DevPlat

Kexiang Wang (kwang66) -Moderator: Sameet Sapra

1. Abstract

1.1. Project Purpose

I am trying to make a semi office automation for company with tech background. In the platform, manager could release requirements and employees will finish and review them.

1.2. Background/Motivation

I have once wrote a web app about document management. I believe the same idea could apply here and want to explore more stuff in web programming.

2. Technical Specifications

2.1. **Platform:** Website

2.2. **Programming Languages:** Java, JSP, JS/JQuery, CSS, etc

2.3. **Stylistic Conventions:** camelCase

2.4. **SDK:** jQuery

2.5. **IDE:** Eclipse

2.6. **Tools/Interfaces:** Chrome

2.7. **Target Audience:** tech company

3. Functional Specifications

3.1. Features

-User can register and login

-The role of users is distinguished based on each requirement.

-The manager/creator can create requirement and sets its parameters like deadline, priority, etc.

-The engineer will complete the requirement and upload relevant files if needed.

-The reviewer will review the result and give suggestion to engineer or submit the task to the creator.

3.2. Scope of project

The project will only focus on the workflow of requirement engineering. A comprehensive office automation system could have more functionalities such as employee management, role assignment, permission management, document management, statistics. I will first just ignore those.

4. Timeline:

4.1. Week 1 – Project deployment and login/register page

4.1.1 Login

- 2 : Login is implemented and uses some form of form validation to prevent bad input . Provide feedback when login is not successful.

- 2.5 : hash the user's password using some encryption before storage.

4.1.2 Register

- 2 : Register is implemented and uses some form of form validation to prevent bad input . Provide feedback when username is already in database.

- 2.5: Provide immediate feedback on "username" and "confirm password" without actually submit the form.

4.1.3 Database

- 2 : login and register is linked to database. Able to insert and query for user login information.

- 2.5 : Database interaction has MVC structure. Have model and DAO to execute database query.

4.1.4 UI

-2 : Have basic design and formatting. Obviously spent some time on the choice of font, color and overall style.

-2.5 : Formatting, font and color choice appeal to aesthetic. Have hovering effect or other non-trivial animation implemented. The whole design look consistent. Have login form and register form in the same page(switching between the two should not cause loading or refreshing.).

4.1.5 Test

-2 : Have thorough test for functions in back end logic.

-2.5 : Have thorough test for functions in back end logic. Use some framework to test on front-end by filling form and observe outcome.

4.2. Week 2

4.2.1 Manager/Creator Workflow

-2 : A creator can successfully create task and view task in his repository.

-2.5 : A creator can create task, edit task and delete task, any change that the creator made to the task should be logged can displayed as users view the task.

4.2.2 Navigation Bar

-2 : Create a functional navigation bar that could be used in all the pages except the login page.

-2.5 : Navigation bar has a wise choice of color, font, hover effect and animation. The whole design is of high standard.

4.2.3 Task related form(create task), table(view task), and the page for user to view all task related to him/her.

-2 : Form and table gather all necessary information about a task. All the tasks are properly displayed.

-2.5 : Have some nice design details in form and table. Pages should look informative but not messy. And a filter for displaying all tasks in different order.

4.1.4 Database

- 2 : All user action related to task is linked to database. Insert, update, delete and query are implemented.

- 2.5 : Database interaction has MVC structure. Have model and DAO to execute database query or update.

4.1.5 Test

-2 : Have thorough test for functions in back end logic.

-2.5 : Have thorough test for functions in back end logic. Use some framework to test on front-end by filling form and observe outcome.

4.3. Week 3

4.4. Week 4

5. Future Enhancements

What are some cool tweaks you'd want to make to your product after the core functionality is done? Are you planning to work on it in the future?