.
* **Regression Testing:** is one of the best and important testing. Regression testing is the process in which the functionality, which is already tested before, is once again tested whenever some new change is added in order to check whether the existing functionality remains same.
* **Re-Testing:** is the process in which testing is performed on some functionality which is already tested before to make sure that the defects are reproducible and to rule out the environments issues if at all any defects are there.
* **Static Testing:** is the testing, which is performed on an application when it is not been executed. ex: GUI, Document Testing
* **Dynamic Testing:** is the testing which is performed on an application when it is being executed. ex: Functional testing.
* **Alpha Testing:** it is a type of user acceptance testing, which is conducted on an application when it is just before released to the customer.
* **Beta-Testing:** it is a type of UAT that is conducted on an application when it is released to the customer, when deployed in to the real time environment and being accessed by the real time users.
* **Monkey Testing:** is the process in which abnormal operations, beyond capacity operations are done on the application to check the stability of it in spite of the users abnormal behaviour.
* **Compatibility testing:** it is the testing process in which usually the products are tested on the environments with different combinations of databases (application servers, browsers…etc) In order to check how far the product is compatible with all these environments platform combination.
* **Installation Testing:** it is the process of testing in which the tester try to install or try to deploy the module into the corresponding environment by following the guidelines produced in the deployment document and check whether the installation is successful or not.
* **Adhoc Testing:** Adhoc Testing is the process of testing in which unlike the formal testing where in test case document is used, with out that test case document testing can be done of an application, to cover that testing of the future which are not covered in that test case document. Also it is intended to perform GUI testing which may involve the cosmotic issues.

**TCD (Test Case Document)**

**Test Case Document Contains**

* **Test Scope (or) Test objective**
* **Test Scenario**
* **Test Procedure**
* **Test case**

This is the sample test case document for the Acadamic details of student project:

**Test scope**

* Test coverage is provided for the screen “ Acadamic status entry” form of a student module of university management system application
* Areas of the application to be tested

**Test Scenario**

* When the office personals use this screen for the marks entry, calculate the status details, saving the information on student’s basis and quit the form.

**Test Procedure**

* The procedure for testing this screen is planned in such a way that the data entry, status calculation functionality, saving and quitting operations are tested in terms of Gui testing, Positive testing, Negative testing using the corresponding Gui test cases, Positive test cases, Negative test cases respectively

**Test Cases**

* Template for Test Case

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| T.C.No | Description | Exp | Act | Result |
|  |  |  |  |  |

**Guidelines for Test Cases**

1. **GUI Test Cases**

* Total no of features that need to be check
* Look & Feel
* Look for Default values if at all any (date & Time, if at all any require)
* Look for spell check

**Example for GUI Test cases**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| T.C.  No | Description | Expected value | Actual value | Result |
| 1 | Check for all the features in  the screen | The screen must contain  all the features |  |  |
| 2 | Check for the alignment of  the objects as per the validations | The alignment should be  in proper way |  |  |

1. **Positive Test Cases:**

* The positive flow of the functionality must be considered
* Valid inputs must be used for testing
* Must have the positive perception to verify whether the requirements are justified.

**Example for Positive Test cases:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| T.C.  No | Description | Expected value | Actual value | Result |
| 1 | Check for the date Time  Auto Display | The date and time of the system must be displayed |  |  |
| 2 | Enter the valid Roll no into the student roll no field | It should accept |  |  |

1. **Negative Test Cases:**

* Must have negative perception.
* Invalid inputs must be used for test.

**Example for Negative Test cases**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| T.C.  No | Description | Expected value | Actual  value | Result |
| 1 | Try to modify the information in date and time | Modification should not  be allow |  |  |
| 2 | Enter invalid data in to the student details form, click on  Save | It should not accept invalid data, save should not allow |  |  |

