**Online Examination System**

Description:

* This system is used for giving exam online by the student for his registered courses. It also maintain student’s exam history and from that score will be populated.
* Any educational institute or any other department where exam is mandatory to perform further actions are the end users of this application.
* This system will give chance to student to appear for exam again if his previous score is not satisfactory.
* Major functionalities of this application:
  + User can login into the system by correct Username and Password
  + Insert and update Student
  + Get distinct courses
  + Retrieve random questions for exam for selected course
  + Display score of exam and allow user to give retry for exam

System startup URL:

<http://localhost:8080/OnlineExam/>

Username: oesdbuser

Password: password

Technology:

Model: Java classes, Web Service, JSON

View: HTML5, JSP, JSTL, JQuery, CSS

Controller: Servlet

Database: MySQL

Server: Tomcat 8.0

MVC Architecture

Client Browser

**Web Client**

**Web Server (Tomcat 8.0)**

**Views**

JSP Pages

**Model**

**(Business Logic and data representation)**

Service Classes

Web

Service

Domain Classes

**Data Access Layer**

DAO Classes

Factory Classes

**Controller**

Servlet

Exception

Handler Classes

**Database Server (MySQL)**

Database

MVC Architecture Description:

**Controller (Servlet):**

RegistrationServlet

* It contains doGet and doPost method to handle request.
* This servlet class handle student add and update operations.
* In doGet method, it retrieve parameter named “Service”. This parameter specifies whether incoming request is adding student or updating student detail. Based on value of “service” parameter, it set redirectURL variable to specify JSP which need to forward.
* Set require objects and attributes in request objects and forward response to JSP using response dispatcher.
* In doPost method, retrieve parameter named “service”. This parameter specified whether incoming request is add, update or delete employee. Based on value of “service” parameter, it

call service layer classes (Model-StudentService) to perform business logic.

GetQuestionServlet

* It contains doGet method to handle request.
* This servlet class handle question list and update exam score rows.
* By using this servlet question paper will be available to user
* If there are any rows in studentanswer table that will be updated with flag 0 before getting question paper. So this will affect multiple rows in table.
* It also calls web service to retrieve question list in JSON format. Parse JSON to arraylist and set to request attribute. This attribute is requested by JSP page to display questions in label and radio button.

**View (JSP/HTML):**

* It represent data using JSP, JSTL and HTML tag.
* Use Jquery to perform client side scripting and designing UI.
* All .jsp files are hidden in WEB-INF folder expect LoginForm.jsp
* JSP forwarded by controller.

**Service Layer:**

Package: npu.edu.exam.services

Classes: QuestionServices, StudentExamServices, StudentServices

* Called by servlet controllers
* Perform business logic
* It call DAO classes method to retrieve data from database
* Exception handling
* Also call other service classes if require
* Return domain objects to controller

**Service Layer:**

Example (StudentService):

* It contains various methods to accomplish business logic. Mainly contains methods for performing CRUD operations for Employee Data.
* Methods called by controller.
* Each method, call DAO class methods to retrieve requested data from database.
* Throws exception to controller in case of failure.
* Return domain class object which contains requested data to controller.

**DAO Layer:**

Example ( StudentDAO):

* It contains methods to perform database operation.
* Create database connection to perform db operation, and close once done. It rollback transaction, it db operation fail and commit in case of success.
* Called by service layer classes.
* Uses factory classes to convert database result to domain object.
* Return domain objects to service layer.
* Throws exception to service layer in case of failure and rollback transaction.

StudentFactory.java

* Convert database result into database object.

**WebService:**

Package: npu.edu.exam.servlets

Class: GetJsonQuestionServlet

* It returns question list depending on selected course in JSON format
* Call by JSON reader class
  + Package: npu.edu.exam.webservicereader
  + Class: JsonQuestionListReader

Transaction Implementation (modify multiple rows):

Class: StudentServices.java

Method: RetryExam (StudentExam exam)

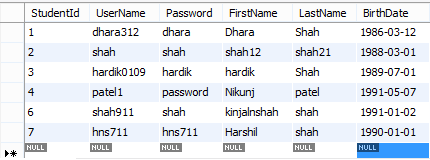
* This method is called when user choose to take a retry for exam for specific course.
* At the time of user’s first attempt rows are inserted in studentanswer table with data consist of coursename, question id, user’s answer and recent flag sets with ‘1’.
* When user wants to take retry at that time recent flag will be modified with ‘0’ for the rows which were entered previously. Thus that data will considered as his previous exam and new rows will be entered with recent flag ‘1’.

Database Table:

* **Student**: This table will store student related information

|  |  |  |  |
| --- | --- | --- | --- |
| **student** | | | |
| **Field** | **datatype** | **constraint** | **reference** |
| StudentId | int | NN, PK, AI |  |
| Username | varchar | NN,UQ |  |
| Password | varchar | NN |  |
| FirstName | varchar | NN |  |
| LastName | varchar | NN |  |
| BirthDate | date | NN |  |

