Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria
5	US1: Public User Sign Up	As Jake (public user) I want to be able to sign up to the website so that I can create a new account and start using the service	User Story	M	9	P0	- Design a user schema in MongoDB to store user details. - Create a sign-up API endpoint. - Implement the front-end React component for the sign-up form. - Add client-side validation for the sign-up form inputs. - Implement server-side validation and error handling for the sign-up process. - Write tests for the API endpoint ensuring it handles requests correctly.	1/28/2024	2/7/2024	Sprint_1	- The sign-up form is accessible on the website. - The form includes fields for username, email, phone number, and password. - The user receives a confirmation email upon successful sign-up. - The user's data is securely stored in MongoDB. - The user cannot sign up with an email that is already in use.
2	US2: Public User Login	As Jake (public user) I want to be able to log into the website so that I can access my profile and services	User Story	M	9	P0	- Develop a login API endpoint. - Create a React component for the login page with form validation. - mplement authentication using JWT (JSON Web Token) for session management. - Set up secure password storage and validation using bcrypt or a similar library. - Conduct integration tests for the login functionality.	1/28/2024	2/7/2024	Sprint_1	- Registered users can log in using their email and password. - Unregistered users cannot log in. - Users receive an error message when entering invalid credentials. - Upon login, users are directed to their profile page.
3	US3: Sign In User Create Pr	As Jake (public user) I want to be able to create my profile after I sign up so that I can personalize my experience on the website	User Story	M	9	P0	- Expand the user schema in MongoDB to include profile details. - Add API endpoints for retrieving and updating user profiles. - Design a profile management page in React. - Ensure secure file upload for the user's picture. - Include input fields for the required profile details. - Write unit tests for profile management functionalities.	1/28/2024	2/7/2024	Sprint_1	- Users can upload a picture, set a username, and provide contact details. - Changes to the profile are saved and persist upon logging out and back in. - Profile details are validated on both client and server sides.
4	US4: Sign In User Entering F	As Eric (owner user) I want to enter a registration key when signing up so that I can be recognized as either a condo owner or a rental user	User Story	S	6	P0	- Modify the sign-up process to include an input for the registration key. - Create a system to generate and validate keys for condo owners and rental users. - Implement logic to assign different roles and permissions based on the key. - Update the database schema to store user roles. - Test the key validation and role assignment process thoroughly.	2/8/2024	2/28/2024	Sprint_2	- Condo owners and rental users can sign up with their respective registration keys. - Users signing up with an invalid key receive an appropriate error message. - The system correctly assigns roles based on the provided key during sign-up.
7	US5: Create owner's propert	As Eric I want to be able to view a dashboard of the properties I own so that I can have access to their detailed information easily	User Story	L	12	P1	- Create a property schema in MongoDB to include to include the following: owner profile, condo information, financial status (remaining balance on condo fees), status of submitted request, etc. - Develop a property card layout in React that will display property information. - Develop a dashboard layout in React - Develop RESTful API endpoints in Django to handle CRUD operations for property information. - Utilize Axios or Fetch API in React to connect with Django's API endpoints. - Implement user authentication to ensure secure access to the owner's dashboard. - Write unit and integration tests for both frontend and backend components	2/8/2024	2/28/2024	Sprint_2	- The dashboard should display the owner's personal profile information. - The dashboard must provide detailed information about the condo, including location, size, amenities, and other relevant details - The dashboard should clearly show the financial status related to the condo - The dashboard must be intuitive and easy to navigate, allowing condo owners to find information quickly. - Access to the dashboard should be protected by secure login credentials
11	US6: Create Property Profile	As Brian (condo management compnay manager) , I want to create a profile for each property under my management with essential details like property name, unit count, parking count, locker count, and address, to be able to organize and manage property information efficiently.	User Story	L	12	P1	- Design a property profile schema in MongoDB to store property details, including name, unit count, parking count, locker count, and address. - Develop a Django RESTful API endpoint for creating property profiles. Ensure it handles validations for required fields. - Create a React component for the property profile creation form. Include input fields for property name, unit count, parking count, locker count, and address. - Utilize Axios or Fetch API in the React component to submit the form data to the Django backend. - Implement form validation in React to ensure all required fields are filled out before submission. - Write integration tests to verify the interaction between the React frontend, Django backend, and MongoDB database for creating property profiles.	2/8/2024	2/28/2024	Sprint_2	- The property profile creation form displays all required input fields. - Form submission fails with appropriate feedback if any required field is empty. - A property profile is successfully created in the MongoDB database upon submitting the form with valid data. - The user receives confirmation on the UI after successful profile creation.
