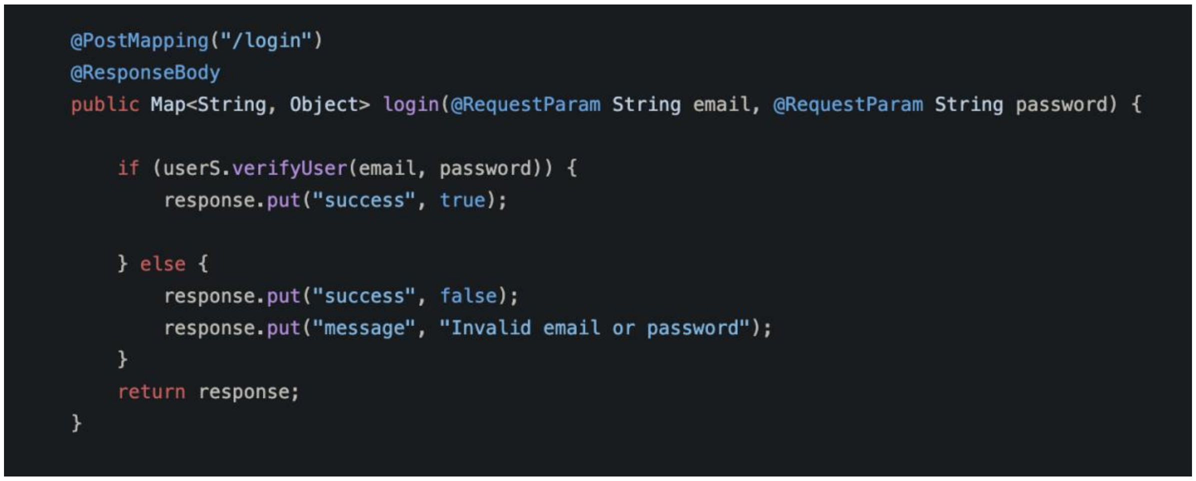
REFACTORING REPORT

Log In(Back-End)

UserController:

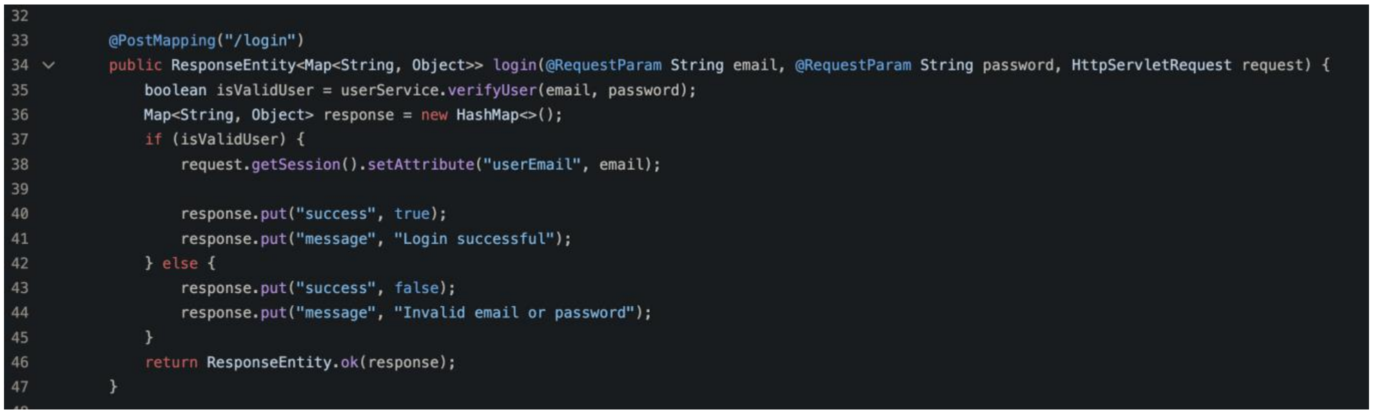


```java
@PostMapping("/login")
@ResponseBody
public Map<String, Object> login(@RequestParam String email, @RequestParam String password) {

    if (userS.verifyUser(email, password)) {
        response.put("success", true);

    } else {
        response.put("success", false);
        response.put("message", "Invalid email or password");
    }
    return response;
}
```

