CPSC 332 Web Project Deliverable 4

Project Management Application

Evan Shoemaker

12/8/2022

GitHub URL

https://github.com/GUWebDevelopment/cpsc-332-web-development-final-project-Colexeco.git

Functional Requirement 1

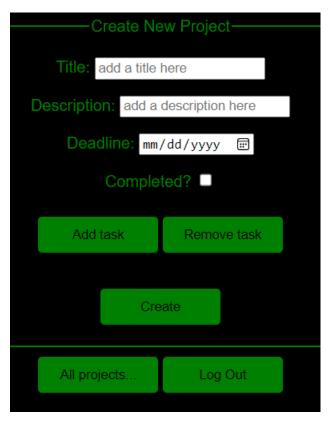
Users must be able to create project/task entries (front)

User enters title, description, deadline information and task information as desired...



Users must be able to retrieve previously created project/task entries (back)

User clicks on the "All projects..." button...



User is provided with central page to view all projects...



```
<body class="page-wrap"</pre>
   <h1>Project created successfully</h1>
      <caption>View all projects here...</caption>
         title
         description
          deadline
         completed
         tasks

    projectResults.forEach(function(response) { //debugger; %>

          <% console.log(response) %>
             <%= response.title %>
             <%= response.description %>
          <%= response.deadline %>
          <%= response.completed %>
             <%= response.numtasks %>
             <% if (response.numtasks > 0) { %>
             <a href="/viewTasks/<%= response._id %>" class="button">View Tasks</a>
             <% } %>
           <a href="/edit/<%= response._id %>" class="button">Edit</a>
             <a href="/delete/<%= response._id %>" class="button">Delete</a>
      «% }); %>
       <button class="button"><a class="button" href="/newProject">Create a new project</a></button>
      <button class="button"><a class="button" href="/logout">Log Out</a></button>
```

Users must be able to modify projects/tasks to reflect progress (front)

User clicks "edit" on project they would like to modify...



User edits desired fields and clicks "Edit (Save Changes)..."

	Edit Project							
	Title proj1 Description desc							
Deadline mm/dd/yyyy Completed false			i:					
All Projects (No	o changes saved)	Edit Tasks (No Changes Saved)	Edit (Save 0	Changes)				

Functional Requirement 4

Users must be able to retrieve past modifications to view progress (back)

Changes are sent to database when user clicks "Edit (Save Changes)..."

User can view project with new changes...



```
//UPDATE
app.route("/edit/:id")
    .get((req, res) => {
        if (req.session.userId) {
           validateSession(req.session.userId, res);
           let id = req.params.id;
            id = sanitize(id);//sanitization
            projectResult.findById(
                id,
                (err, results) => {
                    if (err) {
                        return res.render("error.ejs", {
                            errors: err
                        });
                    console.log("Found project! ");
                    console.log(results);
                    let result = {
                       _id: id,
                       title: results.title,
                        description: results.description,
                        deadline: results.deadline,
                        completed: results.completed,
                        tasks: results.tasks,
                    res.render("editProject.ejs", {
                        response: result
                    });
                });
            return res.redirect("/login");
```

```
.post(function (req, res) {
   if (req.session.userId) {
       validateSession(req.session, res);
       let id = req.params.id;
       id = sanitize(id);//sanitization
       let title = req.body.title;
       let description = req.body.description;
       let deadline = req.body.deadline;
let completed = req.body.completed;
       let tasks = req.body.tasks;
       let data = {
            title: title,
            description: description,
            deadline: deadline,
            completed: completed,
            tasks: tasks
        let validation = new Validator(data, projectRules,
                required: "A valid :attribute is required",
boolean: ":attribute must be true/false",
                date: "Please provide a valid date",
       console.log("Validation Passes: " + validation.passes() + " Validation Fails: " + validation.fails());
        if (validation.fails()) {
            let errorsList = {
                title: validation.errors.first("title"),
                description: validation.errors.first("description"),
                deadline: validation.errors.first("deadline"),
                completed: validation.errors.first("completed")
```

```
if (validation.fails()) {
            let errorsList = {
                title: validation.errors.first("title"),
                description: validation.errors.first("description"),
                deadline: validation.errors.first("deadline"),
                completed: validation.errors.first("completed")
            res.render("error.ejs", {
               errors: 3,
                errors: errorsList,
        } else {
            projectResult
            .where({_id: id })
            .updateOne({
                $set: {
                   title: title,
                   description: description,
                    deadline: deadline,
                    completed: completed,
                    tasks: tasks,
            .exec(function (err, result) {
                if (err) return res.send(err);
                res.redirect("/viewProjects");
                console.log(`Successfully updated ${result.modifiedCount} record`);
       return res.redirect("/login");
});
```

Users must be able to add deadlines to projects/tasks (front)

User enters deadline for project and any desired tasks...