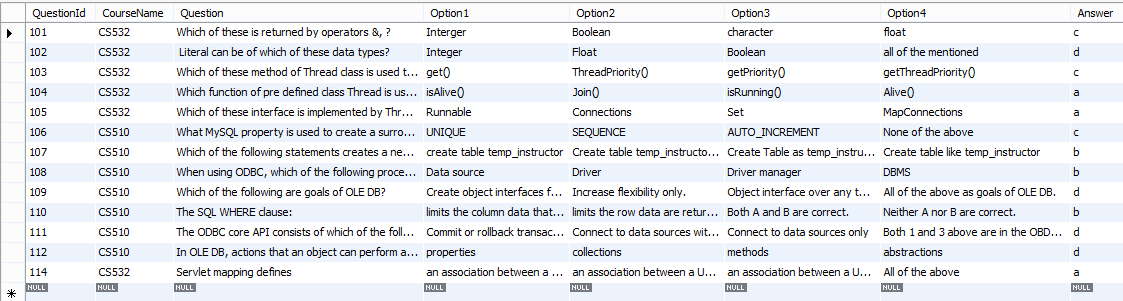
Sample Data:



* **Question**: This table will store question related information

|  |  |  |  |
| --- | --- | --- | --- |
| **student** | | | |
| **Field** | **datatype** | **constraint** | **reference** |
| QuestionId | int | NN, PK, AI |  |
| CourseName | varchar | NN |  |
| Question | varchar | NN |  |
| Option1 | varchar | NN |  |
| Option2 | varchar |  |  |
| Option3 | varchar | NN |  |
| Option4 | varchar |  |  |
| Answer | varchar | NN |  |

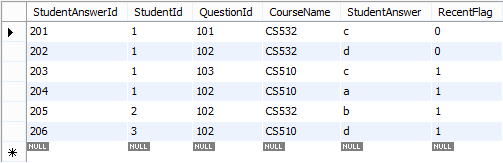
Sample Data:



* StudentAnswer: This table will store student’s exam information.

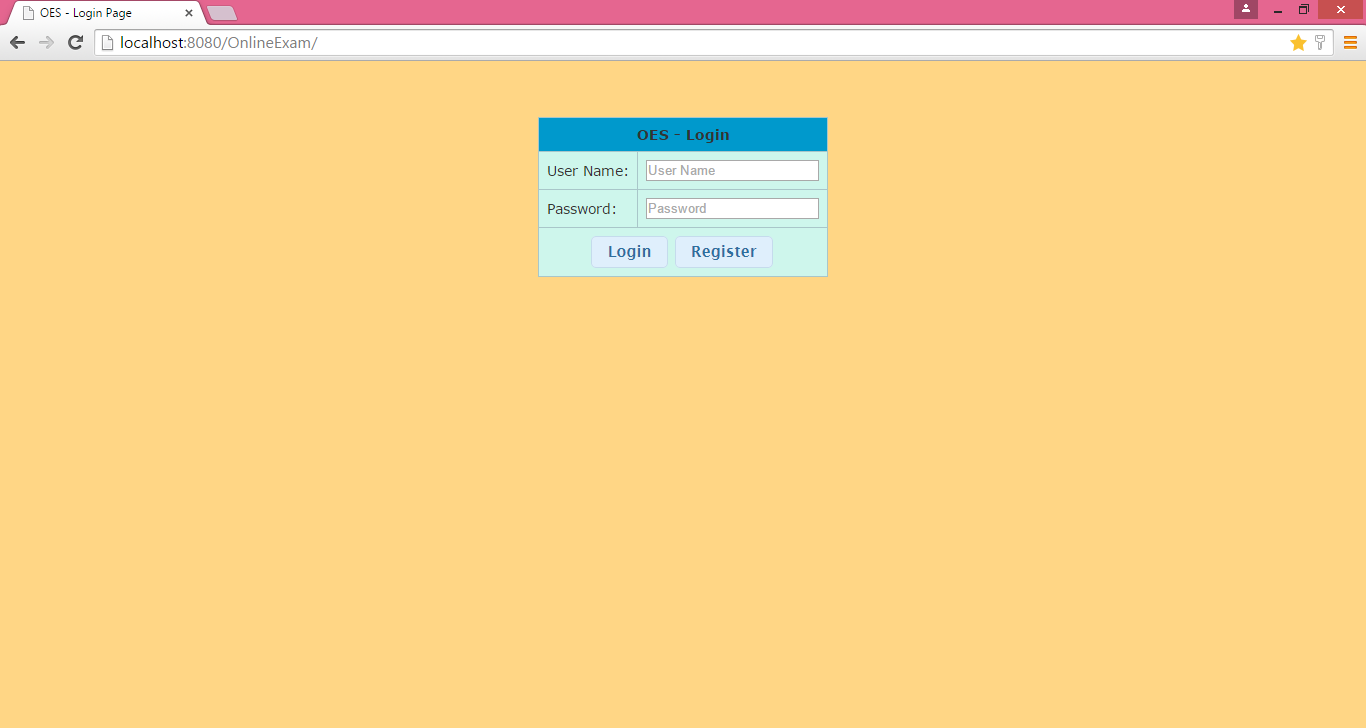
|  |  |  |  |
| --- | --- | --- | --- |
| **student** | | | |
| **Field** | **datatype** | **constraint** | **reference** |
| StudentAnswerId | int | NN, PK, AI |  |
| StudentId | varchar | NN,FK | Foreign key to student table |
| QuestionId | varchar | NN,FK | Foreign key to question table |
| CourseName | varchar | NN |  |
| StudentAnswer | varchar | NN |  |
| RecentFlag | tinyint | NN | Default 0 else 1 |

Sample Data:

