***Condo Management System***

**Testing Plan for Sprint #4**

22/03/2024

**Testing Approaches**

Testing strategies for this project will encompass a diverse range to optimize our testing plan's efficiency and effectiveness. Throughout the project timeline, we will employ Agile methodology, conducting iterative testing during sprints. Complementing Agile, Test Driven Development will be integrated, prioritizing test writing before code implementation. Exploratory testing will also be conducted to enhance user experience, allowing simultaneous testing of various system functions. Moreover, our CI Pipeline will be configured to automatically execute tests, facilitating early issue detection during development.

**Testing Tools, Coverage and Metric**

For our project, we're utilizing the Next.js framework for both backend and frontend development. Therefore, our choice of testing tools must seamlessly integrate with these frameworks. Since Next.js leverages React.js for frontend development, JEST emerges as the optimal testing tool for unit and integration tests due to its compatibility. JEST is renowned for its "zero-configuration" testing environment and includes a built-in code coverage tool, crucial for our project's objectives. We'll be extensively utilizing unit and integration tests to generate testing analytics and validate code quality.

In terms of testing metrics, JEST provides insights into test coverage, execution time, and pass/fail ratios. Running "npx jest --coverage" in the project terminal generates a coverage directory containing details on statements, branches, functions, and lines. Similarly, a command is used to assess execution time. Results from these tests will be stored and evaluated in a concise testing report. Our target is to achieve a minimum code coverage of 80%, a figure we deem both achievable and indicative of the system's robust functionality.

To gauge code quality, we'll leverage ESLint, a static code analysis tool renowned for pinpointing potential errors and enhancing overall code quality. Integrated into IDEs, ESLint offers real-time feedback as the team codes, ensuring adherence to best practices. With the flexibility to install numerous plugins and extensions, we can encompass a wide array of code quality metrics.

ESLint performs static code analysis during execution in the terminal for the entire project, highlighting violations, code duplication, and performance metrics. It provides valuable feedback, indicating potential errors and areas for improvement. Fixable errors can be addressed by running the command "npx eslint --fix" to automatically resolve detected issues. To address code duplication, an ESLint plugin will be utilized, and the ESLint configuration file will be updated to detect duplication during project-wide terminal execution.

Employing these tools will enhance code readability, detect bugs at an early stage of development, and ensure adherence to a uniform coding standard. Integration of static code analysis tools will be seamlessly incorporated into the continuous integration pipeline.

Another key testing metric we prioritize is defect density, which quantifies the number of defects discovered per unit of code. We view this metric as instrumental in guiding testers to craft effective and beneficial tests for the project. Test coverage results will be utilized to compute defect density, aiding in our assessment of testing efficacy.

**Acceptance Tests**

**Feature:** access the details of the property from the mobile app ([US #163](https://github.com/CONCORDIA-SOEN-390/Condo-Mgmt-Web-App/issues/163))

Given I am a condo management company authorized representative accessing the financial system page, I want to have access to all the details, such as annual reports showing all condo fees collected on the mobile app.

| **Acceptance Criteria** | **Example Test Data** |
| --- | --- |
| * The detailed information entry section should be accessible to authorized representatives of the condo management company. * The system should allow the entry of detailed information for each condo unit, parking spot, and locker. * The information for each parking spot and locker should include relevant details and associations. * The entered information should be accurate and reflect the current state of each condo unit, parking spot, and locker. * All of this should be on mobile | -Name: Elderwood condos  -Condo unit: 45k  -Condo area: 300  -Parking Spot: 2  -Unit Size: 1000 sq. ft |

**Feature:** Reserve common facilities in a calendar-like interface. ([US #63](https://github.com/CONCORDIA-SOEN-390/Condo-Mgmt-Web-App/issues/63))

Given I am a condo management company authorized representative, I want to have an easy to use and clear interface where I can reserve facilities. It has to look like a physical calendar and have quick and easy navigation.

| **Acceptance Criteria** | **Example Test Data** |
| --- | --- |
| * The system should provide a calendar-like interface where users can view availability and make reservations for common facilities such as meeting rooms, gym, pool, etc. * The calendar should display dates clearly and allow users to navigate between months and years easily. * Users should be able to select the date and time they want to reserve a common facility. * The system should display the availability of the facility for the selected date and time, showing booked slots as unavailable. * The system should validate the selected date and time to ensure it falls within the operating hours of the common facility and does not conflict with existing reservations. * If the selected date or time is invalid or conflicts with existing reservations, appropriate error messages should be displayed, guiding users to select alternative options. |  |

**Feature:** access details of profile ([US #161](https://github.com/CONCORDIA-SOEN-390/Condo-Mgmt-Web-App/issues/161))

Given I am a user of the app, I should be able to have access to the details of my profile. I should be able to not only see my details, but also edit them in case they change. It should display things such as my profile picture, name and list of properties.

| **Acceptance Criteria** | **Example Test Data** |
| --- | --- |
| * User should be able to display and upload a profile picture * Users should be able to set and change their name * Users should be able to display their gmail. * Users should be able to display and edit their phone number | PROFILE PICTURE  Name: Gladius Maximus  Email: ramranch@gmail.com  Phone number: 514 7346784673290234 |

**Feature:** Access account on the mobile version of the platform ([US #157](https://github.com/CONCORDIA-SOEN-390/Condo-Mgmt-Web-App/issues/157))

As a user who has a working account, I want to access my account on the mobile version of the platform.

| **Acceptance Criteria** | **Example Test Data** |
| --- | --- |
| * The platform should have a responsive design that adapts to various screen sizes and orientations, ensuring optimal usability and readability on mobile devices. * Users should be able to access their accounts by logging in using their credentials (e.g., username/email and password) on the mobile version of the platform. * The login process should be intuitive and user-friendly, with appropriate error messages displayed for invalid credentials or login failures. * Once logged in, users should be able to access their account information, including profile details, and settings through the mobile interface. | Username: Scooby  Password: Secret123? |