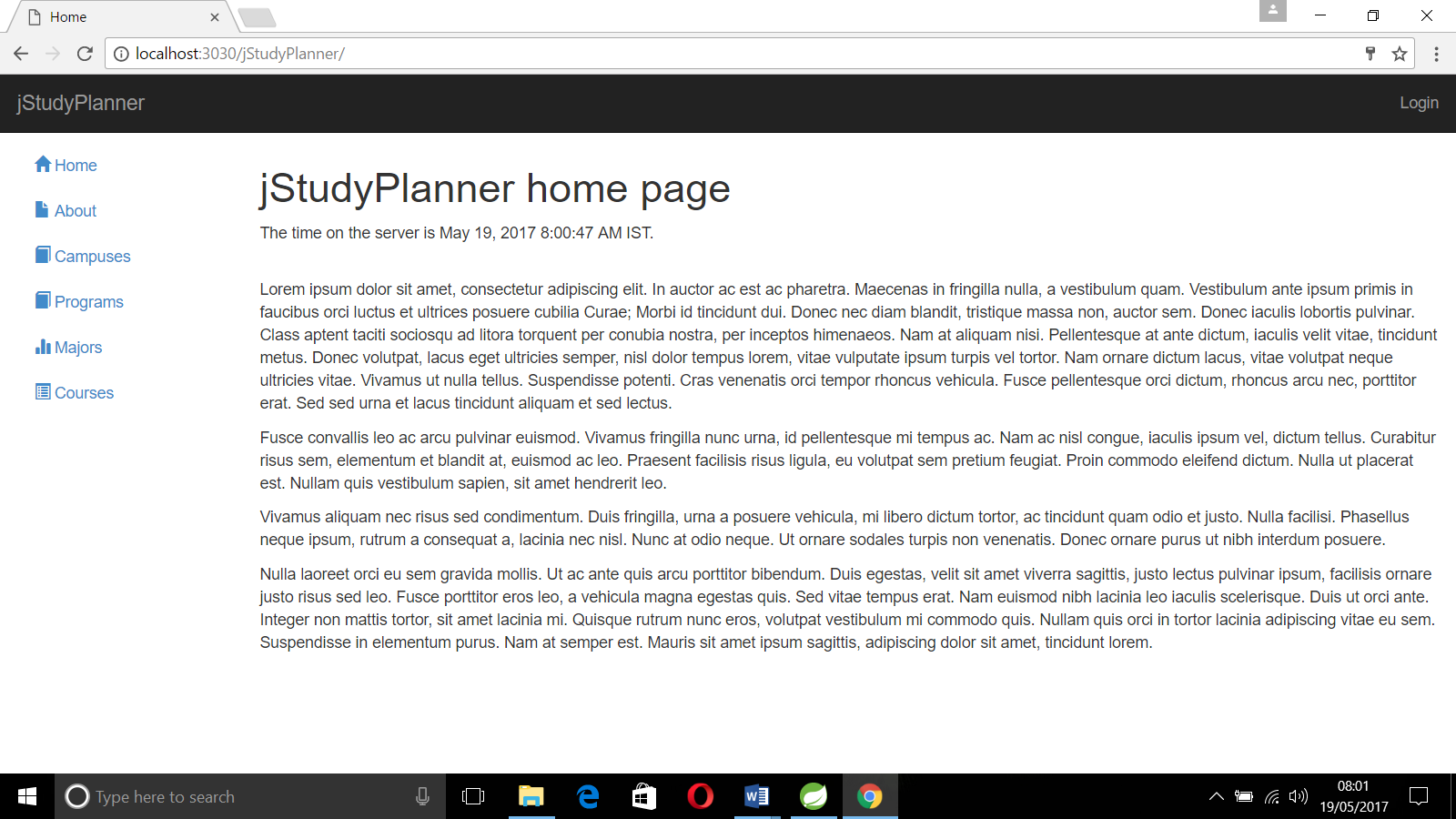
# 

**8. Screen Shorts**

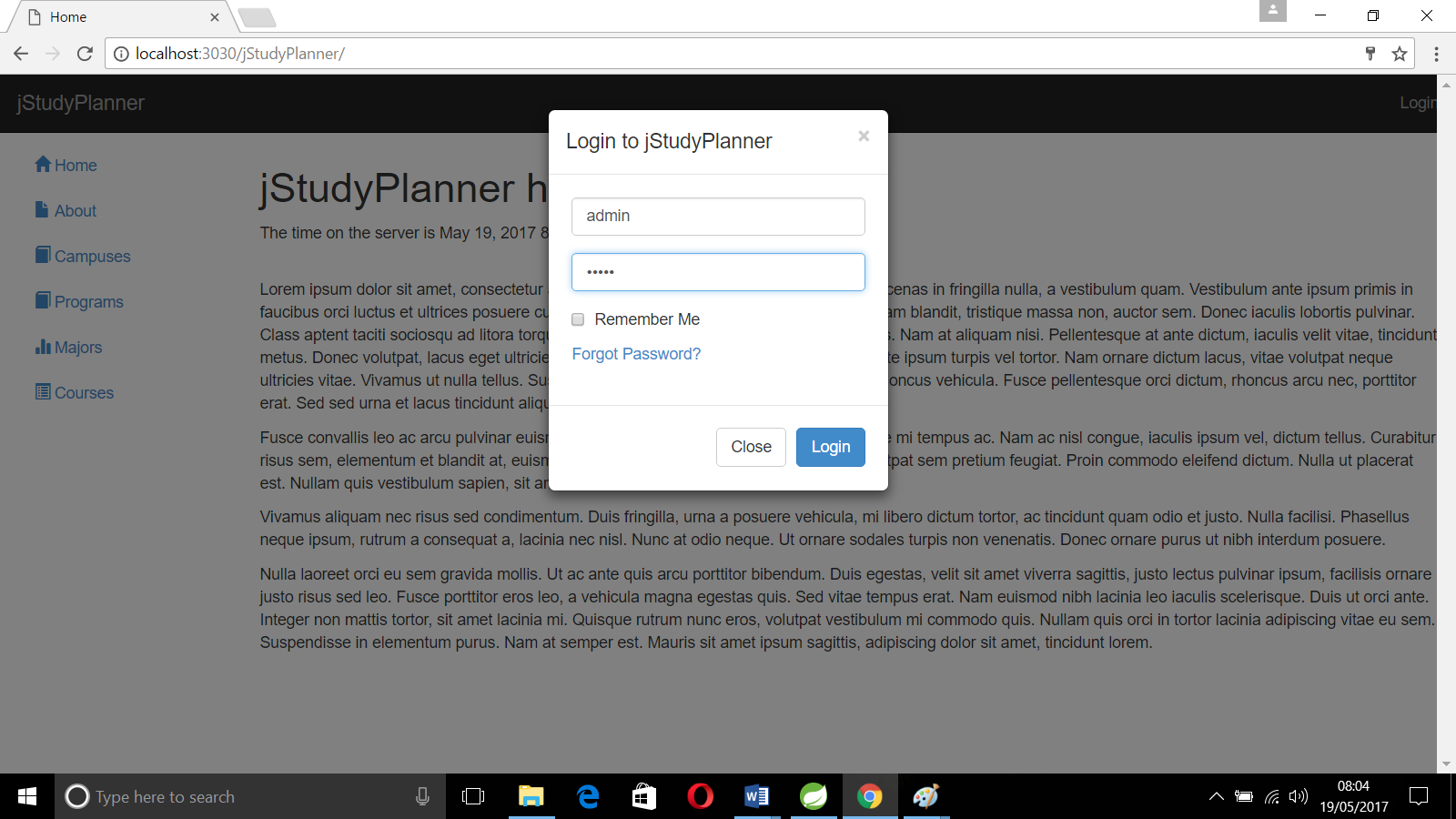