22	US7: Upload File for Each Pr	As Brian (condo management compnay manager) , I want to upload and associate condo files with each property profile, to ensure all condo owners have easy access to important documents.	User Story	M	9	P1	- Update the property schema in MongoDB to include a field for storing file references - Develop a Django API endpoint for file uploads - Integrate a file upload component in the React frontend within the property profile management page. - Modify the React component to display a list of uploaded files for each property. - Write tests to verify that files are correctly uploaded and associated with the property profiles.	2/29/2024	3/21/2024	Sprint_3	- Users can upload files (PDFs) related to a property profile. - The system restricts uploads to acceptable file types and sizes. - Uploaded files are listed under the property profile they belong to. - Files are accessible to condo owners of the respective property.
24	US8: Enter Details for Every Component in the Property	As Brian (condo management compnay manager) , I want to enter and update detailed information for each condo unit, parking spot, and locker in a building, to maintain accurate records of all property components for better management and oversight.	User Story	L	12	P1	- Expand the property schema in MongoDB to include detailed information for condo units, parking spots, and lockers. - Create Django RESTful API endpoints for adding and updating detailed information for each condo unit, parking spot, and locker. - Develop React components for entering and updating details of condo units, parking spots, and lockers. Each component should allow input of all required fields. - Ensure the React components are integrated with the property profile, allowing detail entry from the property's management page. - Implement validation on both the frontend and backend to ensure data integrity. - Write comprehensive tests to validate the functionality of adding and updating details for property components.	2/8/2024	2/28/2024	Sprint_2	- Detailed information for condo units, parking spots, and lockers can be entered and updated. - Each detail entry is correctly associated with its respective property profile. - Data validation prevents incomplete or incorrect detail entries.

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria
25	US9: Implement Registration	As Brian (condo management company manager) , I want to send registration keys to unit owners or rental users to enable secure linking of condo units with their owners' or users' profiles for access control and information sharing.	User Story	S	6	P1	- Develop a system in Django to generate unique registration keys for unit owners or rental users. - Create a Django API endpoint for sending registration keys via email to unit owners or rental users. - Implement a feature in the property management React component that allows condo management companies to enter email addresses of unit owners or rental users and send registration keys. - Ensure security measures are in place to prevent unauthorized access to registration keys. - Write tests to verify that registration keys are generated, stored, and sent securely and correctly.	2/8/2024	2/28/2024	Sprint_2	- Condo management companies can send registration keys to unit owners or rental users. - Each registration key is unique and securely linked to the correct unit. - Unit owners or rental users receive the registration key via email and can use it to link their unit with their profile.
26	US10: Implement Condo Fe	As Brian (condo management company manager) , I want to calculate and communicate the fees applicable to each unit owner based on their unit size and parking spots.	User Story	S	6	P1	- Develop a feature in the financial module to allow entry of condo fee rates per square foot and per parking spot. - Ensure the data model in MongoDB supports storing these rates associated with each property. - Create a Django API endpoint to handle the submission of condo fee rates. - Design a React component for condo management companies to enter and update these rates. - Implement validation on the input to ensure the rates are positive numbers.	2/29/2024	3/21/2024	Sprint_3	- Condo management companies can enter and update condo fee rates per square foot and per parking spot. - The system stores these rates correctly associated with the respective property. - Validation prevents the entry of negative or invalid rate values.
27	US11: Calculate and Presen	As a Brian (condo management company manager) , I want the platform to automatically calculate and present the condo fee for each unit based on its size and associated parking spots, so that unit owners are informed about their dues in a transparent and consistent manner.	User Story	M	9	P2	-Implement a calculation engine in Django that automatically computes the condo fee for each unit based on its size, the number of parking spots, and the entered rates. - Update the unit detail view in React to present the calculated condo fees to unit owners. - Ensure the calculation takes place every time there is a change in unit size, parking spot allocation, or fee rates. - Develop a Django RESTful API endpoint that provides the calculated condo fees on request. - Write unit tests to ensure accurate calculation of condo fees under various scenarios.	2/29/2024	3/21/2024	Sprint_3	- The system accurately calculates condo fees for each unit based on the provided rates and unit specifics. - Unit owners can view the calculated condo fees for their units. - The calculations update in real-time following any changes to unit details or fee rates.
