Sprint (2) Release Plan

User story 5: 5 USP

1. Create a property schema in MongoDB to include the following: owner profile, condo information, financial status (remaining balance on condo fees), status of submitted request, etc.
2. Develop a property card layout in React that will display property information.
3. Develop a dashboard layout in React
4. Develop RESTful API endpoints in Django to handle CRUD operations for property information.
5. Utilize Axios or Fetch API in React to connect with Django's API endpoints.
6. Implement user authentication to ensure secure access to the owner's dashboard.
7. Write unit and integration tests for both frontend and backend components

User Story 6: 8 USP

1. Design and implement a schema in MongoDB to store property profiles, including fields for property name, unit count, parking count, locker count, and address. Use Django with Django REST Framework for creating the API layer that interacts with MongoDB, ensuring data integrity and ease of data manipulation.
   - Design MongoDB Schema for Property Profiles
   - Implement Django Models for Property Profiles
   - Set Up Django REST Framework for API Development
   - Implement CRUD Operations in Django API Layer
   - Test MongoDB Schema and Django API

2. Create models and views in Django to handle the upload, storage, and management of condo files (associated with each property. Store these files in MongoDB GridFS if file size exceeds BSON document size limit or choose an appropriate storage solution based on the project requirements.
   - Design Django Models for Condo Files
   - Implement File Upload Views in Django
   - Configure File Storage Settings in Django:
   - Implement Logic for File Storage in MongoDB
   - Create Views for File Management
   - Test File Upload and Management Functionality

3. Use React to develop dynamic, responsive interfaces that allow condo management companies to input and update detailed information for each condo unit, parking spot, and locker. These interfaces should include forms for ID, size, owner, occupant information, and associated condo fees.
   - Design React Components for Condo Management Interfaces
   - Implement Input Forms for Condo Unit Details
   - Create Input Forms for Parking Spot Details
   - Develop Input Forms for Locker Details
   - Handle Input Data and State Management
   - Integrate React Components with Backend API
   - Test React Components and Functionality

4. Utilize Django to develop a system for generating and sending registration keys to unit owners or rental users. This system should securely enable users to link their condo units with their profiles, using MongoDB to store the registration keys and their associations.
   - Design Django Models for Registration Keys
   - Implement Logic for Generating Registration Keys
   - Create Views for Sending Registration Keys
   - Develop Mechanism for Linking Condo Units with Profiles
   - Integrate MongoDB for Storing Registration Keys
   - Handle User Authentication and Authorization
   - Test Registration Key Generation and Linking Functionality

5. Make sure that all uploaded condo files are accessible to condo owners of the respective property through the React-based platform. Implement secure access controls and user authentication using Django to safeguard sensitive information.
   - Design Access Control Mechanism
   - Implement User Authentication in Django
   - Configure Role-based Access Control (RBAC)
   - Integrate Access Control with File Upload and Management
   - Implement Secure Access Controls in React Components
   - Handle Error Cases and Edge Scenarios
   - Test Access Control Functionality