# 8. OUT-PUT RESULTS

## 8.1 Screens

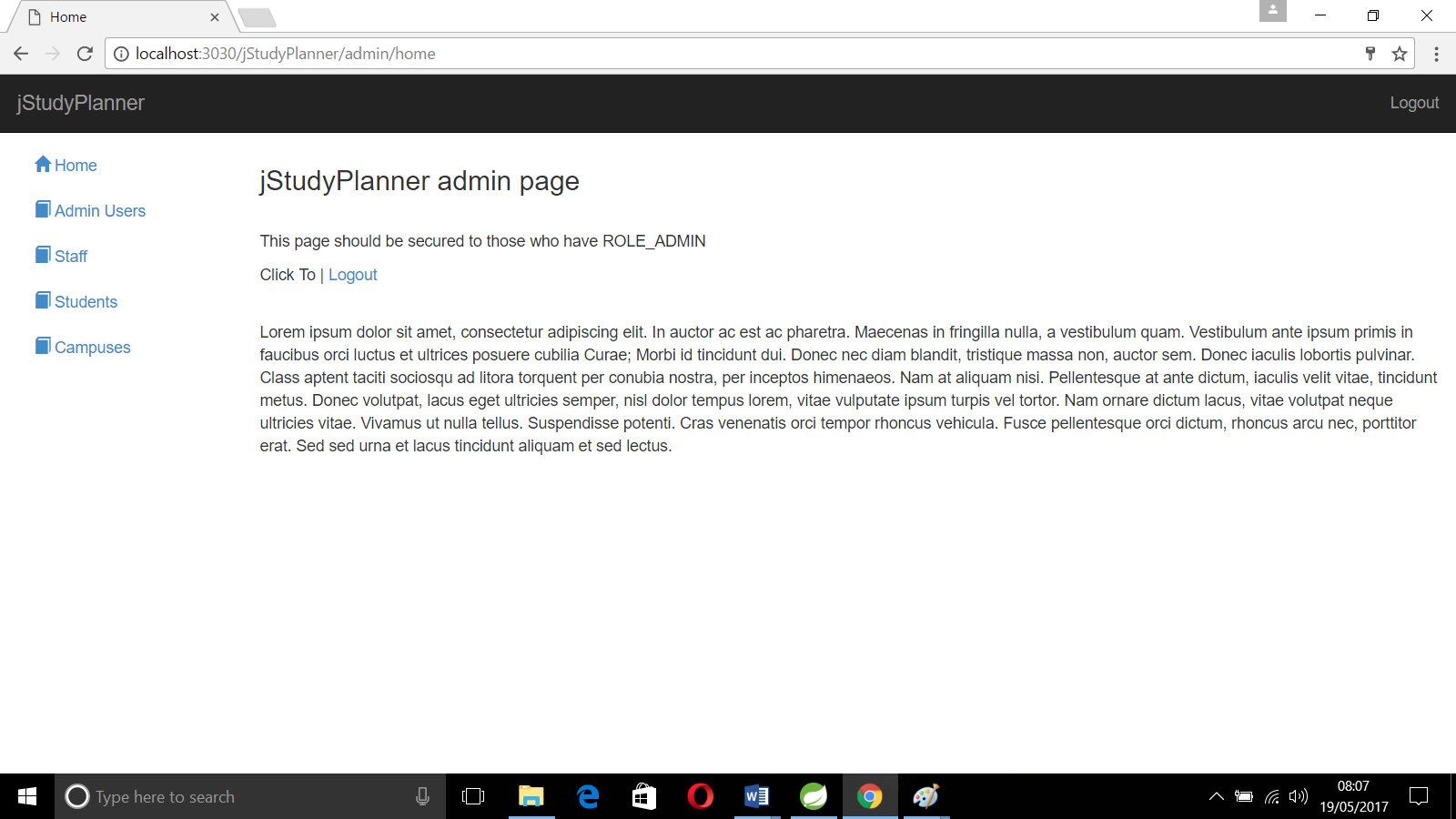
**8.1.1. Home Page**

****

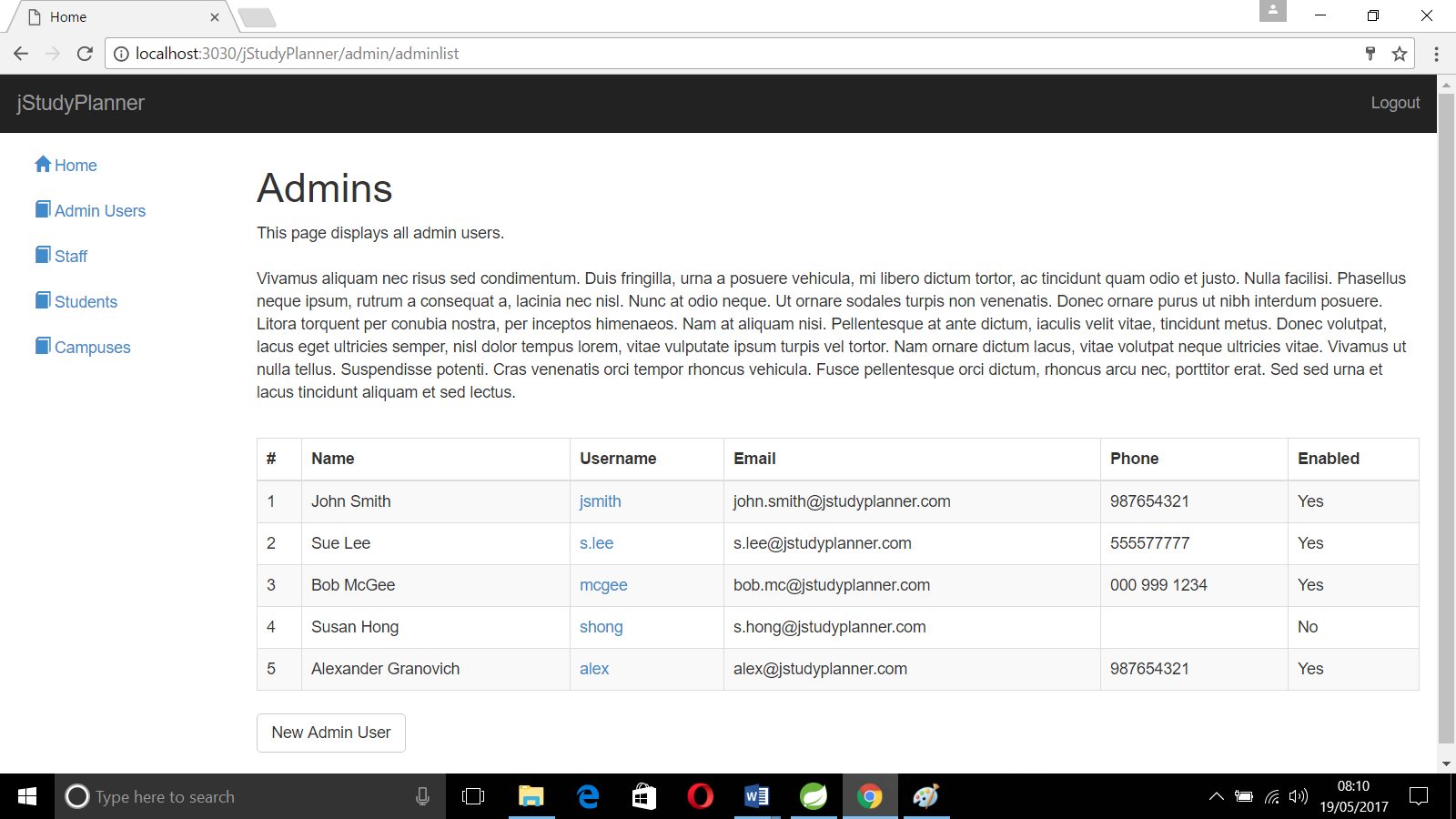