Refactoring 1

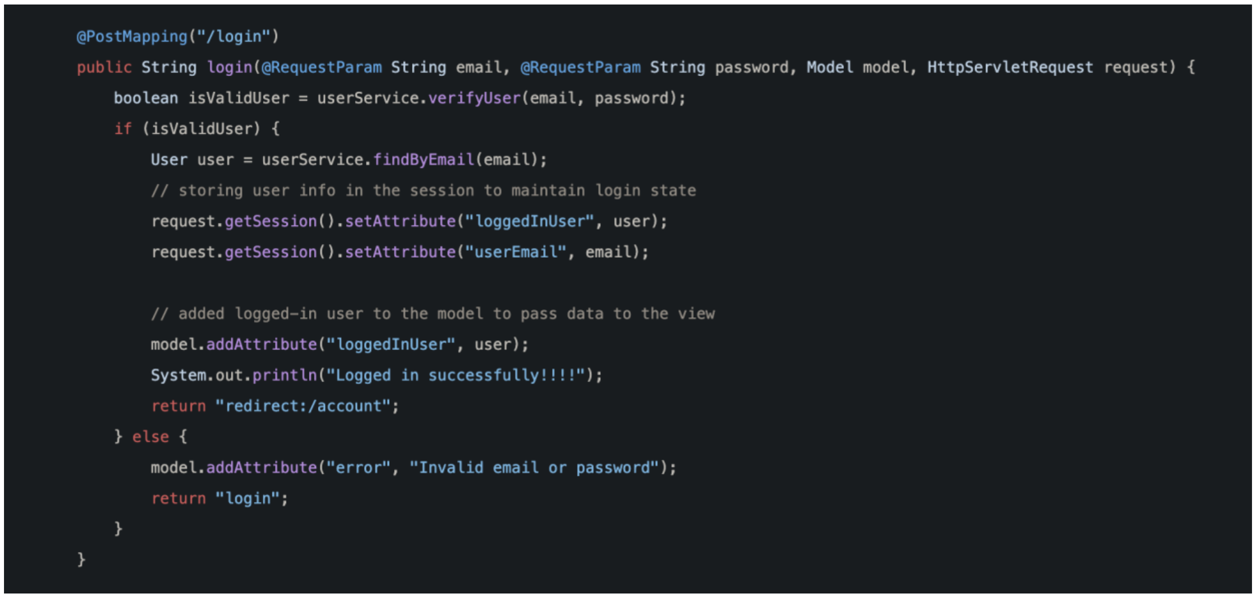


```java

    @PostMapping("/login")
    public ResponseEntity<Map<String, Object>> login(@RequestParam String email, @RequestParam String password, HttpServletRequest request) {
        boolean isValidUser = userService.verifyUser(email, password);
        Map<String, Object> response = new HashMap<>();
        if (isValidUser) {
            request.getSession().setAttribute("userEmail", email);

            response.put("success", true);
            response.put("message", "Login successful");
        } else {
            response.put("success", false);
            response.put("message", "Invalid email or password");
        }
        return ResponseEntity.ok(response);
    }
```