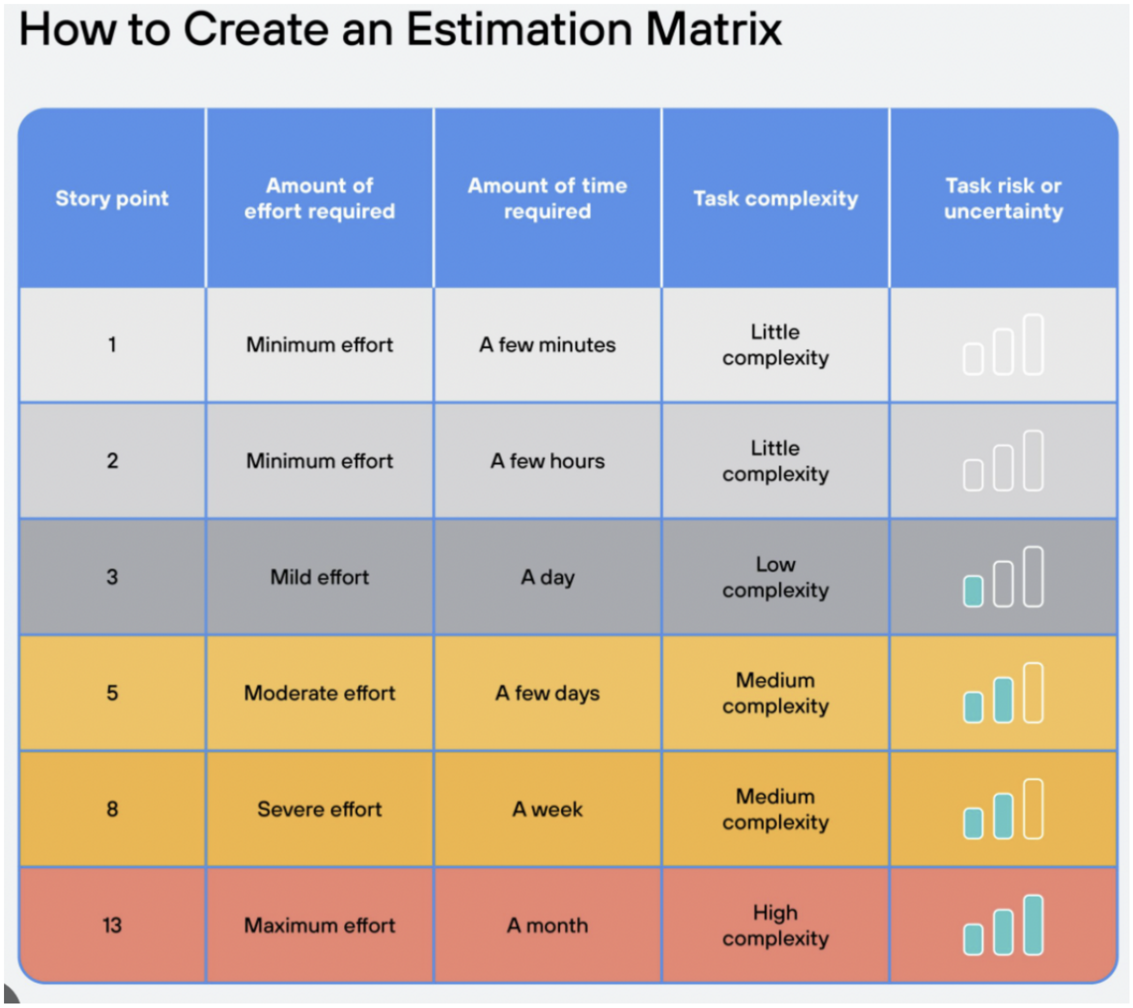
6. Craft a user-friendly, intuitive UI for condo management companies to effortlessly create property profiles, upload documents, and manage unit details. The design should prioritize ease of use and accessibility, ensuring a seamless experience for all users.
   - UI Layout Design
   - Create Property Profile Form
   - Implement Document Upload Feature
   - Design Unit Details Management Interface
   - Test Features Ensuring Optimal Usability

User Story ID	User Story Points (USP)	Priority	Status
US5 (#7)	5 USP	● Medium	● TODO
US6 (#11)	8 USP	● High	● TODO
Total USP	13 USP		

For Future deployments:

The website will be deployed using GitHub Pages, which serves the site directly from a repository on GitHub, offering an efficient and integrated approach to hosting. By pushing the site's build to a `gh-pages` branch or a `/docs` folder, updates and maintenance become a natural extension of the version control workflow.

For android deployment we are planning on using a progressive web app (PWA) to allow our app to be used on an android device.



How to Create an Estimation Matrix

Story point	Amount of effort required	Amount of time required	Task complexity	Task risk or uncertainty
1	Minimum effort	A few minutes	Little complexity	▯▯▯
2	Minimum effort	A few hours	Little complexity	▯▯▯
3	Mild effort	A day	Low complexity	▮▯▯
5	Moderate effort	A few days	Medium complexity	▮▮▯
8	Severe effort	A week	Medium complexity	▮▮▯
13	Maximum effort	A month	High complexity	▮▮▮

Task Name	Responsible	Start	End	Days	Status
Sprint 1	Joud	1/18	2/7	20	Complete
Login	Joud	1/25	2/3	9	Complete
Logout	Jeremy	2/1	2/7	6	Complete
Signup	Yashi	1/18	1/25	7	Complete
User Profile UI	Patrick	1/25	2/3	9	Complete
Registration Key: Condo Owners	Nicholas	1/26	2/7	12	Complete
Registration Key: Rental Users	Nicholas	1/26	2/7	12	Complete
Edit Profile	Joud	1/25	2/7	13	Complete
Sprint 2	Jeremy	2/8	2/28	20	Not Started
Create condo profile	Patrick	2/8	2/9	1	Not started
Upload files to condo profile	Ishai	2/17	2/21	4	Not started
Condo owners access	Alex	2/22	2/24	2	Not started
Daschboard of Properties	Eden	2/9	2/16	7	Not started
Financial Status of Request	Vithu	2/22	2/28	6	Not started