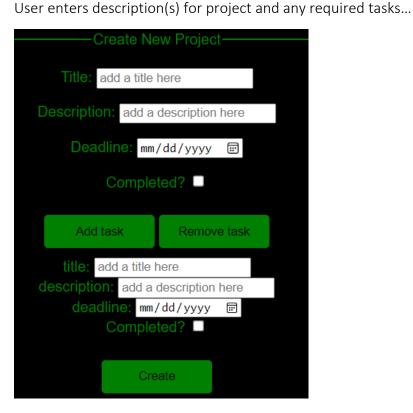


Users must be able to retrieve previously created deadlines for projects/tasks (back)

User can view deadlines at the central project page...

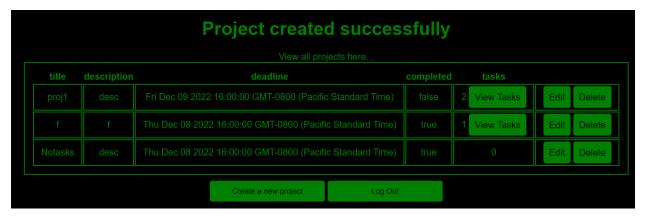


Users must be able to fill out descriptions for projects/tasks (front)



Users must be to retrieve previously created descriptions for projects/tasks (back)

User navigates to view all projects...



```
//READ

vapp.get("/viewProjects", (req, res) => {

vif (req.session.userId) {
 validateSession(req.session.userId, res);

projectResult.find(
 {},
 (err, results) => {
 console.log("Results: " + results)
 res.render("viewProjects.ejs", {
 projectResults: results,
 });
 });

v else { //not logged in
 return res.redirect("/login");
 }
});
```

Functional Requirement 9

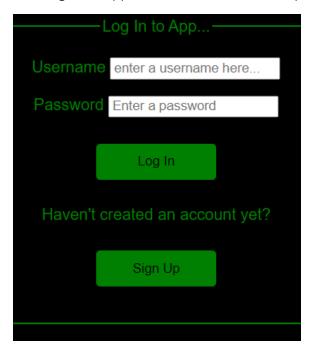
Users must be able to log in and out of application using a username and password (front)

User logs in with username and password...

User creates a new account...

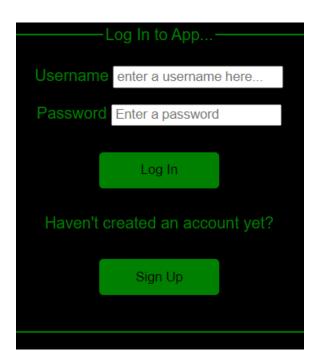
Create a New Account	
Username enter a username here	
Password ••••	
Password ····	
Sign Up	

User logs into app with their username and password...



Functional Requirement 10

Users must be able to log back in with previous name and password from database (back)
User enters credentials to log back in...



```
app.route("/signUp")
    .get((req, res) => {
        let errors = {
           usernameError: "",
       res.render("signUp.ejs", errors);
    .post((req, res) => {
        if (req.body.username && req.body.password && req.body.passwordConfirmation) {
            var userData = userCredentials({
                username: req.body.username,
                password: req.body.password,
            });
            userData.save(function (err, user) {
                if (err) {
                    let errors = {
                        usernameError: "Error: " + err
                    res.render("signUp.ejs", errors);
                   return res.redirect("/newProject");
            });
app.route("/login")
    .get((req, res) => {
        let errors = {
           usernameError: ""
       res.render("login.ejs", errors);
    .post((req, res) => {
        if (req.body.username && req.body.password) {
            var userData = {
               username: req.body.username,
                password: req.body.password,
            let temp = userCredentials.authenticate(userData, req, res);
    });
```