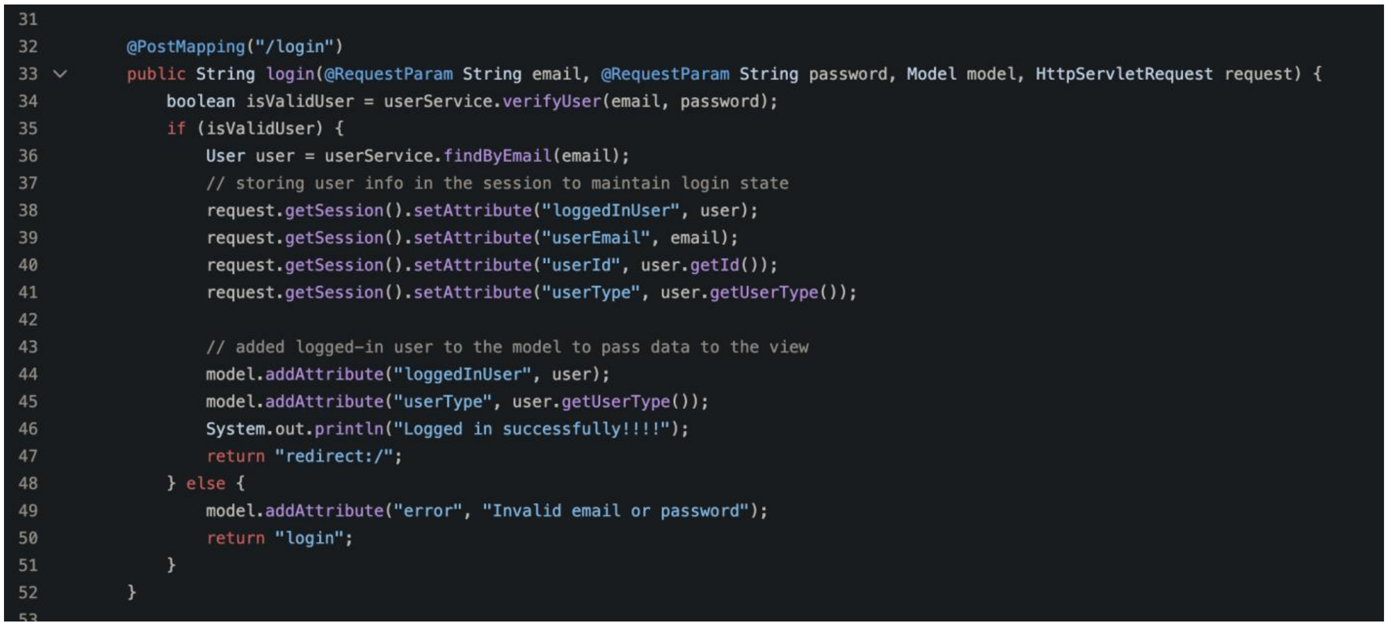
Refactoring 2



```
@PostMapping("/login")
public String login(@RequestParam String email, @RequestParam String password, Model model, HttpServletRequest request) {
    boolean isValidUser = userService.verifyUser(email, password);
    if (isValidUser) {
        User user = userService.findByEmail(email);
        // storing user info in the session to maintain login state
        request.getSession().setAttribute("loggedInUser", user);
        request.getSession().setAttribute("userEmail", email);

        // added logged-in user to the model to pass data to the view
        model.addAttribute("loggedInUser", user);
        System.out.println("Logged in successfully!!!!");
        return "redirect:/account";
    } else {
        model.addAttribute("error", "Invalid email or password");
        return "login";
    }
}
```

Refactoring 3


```java
    @PostMapping("/login")
    public String login(@RequestParam String email, @RequestParam String password, Model model, HttpServletRequest request) {
        boolean isValidUser = userService.verifyUser(email, password);
        if (isValidUser) {
            User user = userService.findByEmail(email);
            // storing user info in the session to maintain login state
            request.getSession().setAttribute("loggedInUser", user);
            request.getSession().setAttribute("userEmail", email);
            request.getSession().setAttribute("userId", user.getId());
            request.getSession().setAttribute("userType", user.getUserType());

            // added logged-in user to the model to pass data to the view
            model.addAttribute("loggedInUser", user);
            model.addAttribute("userType", user.getUserType());
            System.out.println("Logged in successfully!!!!");
            return "redirect:/";
        } else {
            model.addAttribute("error", "Invalid email or password");
            return "login";
        }
    }
```

Version 1:

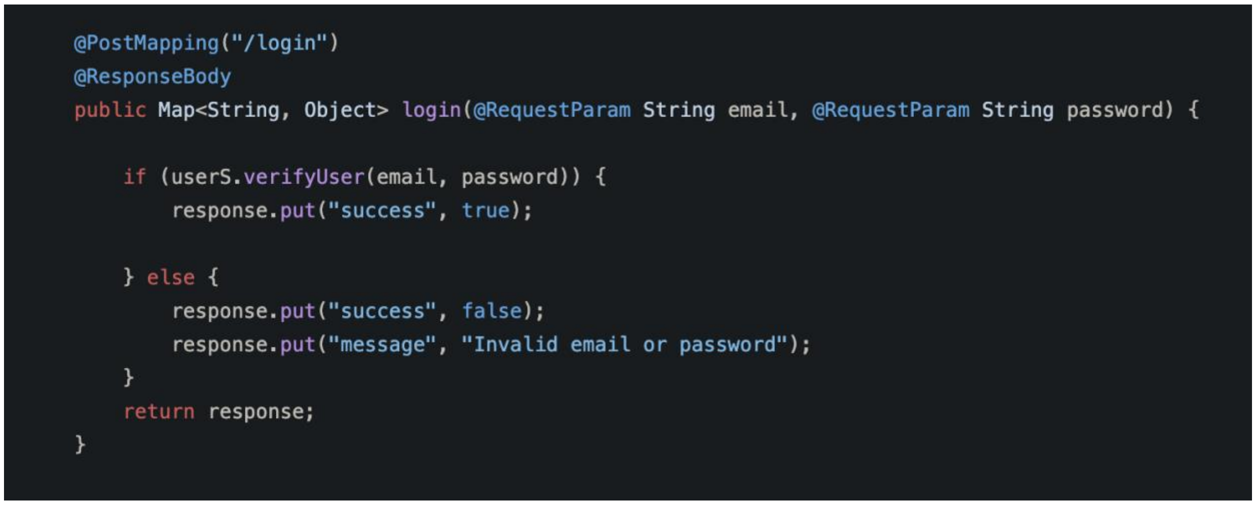
(login Branch)