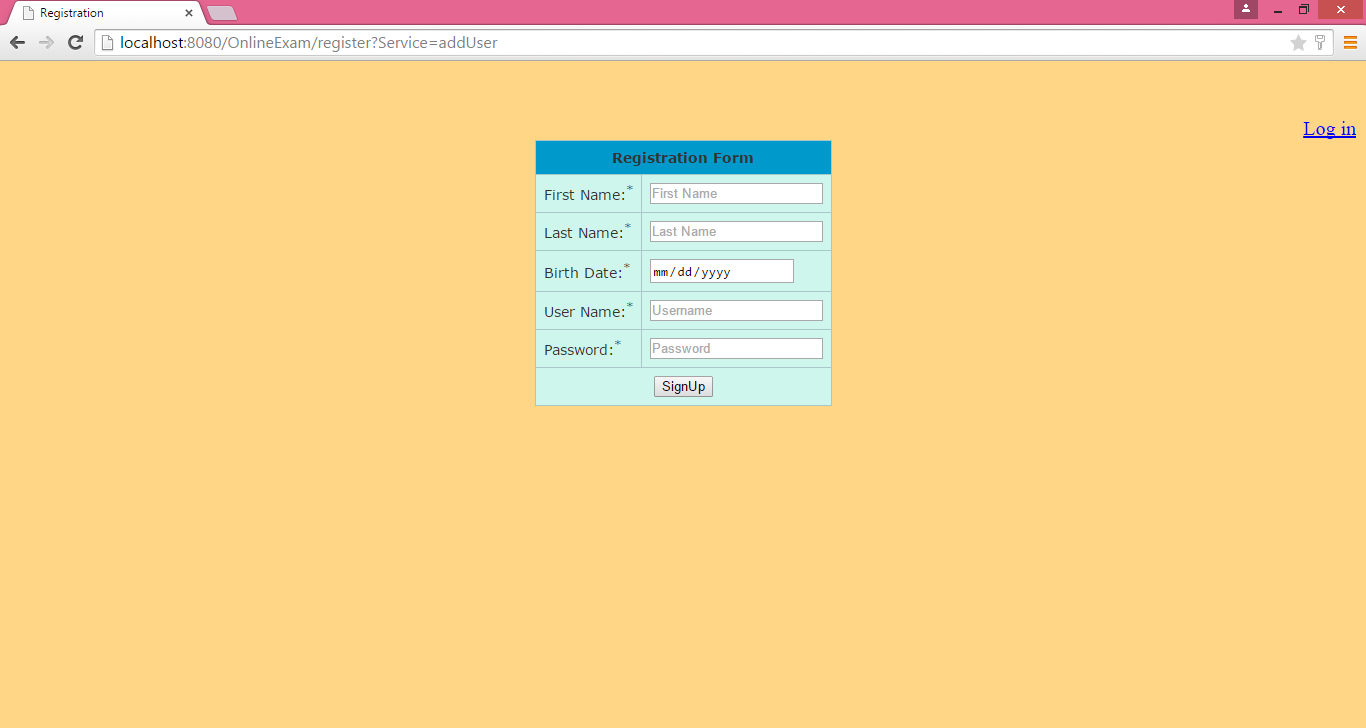


Screenshot:

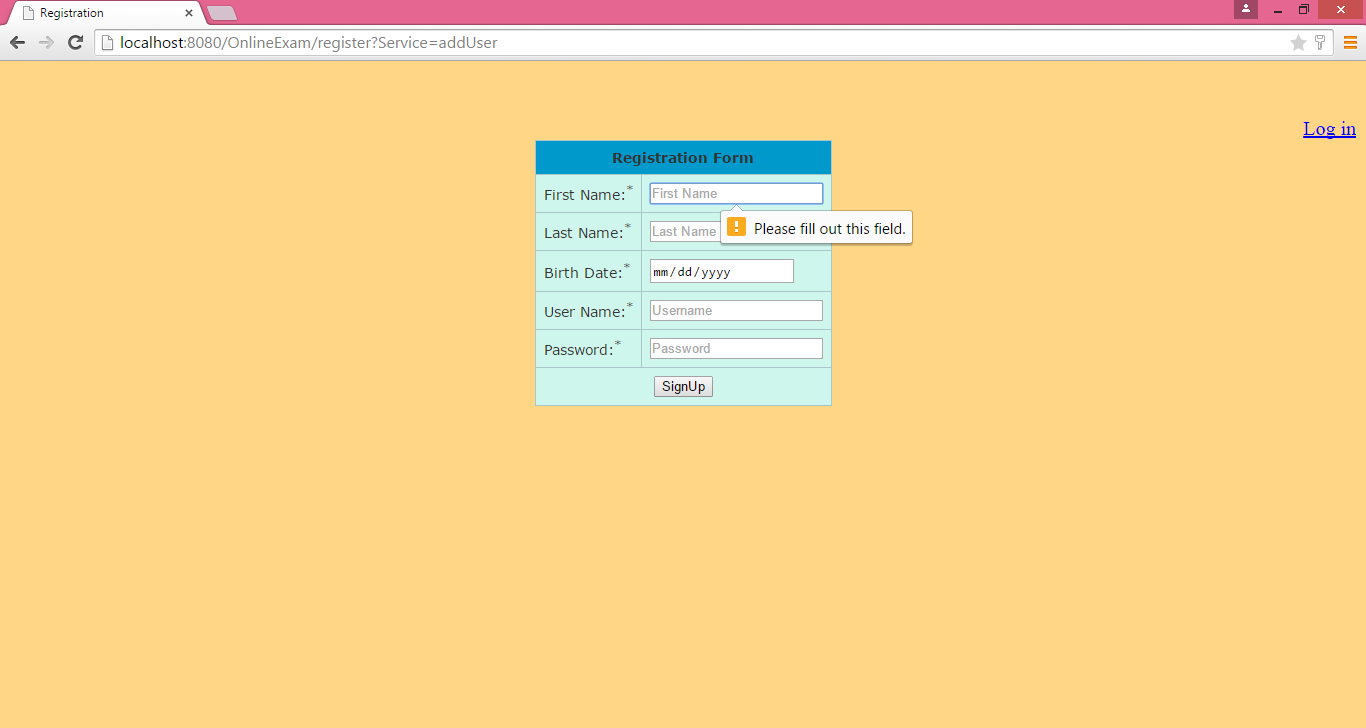
* Login page



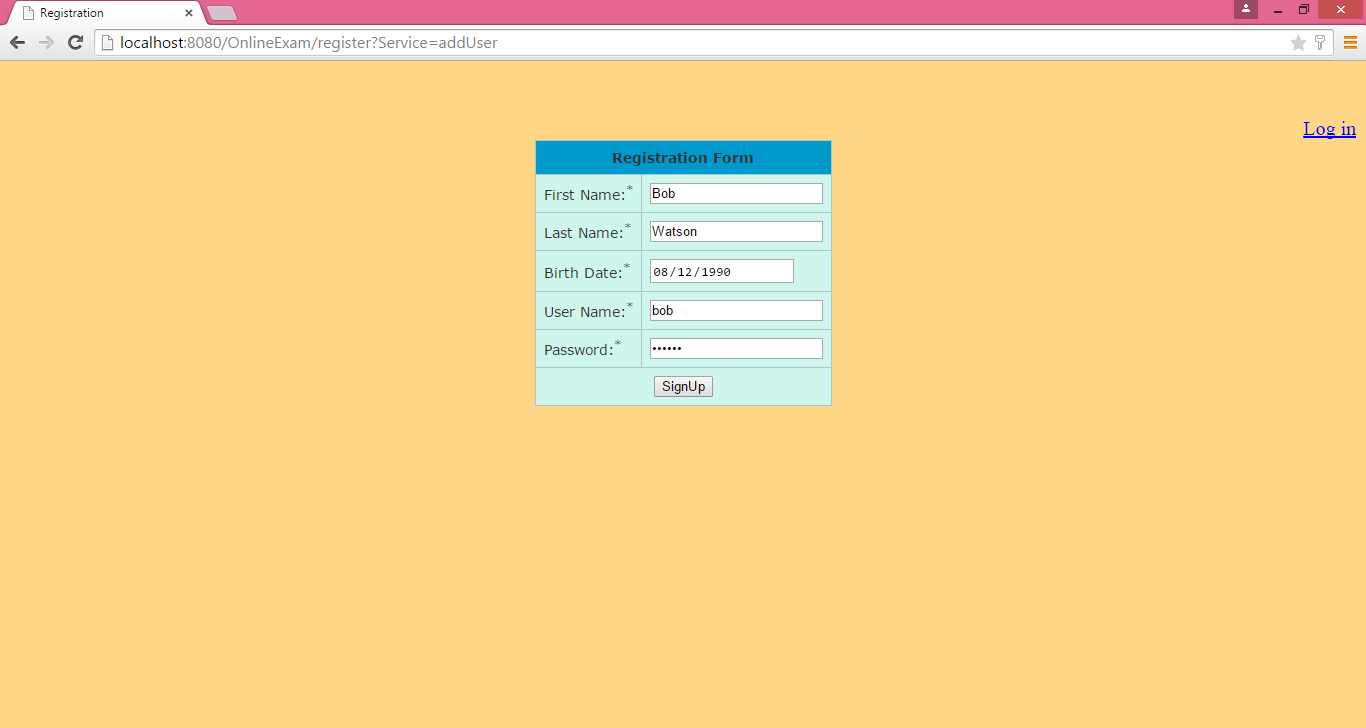
* Registration page