**8.1.2. Login Page**

****

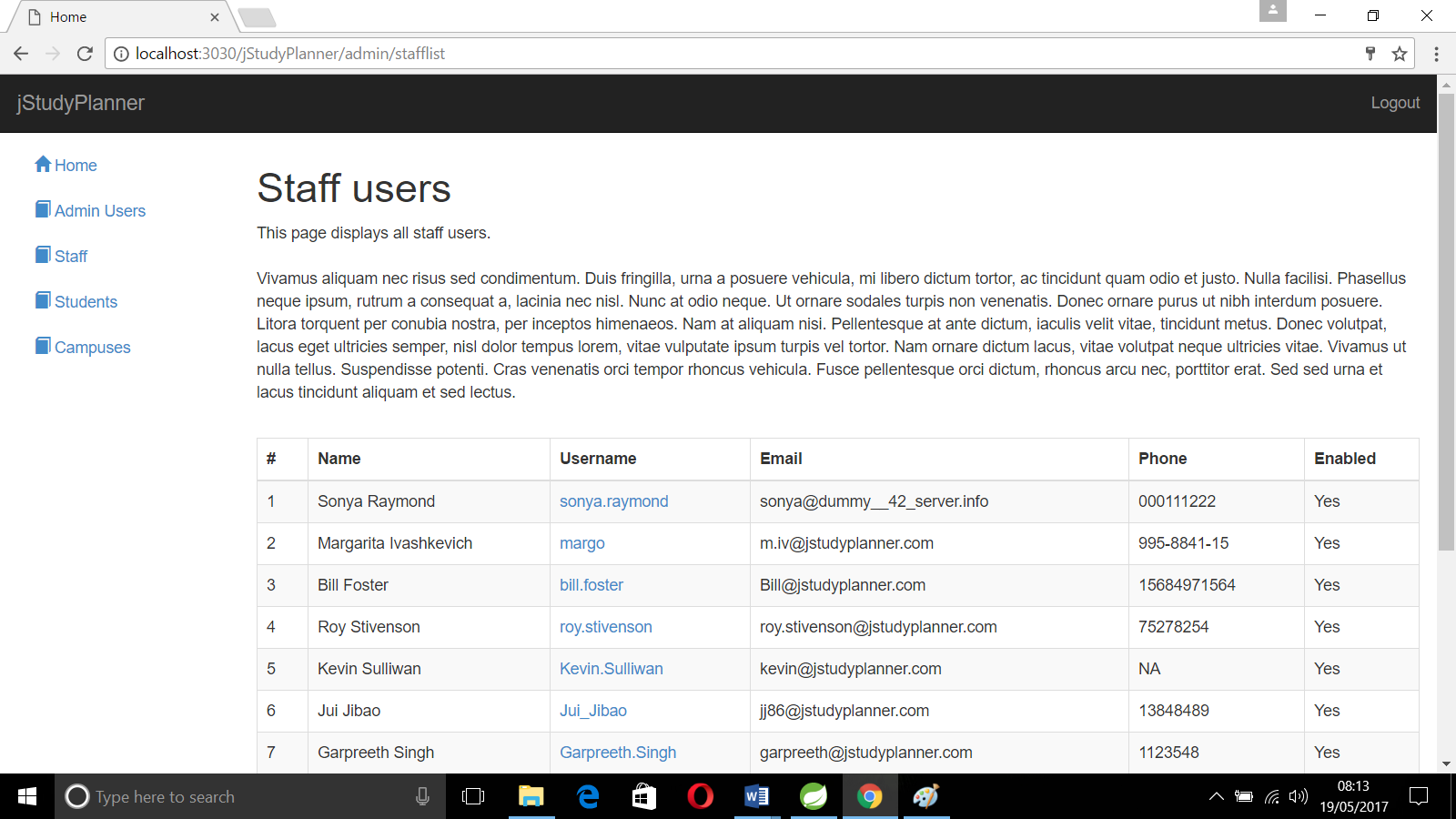
**8.1.3. System Administrator Home Page**