UserController:

Login Method:

https://github.com/cosc2299-2024/team-project-group-p09-02/blob/login/src/main/java/au/edu/rmit/sept/webapp/controllers/UserController.java

Initial stage:



```java
@PostMapping("/login")
@ResponseBody
public Map<String, Object> login(@RequestParam String email, @RequestParam String password) {

    if (userS.verifyUser(email, password)) {
        response.put("success", true);

    } else {
        response.put("success", false);
        response.put("message", "Invalid email or password");
    }
    return response;
}
```

- Uses @ResponseBody, which directly returns a JSON response, making it suitable for API-based interactions.
- Focuses on verifying user credentials.

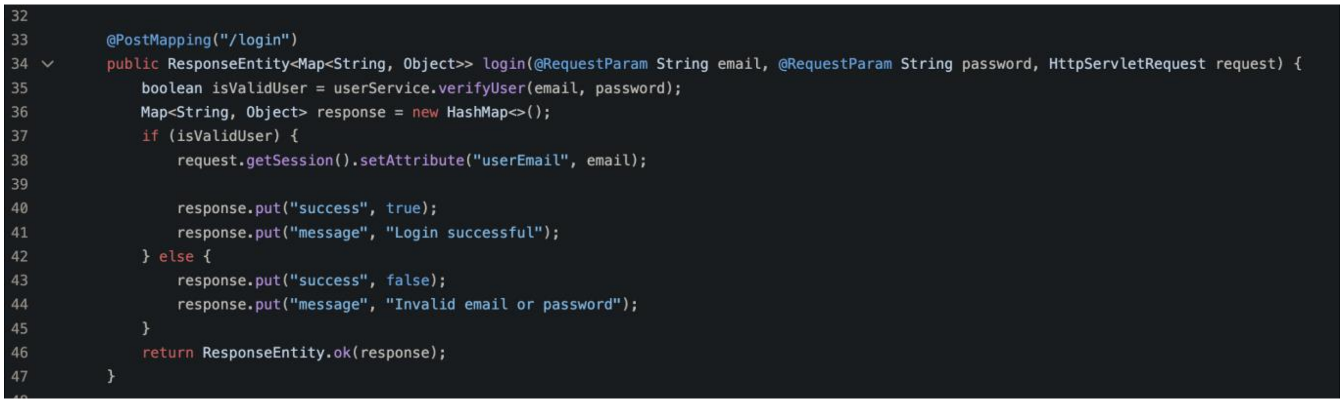
Version 2:

(profile Branch)

UserController:

Login Method: https://github.com/cosc2299-2024/team-project-group-p09-02/blob/profile/src/main/java/au/edu/rmit/sept/webapp/controllers/UserController.java

Refactoring stage 1:



```java

    @PostMapping("/login")
    public ResponseEntity<Map<String, Object>> login(@RequestParam String email, @RequestParam String password, HttpServletRequest request) {
        boolean isValidUser = userService.verifyUser(email, password);
        Map<String, Object> response = new HashMap<>();
        if (isValidUser) {
            request.getSession().setAttribute("userEmail", email);

            response.put("success", true);
            response.put("message", "Login successful");
        } else {
            response.put("success", false);
            response.put("message", "Invalid email or password");
        }
        return ResponseEntity.ok(response);
    }
```

- Uses ResponseEntity to wrap the response, which provides more control over the HTTP response status and headers.

- Includes basic session management by storing the user's email in the session upon successful login.

