

***Java Technologies for Web Applications***

**Project Assignment**

|  |  |
| --- | --- |
| Document Code | 25e-BM/HR/HDCV/FSOFT |
| Version | 1.1 |
| Effective Date | 20/11/2012 |

**Hanoi, 04/2019**

RECORD OF CHANGES

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Effective Date | Change Description | Reason | Reviewer | Approver |
|  | 25/Jun/2018 | Create a new assignment | Create new | DieuNT1 | VinhNV |
|  | 01/Jun/2019 | Apply fsoft template  Update requirements: split Specification1 and 2 by training day | Update | DieuNT1 | VinhNV |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Contents

[Project Assignment 4](#_Toc24110512)

[Objectives 4](#_Toc24110513)

[Business needs 4](#_Toc24110514)

[Product architectures 5](#_Toc24110515)

[Working requirements 5](#_Toc24110516)

[Technical Requirements 5](#_Toc24110517)

[Database Relationship 6](#_Toc24110518)

[Specifications 1 6](#_Toc24110519)

[Specifications 2 9](#_Toc24110520)

[Mark scale : 10](#_Toc24110521)

|  |  |
| --- | --- |
|  | **CODE: JWEB.P.A102 (Polls)**  **TYPE: Project Assignment**  **LOC: N/A**  **DURATION: 12 hours (completed in 3 working day)** |

# Project Assignment

Objectives

* Understand the basic concepts of web development technologies with java (JSP / Servlet)
* Able to write servlets using the Java programming language (Java servlets)
* Create dynamic HTML content with Servlets and JavaServer Pages, using the Expression Language, and the JSP Standard Tag Library (JSTL)
* Create robust web applications using MVC architecture, session management, filters, and database integration (JDBC) or Hibernate framework
* Make Servlets and JSP work together cleanly
* Create secure web applications using the features of the Java EE web container
* Deploy your project using Apache Tomcat Server

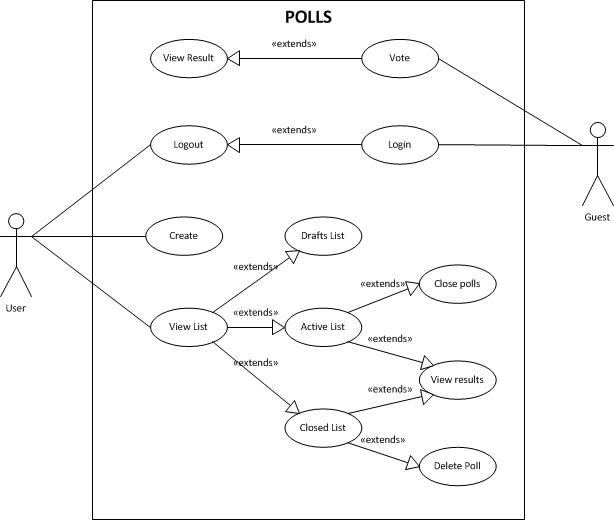
Business needs

Create a Web application based on HTML, CSS, JavaScript, Ajax, JSP, Servlet, JDBC and Hibernate which is a basic poll application.

This application consists of two parts:

* A public site that lets people view polls and vote in them.
* An admin site that lets you add, edit and delete polls.

You can make a poll with a survey app, a survey with a form app, and a simple survey with a poll app. The use case Diagram as below here:



A survey is a form with questions specifically designed to gather information about people's experiences, preferences, wants, and needs. Surveys can be long, so a poll is a survey with just one question—a simple survey.

In this assignment, we will create simple polls - survey web application based on HTML, CSS, JavaScript, Ajax, Servlet, JSP, JDBC, Hibernate framework.

Product architectures

If you use JDBC to connect to Database, then must use the following architecture:



The architecture with Hibernate framework:

**Model/Value objects**

*(represent things in our application domain)*

**Data access layer**

*(handle saving and retrieving data as objects)*

**Presentation Layer**

*(This layer controls the display to the end user)*

HibernateUtils

**Database**

Client tier

Working requirements

Working environment:

* Eclipse IDE,
* SQL DB Server,
* Apache TomCat 8 or later,
* Internet connection,
* Apache Maven.

