**Functional Requirements:**

**Technologies:**

J2EE: Application Architecture.

Oracle 10g: Database.

My Eclipse 8.6: Development Tool.

Tomcat: Web Server.

Star UML: Design Tool.

**Overview**

SRS will include two sections.

Overall Description will describe major components of the system, interconnection and external interfaces.

Specific Requirements will describe the functions of actors, their role in the system and constraints.

**Product Perspective**

The web pages (XHTML/JSP) are present to provide the user interface on customer client side. Communication between customer and server is provided through HTTP/HTTPs Protocols.

The Client Software is to provide the user interface on system user client side and for this TCP/IP protocols are used.

**System Interfaces**

Client on Intranet:

Client Software, Web Browser, Operating System (any)

Web Server:

Tomcat, Operating System (any)

Data Base Server:

Oracle, Operating System (any)

Development End:

My Eclipse (J2EE, Java, servlets, JSP), Oracle, OS (Windows), Web server.

**Communication Interface**

 Client on Internet will be using HTTP/HTTPS Protocol.

Client on intranet will be using TCP/IP protocol.

**Non Functional Requirements**

**Performance Requirements**

Some Performance requirements identified is listed below:

The database shall be able to accommodate a minimum of 10,000 records of students.

The software shall support use of multiple users at a time. There are no other specific performance requirements that will affect development.

**Security Requirements**

Some of the factors that are identified to protect the software from accidental or malicious access, use, modification, destruction, or disclosure are described below. Specific requirements in this area could include the need to:

Utilize certain cryptographic techniques

Keep specific log or history data sets

Assign certain functions to different modules

Restrict communications between some areas of the program

Check data integrity for critical variables

Later version of the software will incorporate encryption techniques in the user/license authentication process.

The software will include an error tracking log that will help the user understand what error occurred when the application crashed along with suggestions on how to prevent the error from occurring again.

Communication needs to be restricted when the application is validating the user or license.(i.e., using https)

**Portability Requirements**

Some of the attributes of software that relate to the ease of porting the software to other host machines and/or operating systems.

This may include:

Java is used to develop the product. So it is easiest to port the software in any environment.

**Maintainability**

The user will be able to reset all options and all stored user variables to default settings.

**Reliability**

Some of the attributes identified for the reliability is listed below:

All data storage for user variables will be committed to the database at the time of entry.

Data corruption is prevented by applying the possible backup procedures and techniques.

**Usability requirements**

Some of the usability requirements identified for this system are listed below:

A logical interface is essential to an easy to use system, speeding up common tasks.

Error prevention is integral to the system and is provided in a number of formats from sanity hacks to limiting free-text input.

**Availability**

All cached data will be rebuilt during every startup. There is no recovery of user data if it is lost. Default values of system data will be assigned when necessary

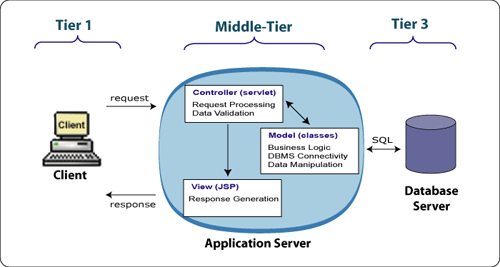
**Software System Attributes**

There are a number of attributes of software that can serve as requirements. It is important that required attributes by specified so that their achievement can be objectively verified. The following items provide a partial list of examples.

The input system will allow for inputting numbers, operands, special symbols and letters of the alphabet.

**PROCESS FLOW**

**ARCHITECTURE DIAGRAM**



1. **THE PRESENTATION LAYER**

Also called as the client layer comprises of components that are dedicated to presenting the data to the user. For example: Windows/Web Forms and buttons, edit boxes, Text boxes, labels, grids, etc.

1. **THE BUSINESS RULES LAYER**

This layer encapsulates the Business rules or the business logic of the encapsulations. To have a separate layer for business logic is of a great advantage. This is because any changes in Business Rules can be easily handled in this layer. As long as the interface between the layers remains the same, any changes to the functionality/processing logic in this layer can be made without impacting the others. A lot of client-server apps failed to implement successfully as changing the business logic was a painful process

1. **THE DATA ACCESS LAYER**

This layer comprises of components that help in accessing the Database. If used in the right way, this layer provides a level of abstraction for the database structures. Simply put changes made to the database, tables, etc do not affect the rest of the application because of the Data Access layer. The different application layers send the data requests to this layer and receive the response from this layer.

1. **THE DATABASE LAYER**

This layer comprises of the Database Components such as DB Files, Tables, Views, etc. The Actual database could be created using SQL Server, Oracle, Flat files, etc.   
In an n-tier application, the entire application can be implemented in such a way that it is independent of the actual Database. For instance, you could change the Database Location with minimal changes to Data Access Layer. The rest of the Application should remain unaffected.