Version 3:

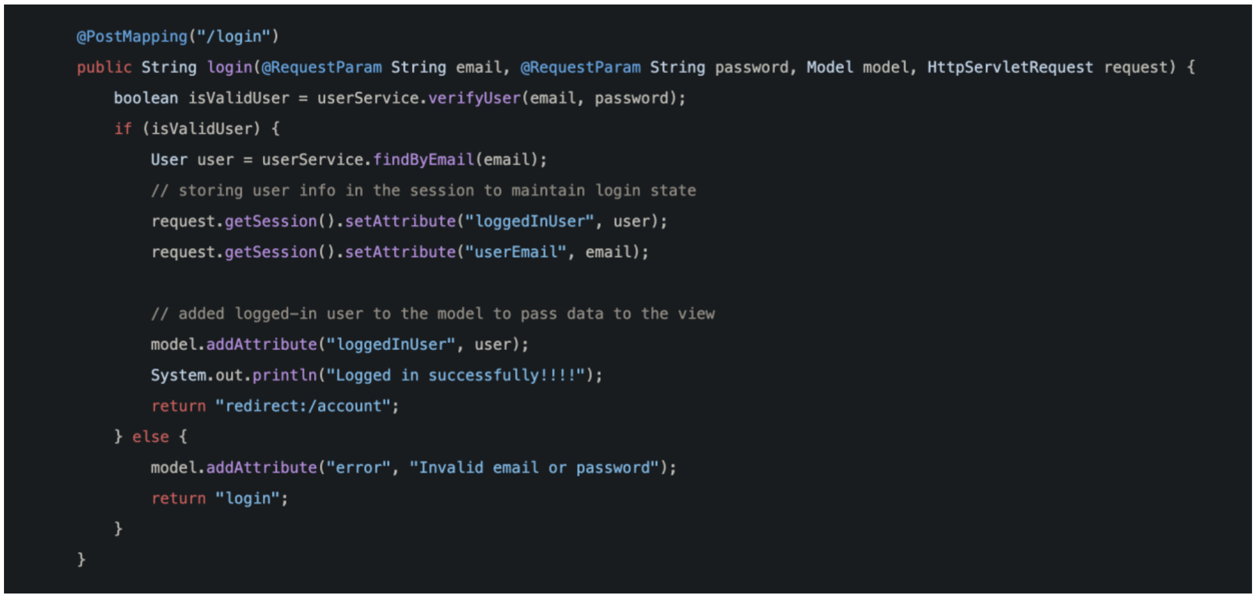
(editprofile Branch)

UserController:

Login Method: https://github.com/cosc2299-2024/team-project-group-p09-02/blob/editprofile/src/main/java/au/edu/rmit/sept/webapp/controllers/UserController.java

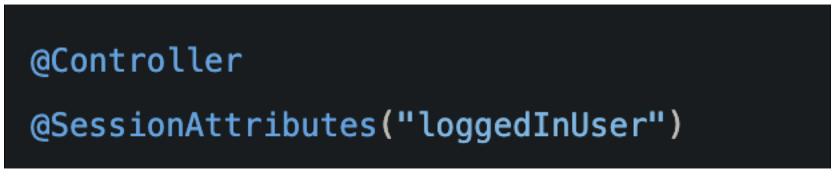
Refactoring stage 2:

https://github.com/cosc2299-2024/team-project-group-p09-02/commit/69d8bd09052e3be56414b39e39c15cccaabe643f#diff-0829d0339939f3d55b3a2180a96f12b1d387d7b55ee42353d6dc99af768417cfR40



```
    @PostMapping("/login")
    public String login(@RequestParam String email, @RequestParam String password, Model model, HttpServletRequest request) {
        boolean isValidUser = userService.verifyUser(email, password);
        if (isValidUser) {
            User user = userService.findByEmail(email);
            // storing user info in the session to maintain login state
            request.getSession().setAttribute("loggedInUser", user);
            request.getSession().setAttribute("userEmail", email);

            // added logged-in user to the model to pass data to the view
            model.addAttribute("loggedInUser", user);
            System.out.println("Logged in successfully!!!!");
            return "redirect:/account";
        } else {
            model.addAttribute("error", "Invalid email or password");
            return "login";
        }
    }
```


```
@Controller
@SessionAttributes("loggedInUser")
```

- Implements full session management, storing the complete user object in the session.
- Follows the MVC pattern, utilizing the Model to pass data to the view and handling redirection appropriately.
- Suitable for render server-side views as some features are only available to logged-in users.