Delivery:

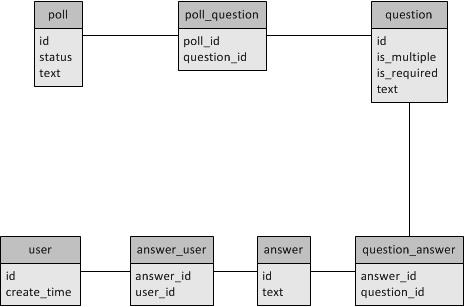
* Source code, deployment and testing evident packaged in a compress archive.

Technical Requirements

* Must use HTML/CSS, JS/jQuery, JSP/Servlet and JDBC or Hibnerate Framework

Database Relationship

The following entity-relation (ER) diagram shows the table structure and relationships about the above tables



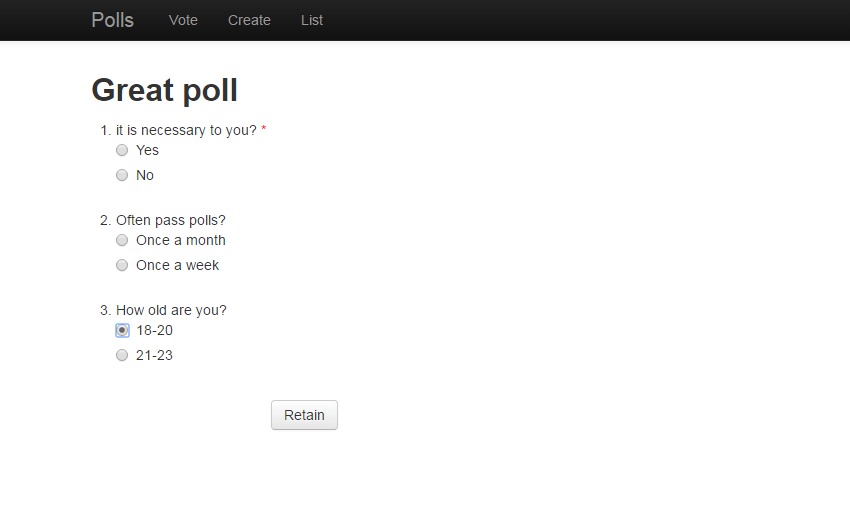
* ***question*** stores the question. The table mainly contains the following columns:
  + is\_multiple: option, the multiple answer for this question;
  + is\_required: option, choose for show this question in the polls.
* ***poll*** stores the polls information, including status of the poll and description of poll.
* ***poll***\_***question*** stores the map between the poll and question. The table mainly contains the following columns:
  + poll\_id: required, it map with a poll.
  + question\_id: required, it map with a question.
* ***answer*** stores the answer information, including description for the answer.
* ***question\_answer*** stores the map between the answer and question. The table mainly contains the following columns:
  + answer\_id: required, it map with an answer.
  + question\_id: required, it map with a question.
* ***user*** stores the user IP address information, including IP address and the time created the vote.
* ***answer\_user*** stores the map between the answer and user. The table mainly contains the following columns:
  + answer\_id: required, it map with an answer.
  + user\_id: required, it map with a user.

Specifications 1

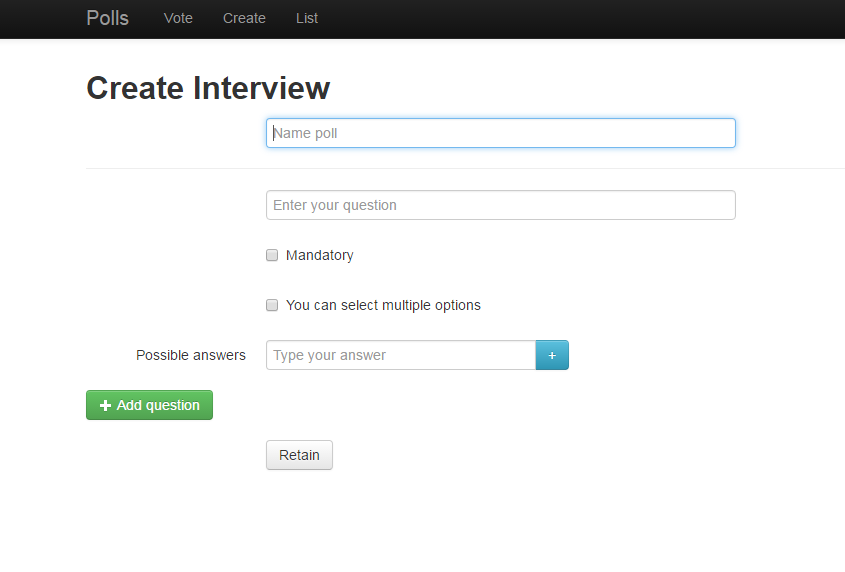
1. **Create screens**

You have to create the 3 .jsp pages for a survey app, the design for each page is as below:

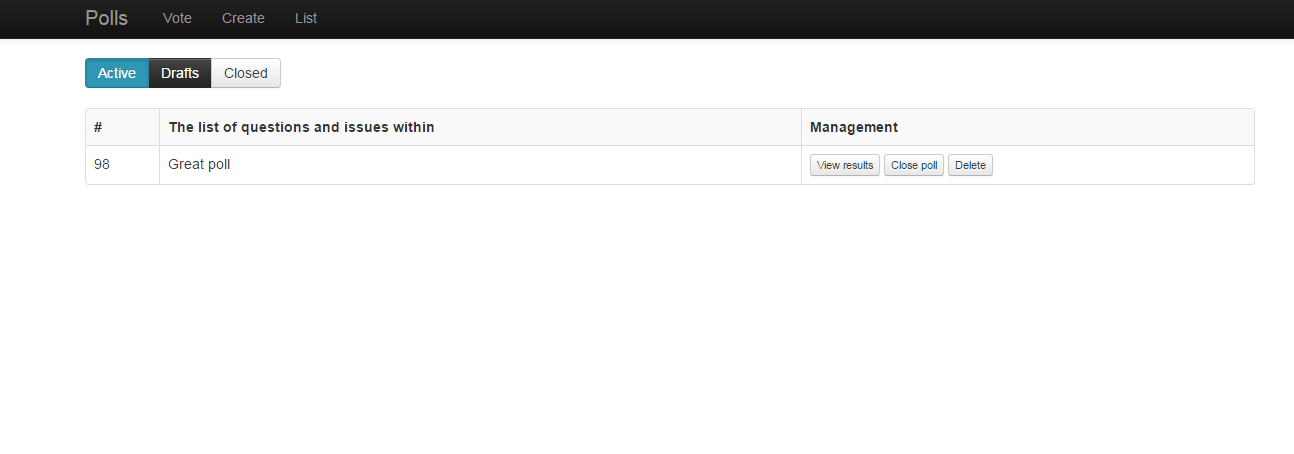
* *Home page*



* *Create Page*

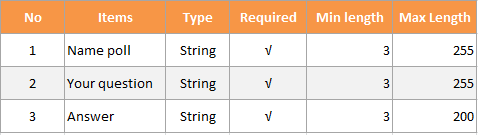


* *List page*



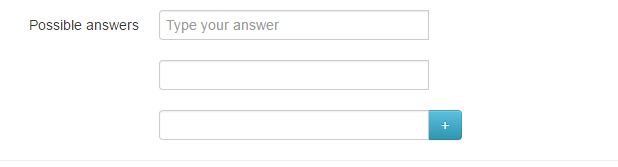
1. **Validates**

You must use jQuery (or Javascript) to validate data in the create interview page, the item definition as below:

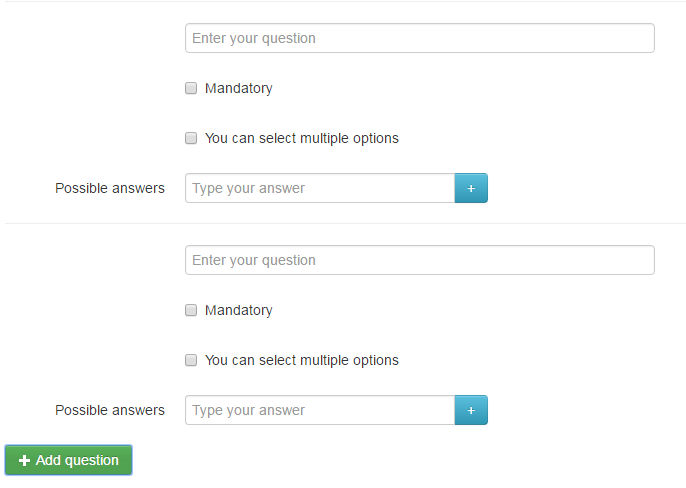


Use Javascript or jQuery to add a new question and add new answer. The design for this feature as the photo below:

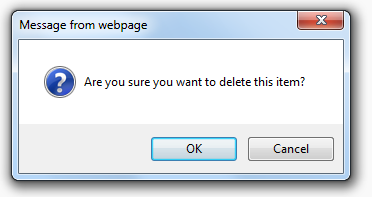
* *Add a new answer*



* *Add a new question*



Use Ajax in the manage question page. If user click to delete button, its will show the confirm box to make sure user want to delete this item.



After that, if it is confirm ok, you must send data to server to delete this item via Ajax.

If click into login button at right menu, it will show a popup, it show form to login user.

|  |
| --- |
|  |

The form must be checked to make sure all the mandatory fields are filled in the input tag. It would require just a loop through each field in the form and check for data.

If validation fails, the system should let the users know it by providing a clear and unambiguous message (usually one or two sentences) and ways to correct errors. Since users need to notice an error message immediately, it is a good practice to position it at the top of a web form, before all the other fields. This will also allow screen readers to easily access the message.

Apart from the error message and a list of invalid fields, the system should clearly mark fields that are invalid. This can be done in one of the following ways (or any combination of them):

* By providing red inline messages or markers next to every invalid field
* By changing the color of field labels

Rules of thumbs in web form validation design

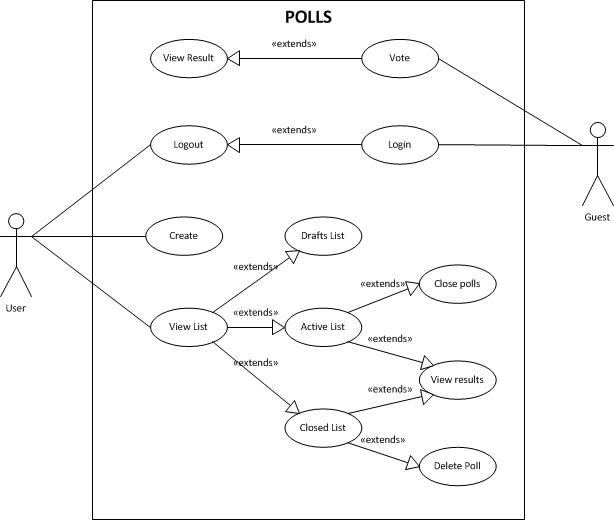
* Never omit server-side validation.
* Don’t provide confusing validation feedback. It should clearly communicate the errors and ways to fix them.
* Don’t let users think about what information is required, always clearly mark required fields.
* Never provide validation feedback on a single page or in a popup alert.
* Don’t use dynamic effects as compensation for a badly designed form. Fancy effects won’t hide a poorly designed web form.
* Don’t forget to inform users when the form was completed successfully. It is as important as a good validation feedback.

Use Ajax to get data from a HTML file

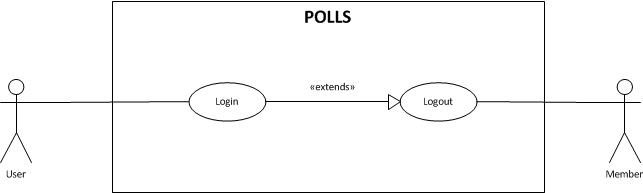
* It loads the HTML document and sanitizes it
* It only returns the body content of the HTML - so no styling (other than inline styles) will get through.

Specifications 2

In this specification of assignment, you must to finish all feature in bellow Use case:



As you may have found that the skeleton application already provides user authentication by checking if the username and password are both demo or admin.



You must implement a program that manages Survey and it will follow the above use case.

* The user to do the login using user and password if existed account in DB.
* User need use java code to validate data before insert, update user information to database. So that, we was validated data on client by Javascript and you will validate by Java on server.
* Store user IP address information to “user” and “answer\_user” tables.
* Deploy your application into Tomcat server.

Mark scale :

|  |  |
| --- | --- |
| * Create project with architectures, Configuration, Mapping : 20% * Specification 1 : 30% | * Specification 2 : 40% * Non-Functional, Quality requirements: 10% |

**-- THE END --**