****

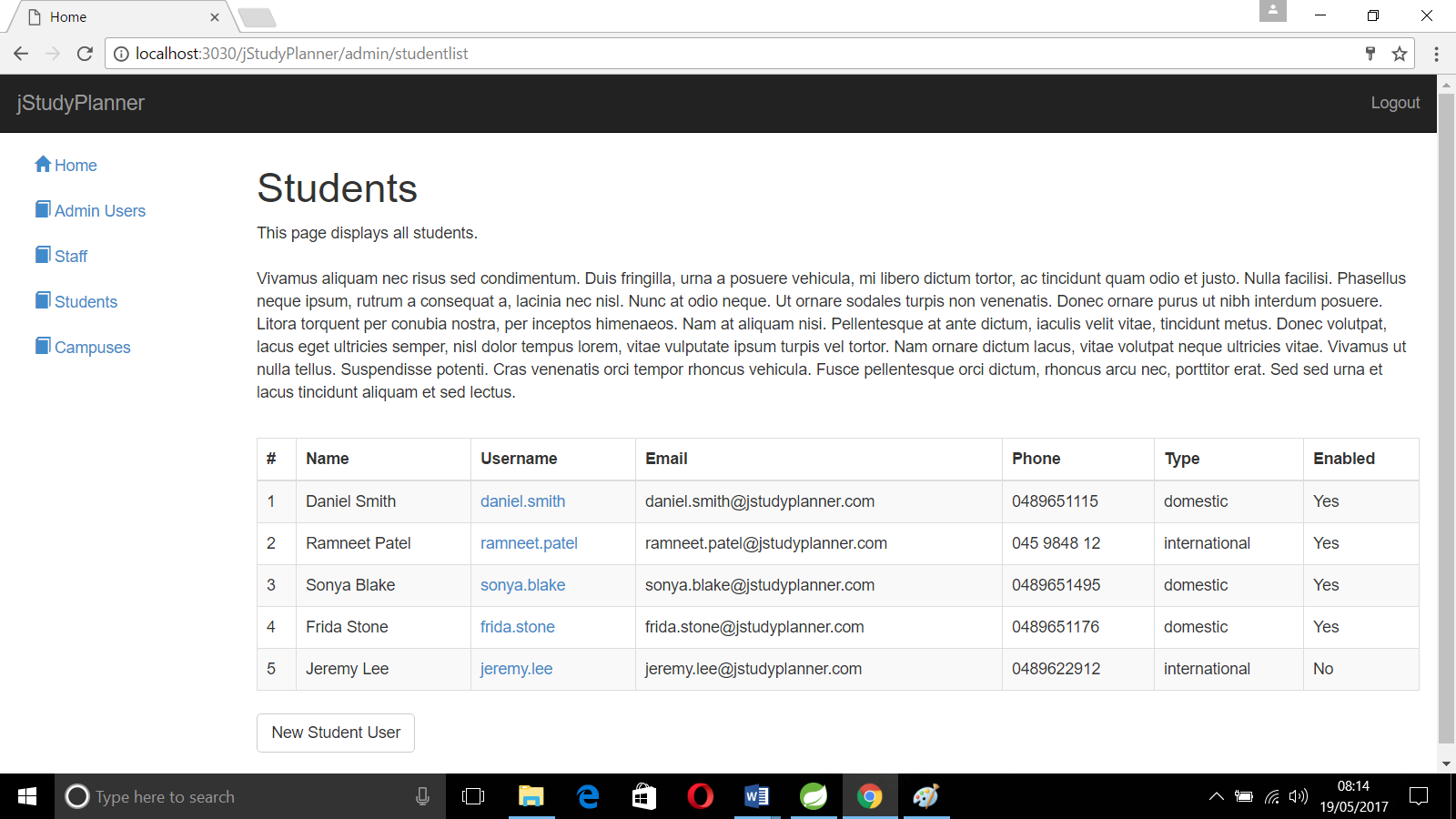
**8.1.4. System Administrator - Admin Page**