Version 4:

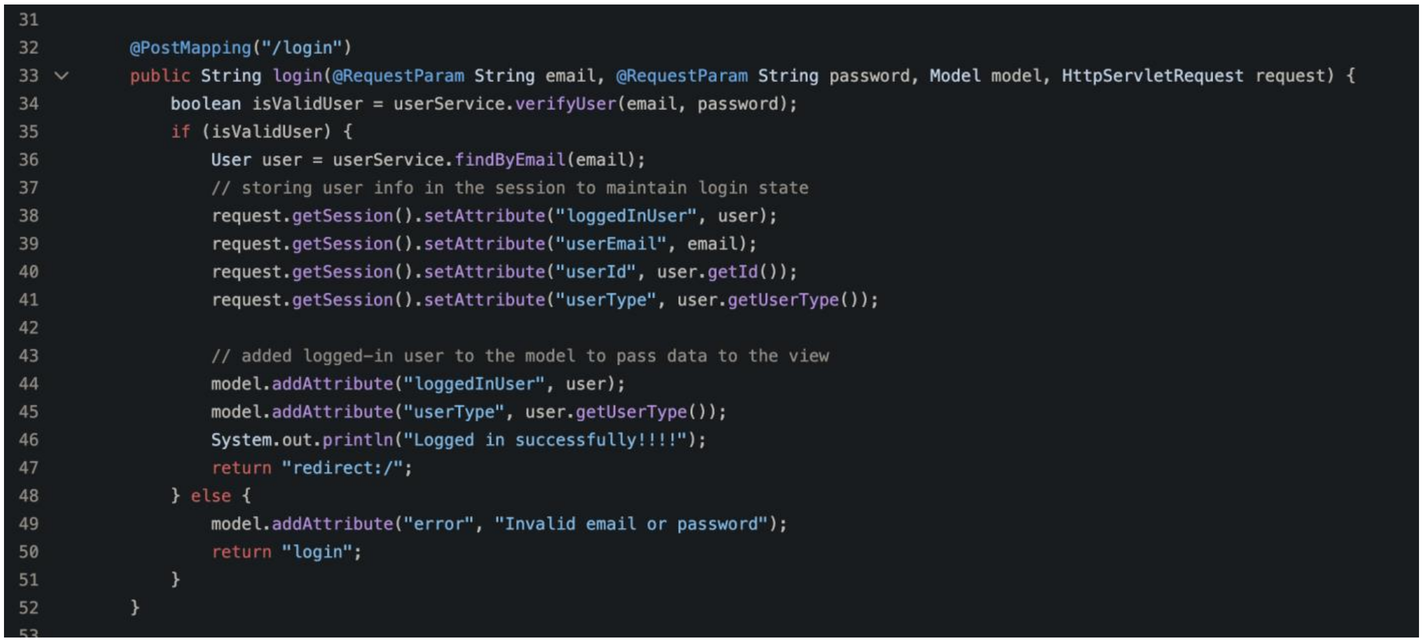
(vet_user Branch)

UserController:

Login Method: https://github.com/cosc2299-2024/team-project-group-p09-02/blob/editprofile/src/main/java/au/edu/rmit/sept/webapp/controllers/UserController.java

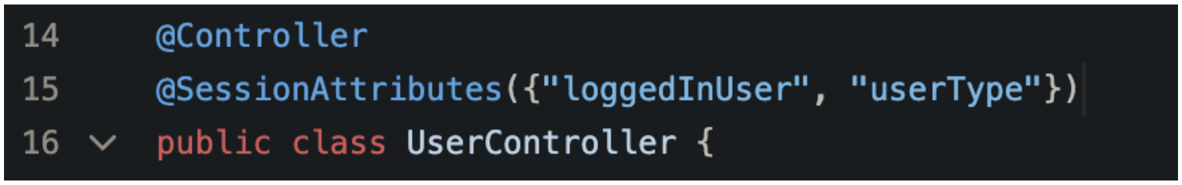
Refactoring stage 3:

