Sprint (2) Release Plan

**Completed Tasks**

Tasks concerning documentation that are expected to be completed include the software architecture, risk assessment and risk management plan, product vision statement, requirements and user stories, testing plan for sprint 3, sprint 2 retrospective, release plan and UI prototypes for sprint 3 documents.

What has been completed thus far from sprint 1 is a running prototype of the signup and login page for the system.

Specific tasks to be completed during Sprint 2 include the following user stories,

0103: Condo owner account with a registration key,

0104: Rental user account with a registration key,

0105: Employee account,

0106: Dashboard of properties,

0107: Condo company administrator can create a profile of a property,

0208: Condo company administrator can upload files for each of their properties,

0209: Condo company administrator can enter and modify information on a unit,

0310: An administrator can send registration keys to condo owners and renters,

with a main focus on epics 01 and 02. The table below illustrates the features backlog for Sprint 2.

| ***User Story Backlog Sprint (2)*** | | | |
| --- | --- | --- | --- |
| **User Story ID** | **User Story Points**  **(USP)** | **Priority** | **Status** |
| 0101 | 2 | High | DONE |
| 0102 | 2 | High | DONE |
| 0103 | 2 | High | PUSHED TO SPRINT 2 |
| 0104 | 2 | High | PUSHED TO SPRINT 2 |
| 0105 | 2 | High | TODO |
| 0106 | 8 | High | TODO |
| 0107 | 3 | High | TODO |
| 0208 | 3 | Medium | TODO |
| 0209 | 5 | High | TODO |
| 0310 | 5 | High | TODO |
| **Total USP** | 32 |  |  |

**Future Deployment**

The system will be built first as a website, then integrated to create a progressive web app (PWA), by embedding the website into a webview in an android activity. Users can install the PWA to their home screen, desktop or any OS and launch it as you would an application. Given that our system will be built using NextJs, we must convert the web application into a native app by installing an ‘npm’ plugin through the command line. Instructions for the plugin are given from the website in which they are downloaded. Such instructions will be followed. However, generally after installing the plugin through the command line, a dependency object will be added to the ‘package.json’ application file. Further, we must update ‘next.config.js’ file, create a service worker that enables PWA for the NextJs application, create ‘manifest.json’ and properly set up the file and update layout ‘layout.tsx’ by adding the manifest file to the ‘MetaData’. Once more, we run ‘npm-run-dev’ to add the service worker Js files to the project. It will generate code in a file titled ‘workbox’. After running the command, the terminal should yield the file path for ‘sw.js’ where ‘sw’ is service worker. Then, refresh the application in the browser. Finally, inspect the NextJs app in the browser, open LightHouse and choose the option ‘Progressive Web App’ from the list to convert the NextJs application into a PWA.

For mobile devices such as iOS and Android with Chrome, we must open the application in the browser and click the option to “add to Home screen”. Moreover, we can deploy the PWA to any hosting service provider.

Such steps will be taken in Sprint 2.