28	US12: Record Operational B	As Brian (condo management compnay manager) , I want to record the operational budget, including collected condo fees and enter costs for each operation or activity, so that I can manage financial resources effectively and maintain a clear overview of the property's financial health.	User Story	L	12	P1	- Expand the financial module to include functionality for recording operational budgets and costs. - Design and implement a MongoDB schema to store operational budgets, collected condo fees, and specific costs for activities and maintenance operations. - Create Django RESTful API endpoints for entering and updating these financial details. - Develop a React interface for condo management companies to input and manage operational budgets and costs. - Ensure data integrity with validation checks for budget and cost entries.	2/29/2024	3/21/2024	Sprint_3	- Condo management companies can enter and update operational budgets and specific costs. - The system correctly records and associates these financial details with the respective properties. - The interface provides a clear overview and easy management of financial data.
29	US13: Generate Annual Fina	As a Brian (condo management compnay manager) , I want the financial system to generate annual reports summarizing the collected condo fees and operational costs, so that I can provide comprehensive financial insights to condo owners and stakeholders.	User Story	M	9	P2	- Develop a reporting tool within the financial system to generate annual financial reports. - The tool should aggregate collected condo fees and operational costs for each property. - Design a Django RESTful API endpoint to trigger report generation and retrieval. - Create a React component for condo management companies to generate and view these reports. - Ensure the report includes detailed financial insights and is formatted for clarity and professionalism.	2/29/2024	3/21/2024	Sprint_3	- Condo management companies can generate annual financial reports for each property. - The reports accurately summarize collected condo fees and operational costs. - The generated reports are accessible and provide valuable financial insights to condo management companies and stakeholders. - Reports are presented in a clear, professional format that is easy to understand.
30	US14: Set Up Common Faci	As Brian (condo management compnay manager) , I want to set up and manage common facilities that require reservations, such as a sky lounge and spa fitness, to organize and offer these amenities efficiently to condo owners and rental users.	User Story	L	12	P2	- Design and implement a MongoDB schema to store information about common facilities, including names, descriptions, and available reservation time slots. - Create Django RESTful API endpoints for managing (creating, updating, deleting) the details of these facilities. - Develop a React component for condo management companies to manage the setup and details of common facilities. - Implement input validation to ensure the integrity of the data being entered for each facility. - Establish access control measures to ensure that only authorized personnel can manage these details.	3/22/2024	4/11/2024	Sprint_4	- Condo management companies can easily add, modify, and remove information about common facilities within the platform. - Details such as the name, description, and reservation availability of facilities are accurately captured and stored. - Management of these facilities is secured and limited to authorized personnel.
31	US15: Calendar-like Interfac	As Brian (condo management compnay manager) , I want to offer a calendar-like interface for the reservation system that displays the real-time availability of common facilities, enabling condo owners and rental users to make informed reservations conveniently.	User Story	S	6	P2	- Develop a calendar-like interface in React that displays the availability of common facilities. - Ensure the interface is dynamically linked with the Django backend to fetch and display real-time availability data. - Create an intuitive process within the interface for users to make reservations by selecting available time slots. - Implement real-time updates to the interface to reflect the current availability status of facilities, accounting for new bookings or cancellations.	3/22/2024	4/11/2024	Sprint_4	- The reservation system features a user-friendly calendar-like interface that shows available time slots for each facility. - Condo owners and rental users can easily make reservations through this interface, with immediate feedback on the booking status. - Availability data is updated in real-time, ensuring all users see the current status of facility bookings.
13	US16: Display Real-time Ava	As Brian (condo management compnay manager) , I want the reservation system to show real-time availability of common facilities, so that condo owners and rental users can accurately see which facilities are available for booking at any given time.	User Story	L	12	P2	- Implement functionality in the Django backend to track and update the availability status of common facilities in real-time. - Develop an API endpoint that provides the current availability status of all facilities. - Integrate the real-time availability data into the React calendar-like interface for reservations. - Ensure the system handles concurrent bookings accurately, preventing double bookings.	3/22/2024	4/11/2024	Sprint_4	- The reservation interface accurately displays the real-time availability of all common facilities. - Updates to facility availability are reflected immediately in the reservation system, preventing overbooking and ensuring accurate information for users. - Users experience no delay in seeing the current availability status of facilities when making reservations.