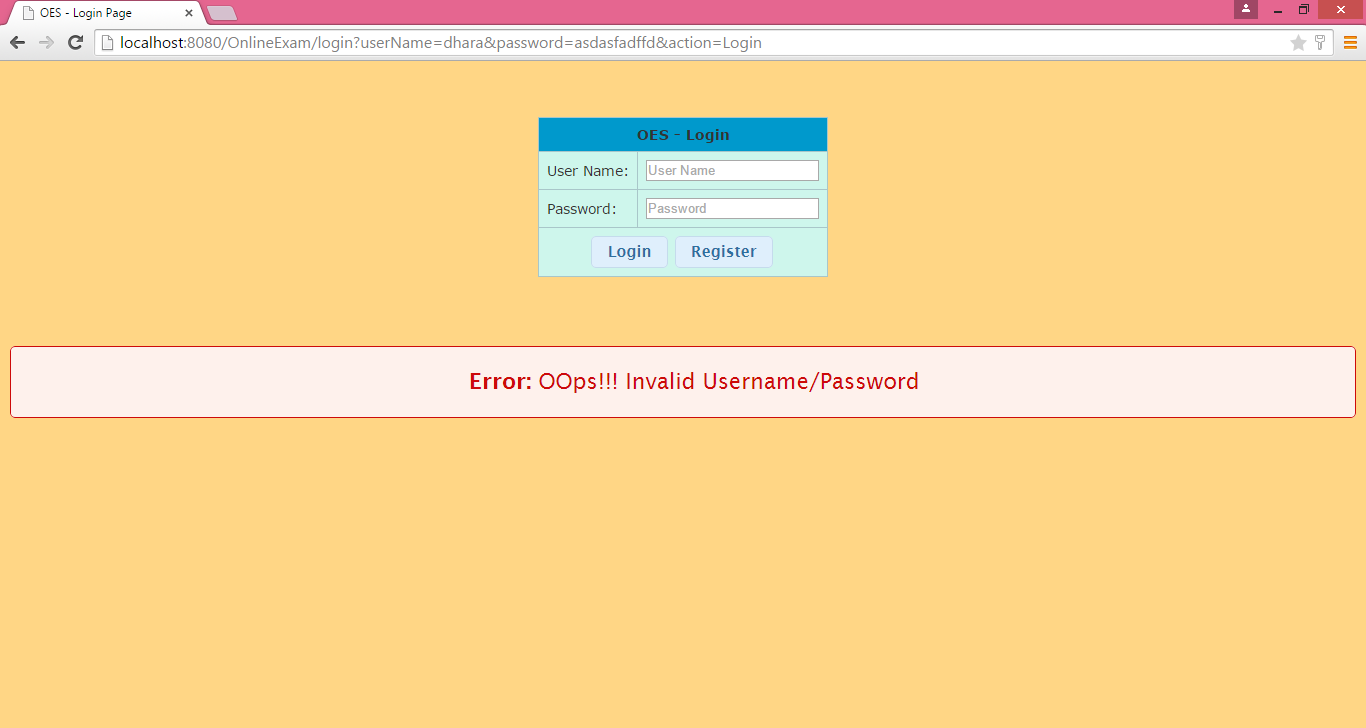
* Registration page validation



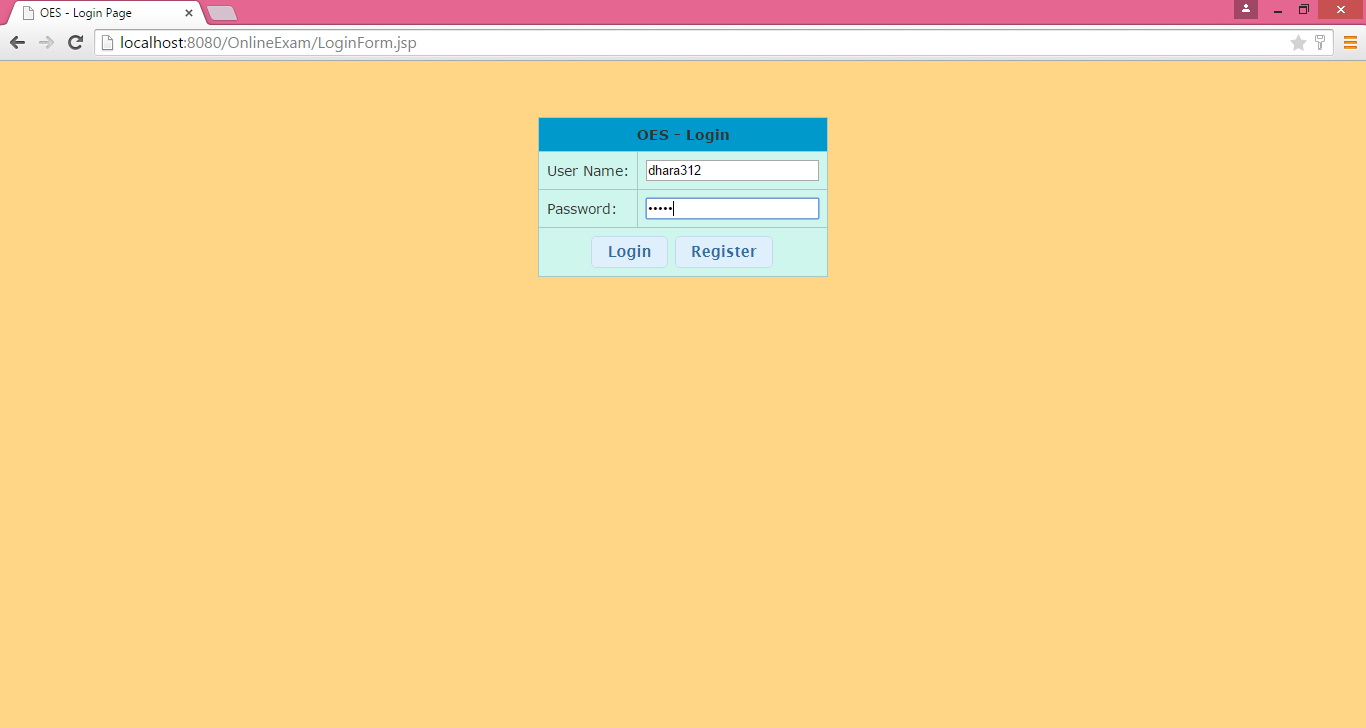
* Registration Page (with Valid Data)