****

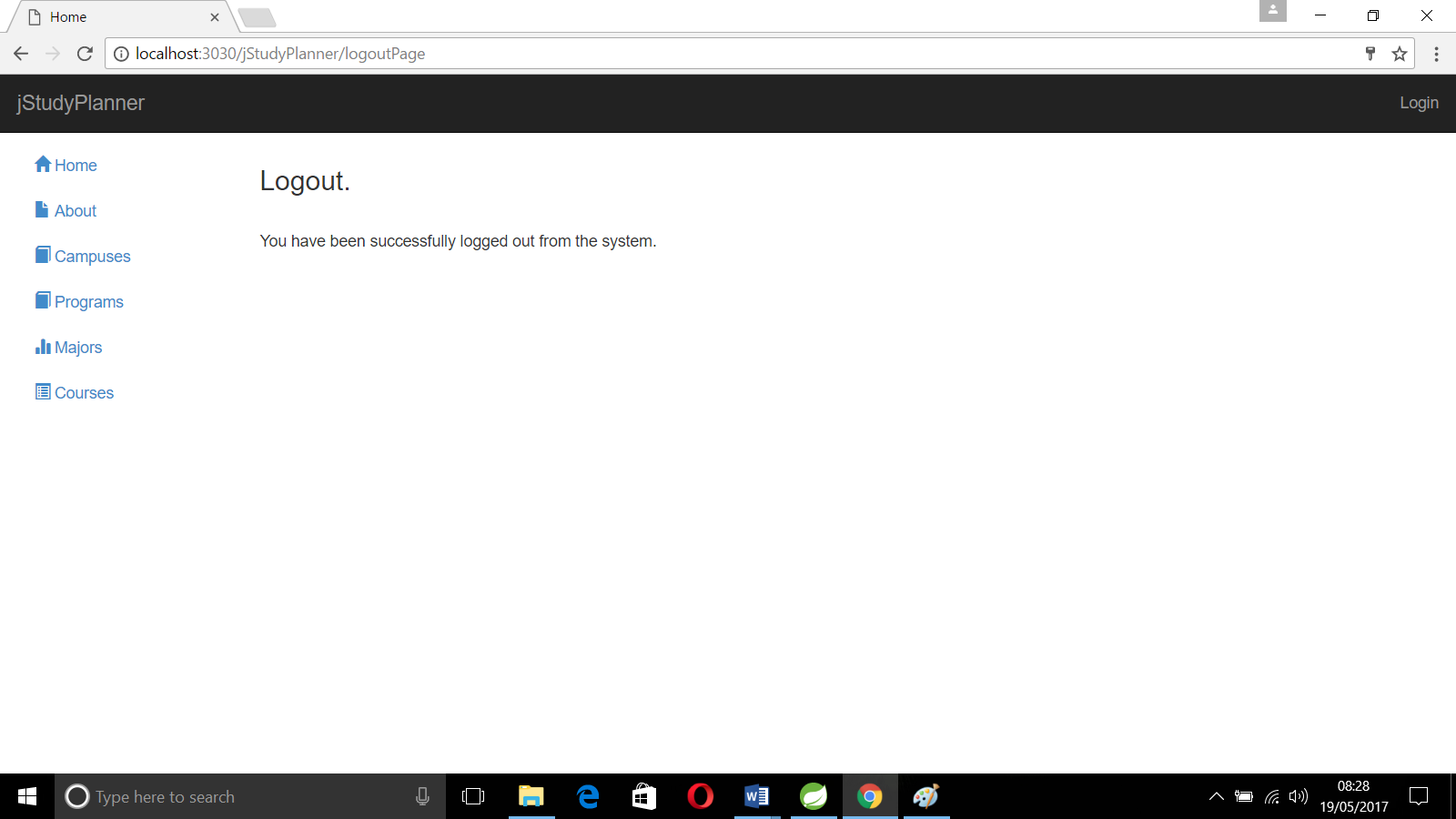
**8.1.5. System Administrator - Staff Page**

****

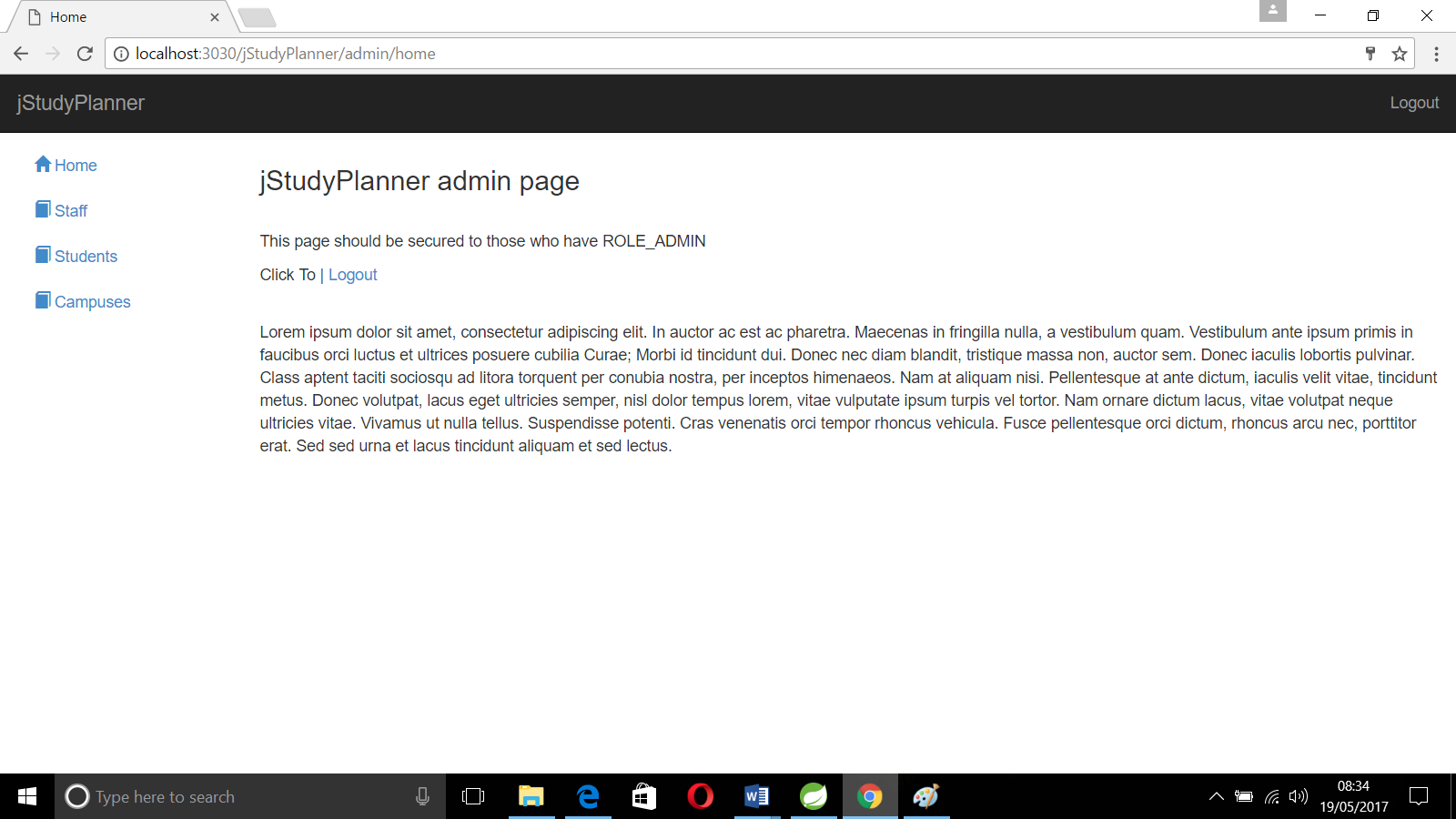
8.1.6. System Administrator – Student Page