https://github.com/cosc2299-2024/team-project-group-p09-02/commit/a54d4f63d3aadd9dcfb5282bdd624955678bd948#diff-0829d0339939f3d55b3a2180a96f12b1d387d7b55ee42353d6dc99af768417cf



```java
    @PostMapping("/login")
    public String login(@RequestParam String email, @RequestParam String password, Model model, HttpServletRequest request) {
        boolean isValidUser = userService.verifyUser(email, password);
        if (isValidUser) {
            User user = userService.findByEmail(email);
            // storing user info in the session to maintain login state
            request.getSession().setAttribute("loggedInUser", user);
            request.getSession().setAttribute("userEmail", email);
            request.getSession().setAttribute("userId", user.getId());
            request.getSession().setAttribute("userType", user.getUserType());

            // added logged-in user to the model to pass data to the view
            model.addAttribute("loggedInUser", user);
            model.addAttribute("userType", user.getUserType());
            System.out.println("Logged in successfully!!!!");
            return "redirect:/";
        } else {
            model.addAttribute("error", "Invalid email or password");
            return "login";
        }
    }
```

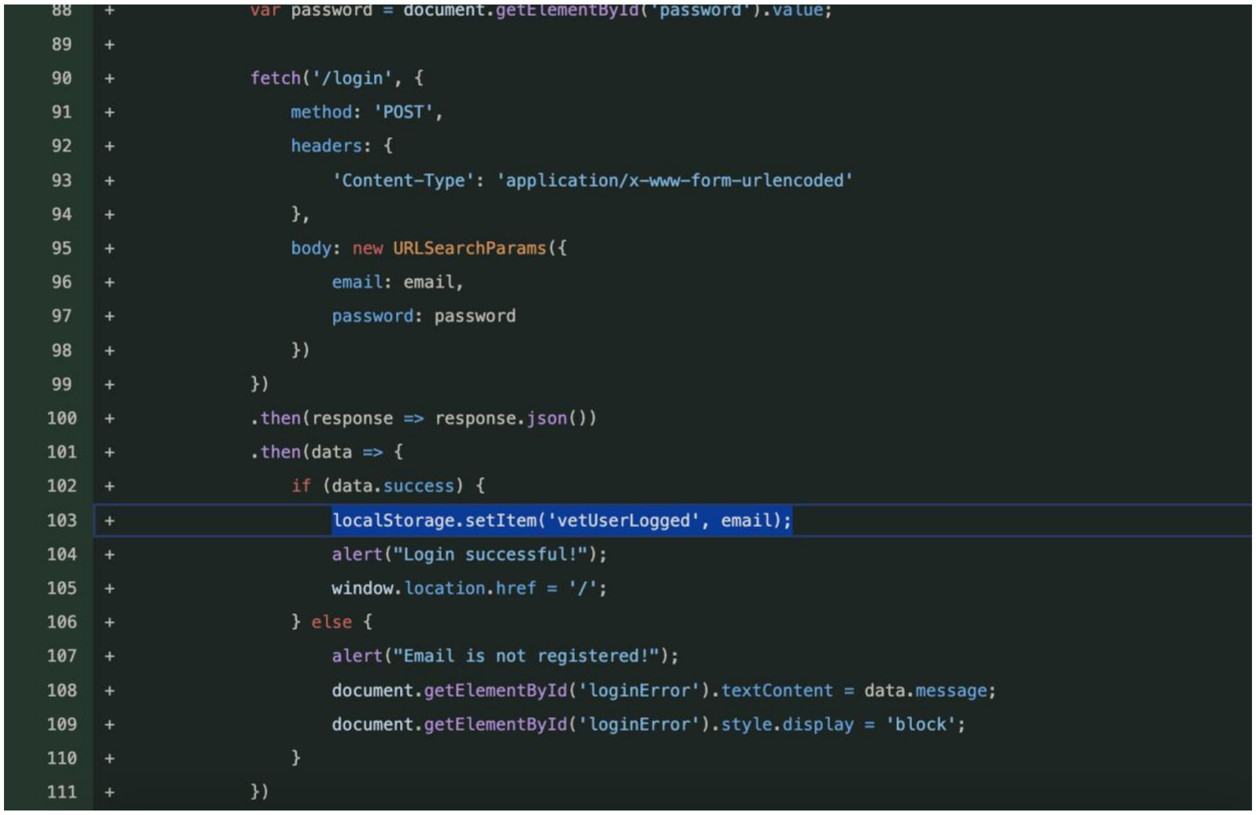


```java
@Controller
@SessionAttributes({"loggedInUser", "userType"})
public class UserController {
```

- Added userType as session attribute for Vet users as some feature's availability depends on userType