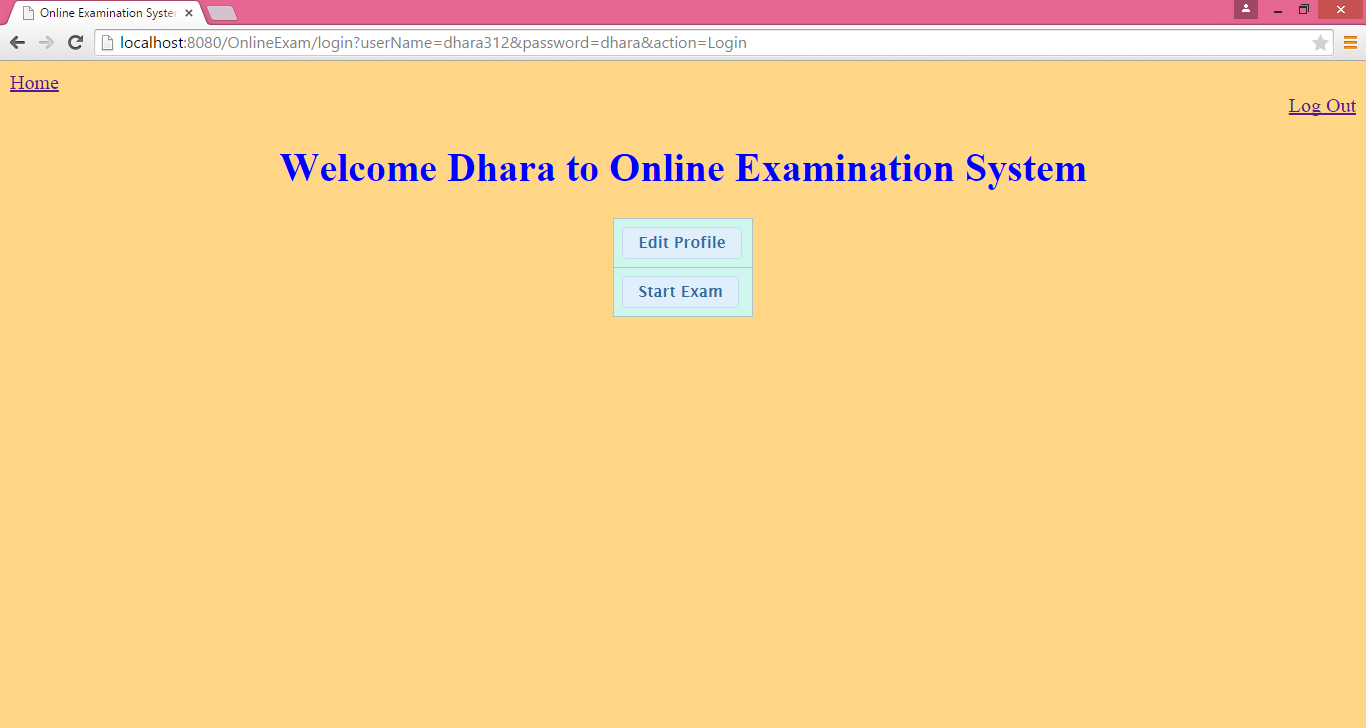
* Login page validation



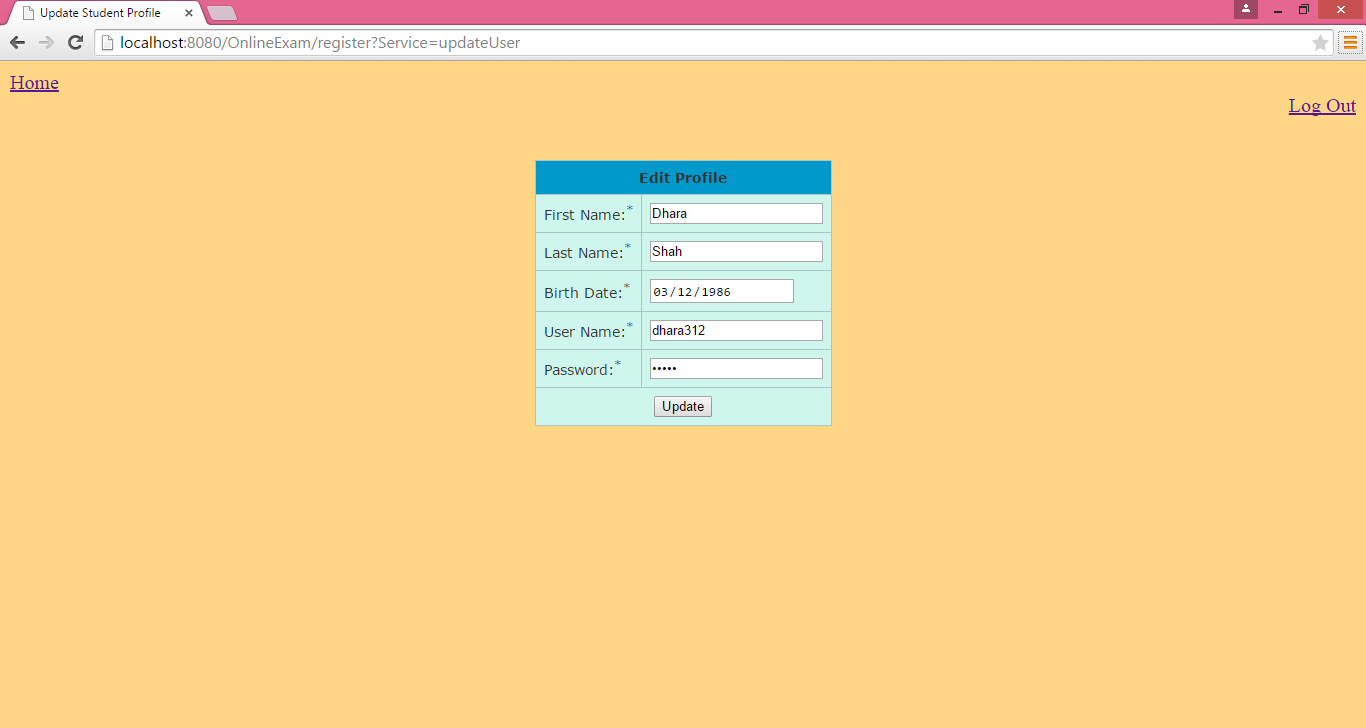
* Login page (with valid data)