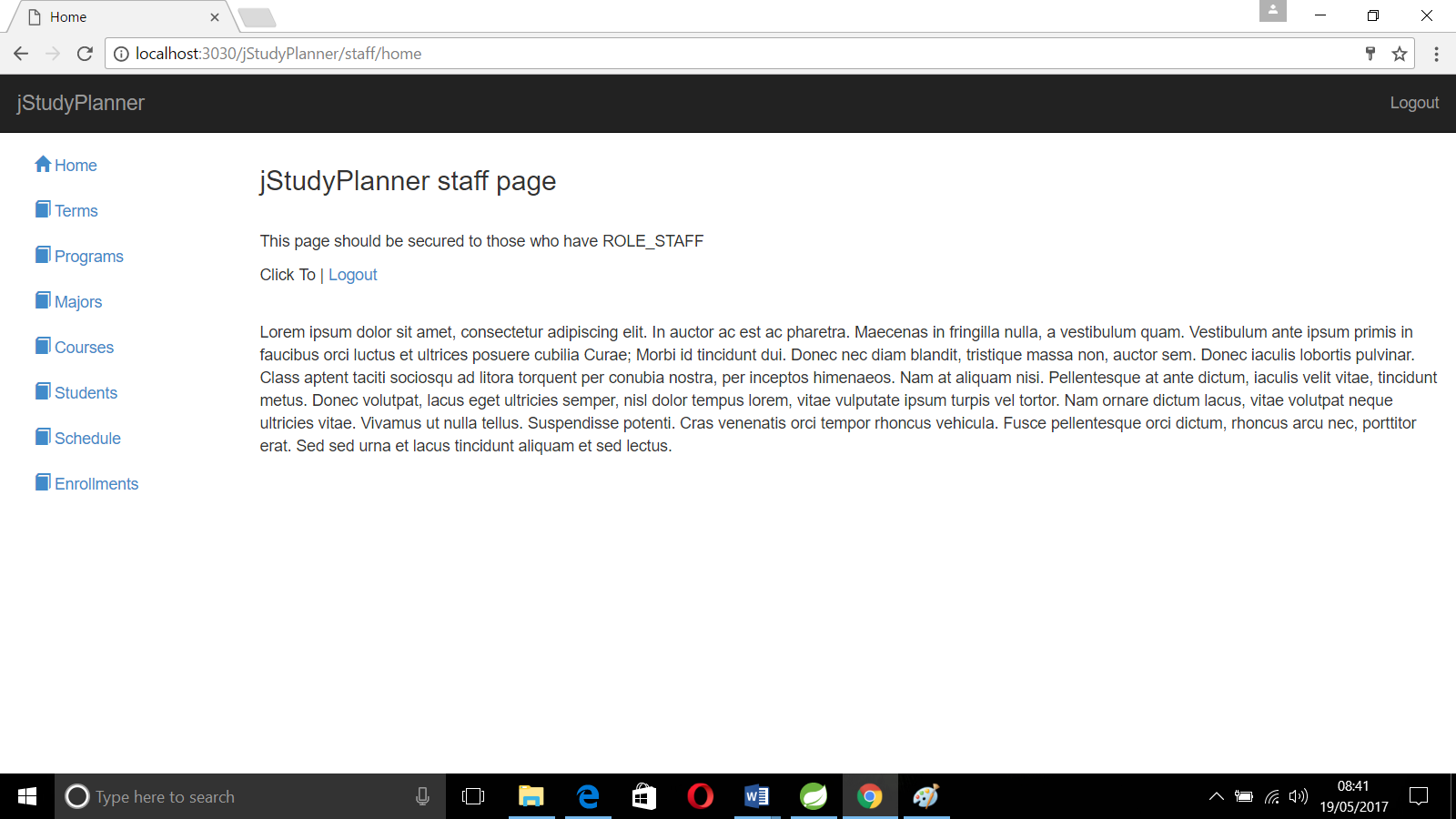
**8.1.7. Logout Page**



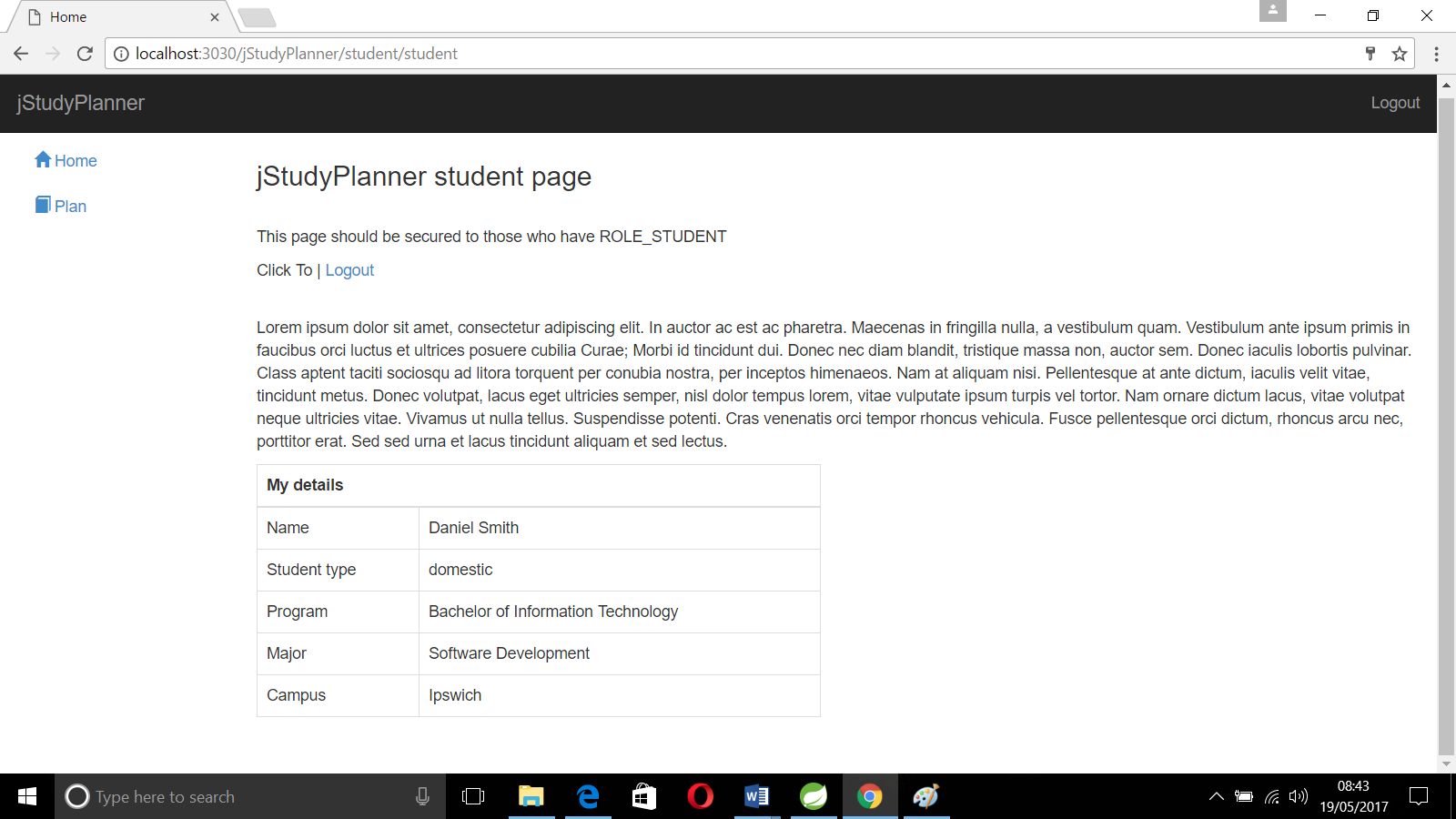
**8.1.8. Application Admin – Home Page**



**8.1.9. Staff – Home Page**



**8.1.10. Student – Home Page**



**9. User/Operational Manual**

**9. User Manual**

The documentation involves collecting, organizing, and maintaining complete record of programs. Documentation is helpful in restarting a project due to some reasons.

Documentation of the software provides the following:

9.1. Comments

Comments are very useful in documenting a program. It is used to explain logic of the program. It should be used to improve the quality and understand ability of the program. It should not be redundant, incorrect or incomplete.

9.2. System Manuals

A good software system must contain standard system manuals. In this the statement is clearly defined, specifies description, detailed flowcharts, and specimen of all input forms and printed outputs.

9.3. Operation Manual

A good software package is supported with a good operation manual to ensure the smooth running of the program.

The operation manual must contain the following information:

* Setup and operational details of each program.
* Loading and unloading procedures.
* Starting, running, and terminating procedures.
* List of error conditions with explanations.
* List of programs to be executed.

The system is completely maintained at any client system to handle all the efficient process that is to be authorized.