Log In(Front-End):

https://github.com/cosc2299-2024/team-project-group-p09-02/commit/a54d4f63d3aadd9dcfb5282bdd624955678bd948#diff-0829d0339939f3d55b3a2180a96f12b1d387d7b55ee42353d6dc99af768417cf

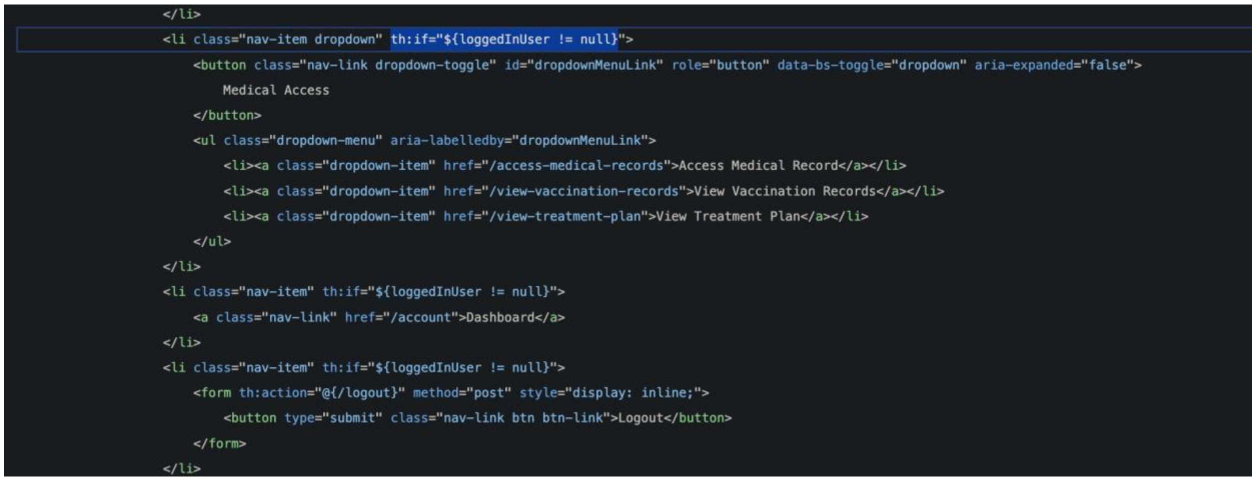


```
+             var password = document.getElementById('password').value;
+
+             fetch('/login', {
+                 method: 'POST',
+                 headers: {
+                     'Content-Type': 'application/x-www-form-urlencoded'
+                 },
+                 body: new URLSearchParams({
+                     email: email,
+                     password: password
+                 })
+             })
+             .then(response => response.json())
+             .then(data => {
+                 if (data.success) {
+                     localStorage.setItem('vetUserLogged', email);
+                     alert("Login successful!");
+                     window.location.href = '/';
+                 } else {
+                     alert("Email is not registered!");
+                     document.getElementById('loginError').textContent = data.message;
+                     document.getElementById('loginError').style.display = 'block';
+                 }
+             })
```

- Used Local Storage to store the logged in user for some specific privileges to them

Refactoring Stage 1:

https://github.com/cosc2299-2024/team-project-group-p09-02/commit/69d8bd09052e3be56414b39e39c15cccaabe643f#diff-47fa33545b28d24c276a71ea1781ba04c1a72c88b94f2bc46880b0e24c92240a



```
            </li>
            <li class="nav-item dropdown" th:if="${loggedInUser != null}">
                <button class="nav-link dropdown-toggle" id="dropdownMenuLink" role="button" data-bs-toggle="dropdown" aria-expanded="false">
                    Medical Access
                </button>
                <ul class="dropdown-menu" aria-labelledby="dropdownMenuLink">
                    <li><a class="dropdown-item" href="/access-medical-records">Access Medical Record</a></li>
                    <li><a class="dropdown-item" href="/view-vaccination-records">View Vaccination Records</a></li>
                    <li><a class="dropdown-item" href="/view-treatment-plan">View Treatment Plan</a></li>
                </ul>
            </li>
            <li class="nav-item" th:if="${loggedInUser != null}">
                <a class="nav-link" href="/account">Dashboard</a>
            </li>
            <li class="nav-item" th:if="${loggedInUser != null}">
                <form th:action="@{/logout}" method="post" style="display: inline;">
                    <button type="submit" class="nav-link btn btn-link">Logout</button>
                </form>
            </li>
```

- Using the session attribute to verify if any user is logged in or not.