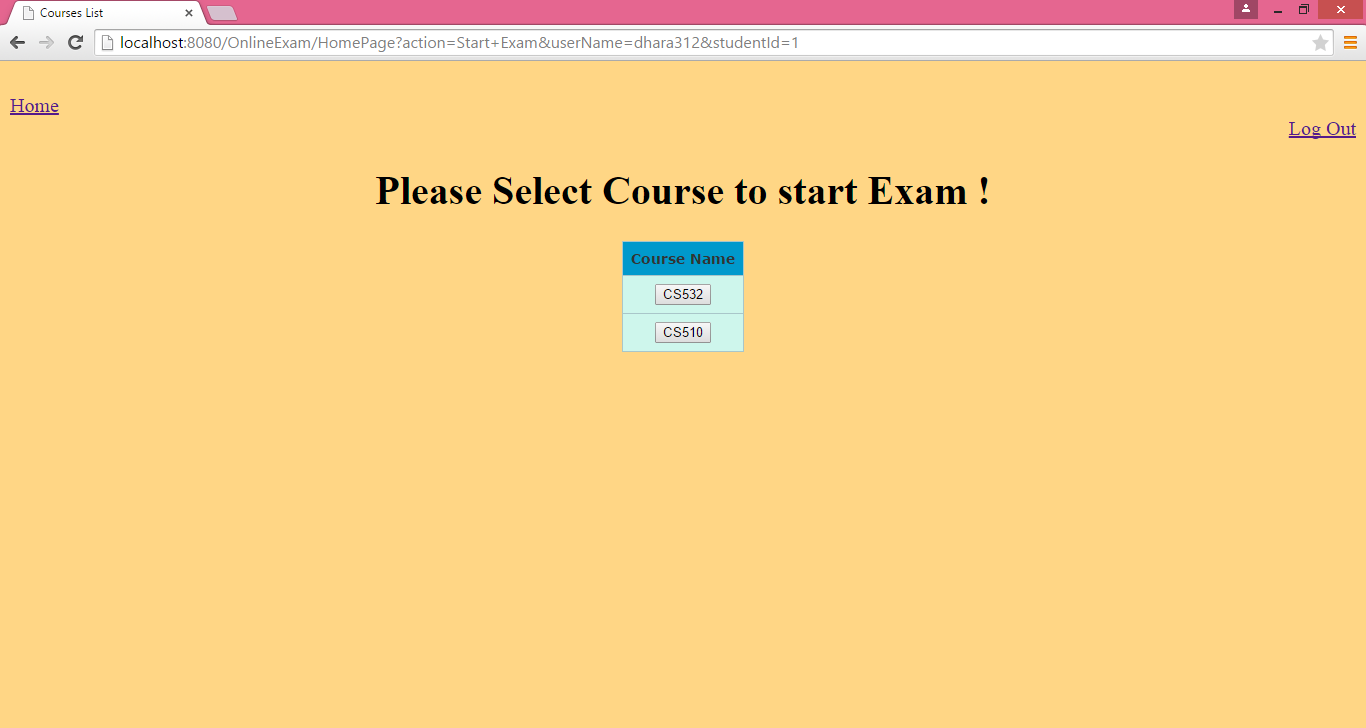
* Home page



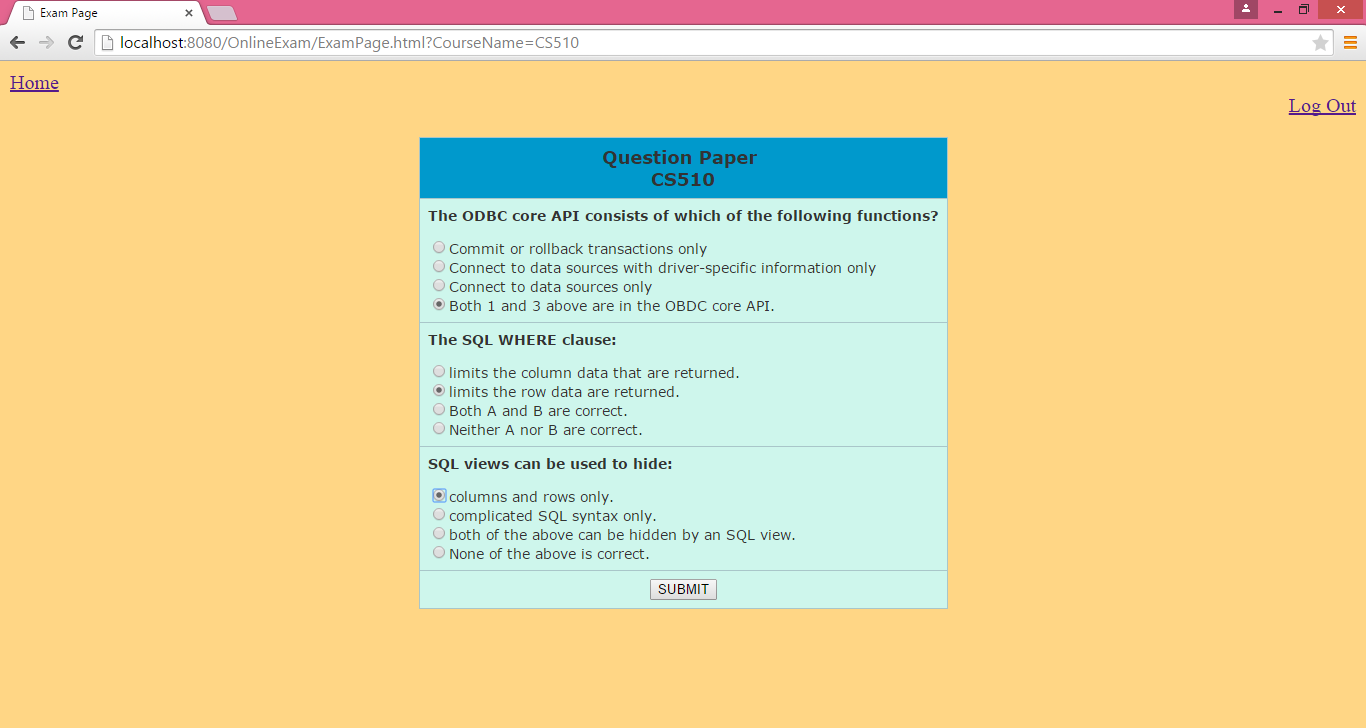
* Edit Profile



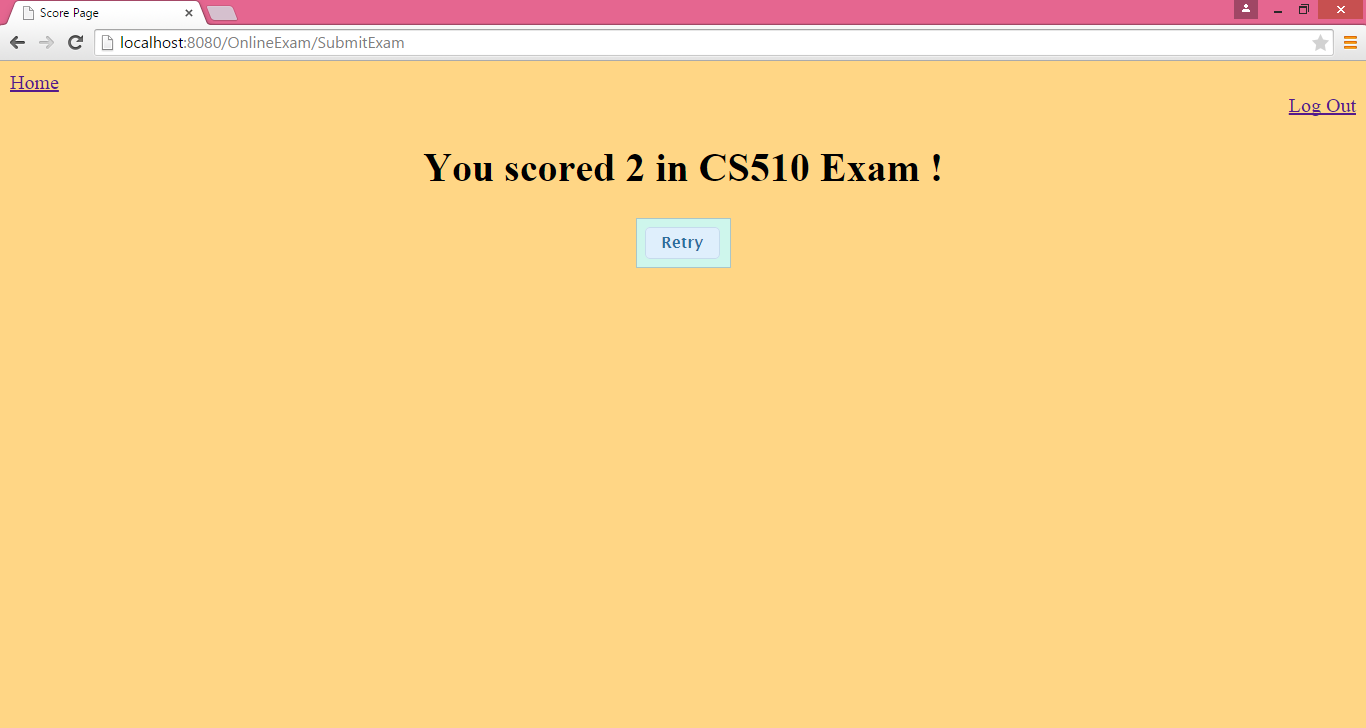
* Course page



* Exam page – display random questions for selected course.



* Score page – here if user clicks on retry button, a new question paper will be generated and he will allow to give exam again.



* Logout – redirects back to login page