**10. Conclusion**

**10. Conclusion**

The “**JStudy Planner”** has been developed to overcome the problem faced in the present system. The developed package was found to work out the operation effectively. The objectives of the system have been achieved.

The goals that have been achieved are:

* Ensure process time and increase throughput.
* Simplifies the operation.
* Avoid some manual work in the existing system.
* Reduce data redundancy and inconsistency.
* User friendly input screens to enter data.

A consistent and efficient system has been successfully developed, implemented and tested. The system has been developed using J2EE and SQL under Windows 10. The system is very flexible and user friendly. So further changes can be incorporated into the system easily. Adequate documentation provides for maintenance and future enhancement.

# 

**11. Bibliography**

# 11. Bibliography

1. System Analysis And Design
   * + Lee
2. An Analysis and Design of Information Systems
   * + James. A. Senn
3. Software Engineering
   * + Roger. S. Pressman
4. Database Management System
   * + William. S. Robert
5. Java Server Pages Programming: A visual professional guide for Design
   * + Robert. D. Nero
6. Java2 complete reference
   * + Herbert Schildt
7. JavaScript
   * + David Flanagan (O’Reilly)
8. Web Resources:
   * + <https://www.spring.io>
     + https:// <http://hibernate.org/orm/documentation/5.2>
     + https://www.w3schools.com