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria
14	US17: Manage Reservations	As Brian (condo management compnay manager) , I want the reservation system to manage bookings on a first-come-first-serve basis and automatically update the facility's availability once a booking is made, to ensure fair access to facilities for all condo owners and rental users.	User Story	M	9	P1	- Design logic in the Django backend to process reservations based on a first-come-first-serve principle. - Ensure that once a reservation is made, the system immediately marks the time slot as unavailable to other users. - Update the React reservation interface to reflect changes in availability in real-time following a booking. - Implement checks to prevent double booking of any facility's time slot.	3/22/2024	4/11/2024	Sprint_4	- Reservations are processed strictly on a first-come-first-serve basis, with immediate feedback provided to the user upon booking. - The system accurately updates and displays the unavailability of booked time slots, preventing further reservations for those slots. - Users can see their bookings confirmed in real-time, with no possibility of double booking a single time slot.
16	US18: Design Role-Based A	As Brian (condo management compnay manager) , I want to design and implement a role-based access control (RBAC) system within the condo management platform, so that we can define distinct roles for employees with specific permissions tailored to their job functions.	User Story	L	12	P1	- Design an RBAC model to support various roles, including manager, operations personnel, and finance personnel, with specific permissions for each role. - Implement the RBAC system in Django, ensuring it integrates with the existing authentication mechanisms for secure access control. - Create a MongoDB schema to store role definitions, permissions, and employee-role assignments. - Develop API endpoints in Django to manage roles and permissions, including creating, updating, and deleting roles.	3/22/2024	4/11/2024	Sprint_4	- The platform supports distinct roles with specific permissions tailored to different areas of the system. - Roles and permissions can be managed through Django API endpoints, with changes reflected in real-time. - The RBAC system is integrated with the platform's authentication mechanisms to enforce access controls securely.
17	US19: Predefined Roles and	As Brian (condo management compnay manager) I want to create predefined roles with specific permissions associated with various system functionalities, enabling a structured access control system tailored to our operational needs.	User Story	S	6	P2	- Define a set of common roles within the platform, including manager, operations personnel, and finance personnel, with predefined permissions for each. - Populate the MongoDB database with these predefined roles and their associated permissions. - Develop functionality in the Django backend to allow for the customization of roles and permissions as needed by the condo management company. - Implement validation mechanisms to ensure roles and permissions are correctly defined and do not conflict.	3/22/2024	4/11/2024	Sprint_4	- Common roles are predefined in the system, each with access rights tailored to specific job functions. - The system allows for customization of roles and permissions, enabling condo management companies to adapt the access control system to their specific needs. - Validations prevent the assignment of conflicting permissions, ensuring data integrity and security.
32	US20: Manage Employee R	As Brian (condo management compnay manager) , I want to develop an interface for assigning and managing employee roles, allowing customization of access based on individual responsibilities, to ensure efficient and secure operation of the condo management system.	User Story	L	12	P2	- Design and implement a React component for managing employee roles within the condo management platform. - Integrate this component with the Django backend to fetch available roles and permissions and to assign roles to employees. - Create a user-friendly UI that allows for easy assignment of roles and modification of permissions for individual employees. - Ensure the interface supports adding new employees to the system and changing their roles as necessary.	4/12/2024	5/1/2024	Sprint_5	- Condo management companies have an intuitive interface to assign and manage employee roles. - The interface allows for easy customization of employee access rights, including adding new employees and changing roles. - Changes made through the interface are accurately reflected in the system, with immediate updates to access controls based on role assignments.
33	US21: Secure and Scalable	As a Brian (condo management compnay manager) , I want to ensure that our RBAC system is secure and scalable, capable of adapting to evolving needs and handling varying numbers of employees and roles efficiently.	User Story	M	9	P2	- Integrate the RBAC system with the existing authentication mechanisms to enforce secure access control based on assigned roles. - Implement logging and auditing features to track changes to roles and permissions, ensuring compliance and facilitating reviews. - Test the system for scalability, ensuring it can handle an increasing number of employees and roles as the company grows. - Review and update security protocols to protect against unauthorized access and ensure data privacy.	4/12/2024	5/1/2024	Sprint_5	- The RBAC system is tightly integrated with authentication mechanisms, ensuring secure access based on roles. - Changes to roles and permissions are logged, allowing for audits and compliance checks. - The system demonstrates scalability, handling a growing number of employees and roles without performance degradation. - Security reviews confirm the system is protected against unauthorized access, ensuring employee and company data remain secure.
18	US22: owner can submit req	As Eric (owner), I want to submit various requests through the system, so that they can be addressed promptly by the appropriate employee	User Story	L	12	P2	- Extend the MongoDB schema to include request types and details. - Implement API endpoints in Django for request submission and assignment based on request type. - Design and develop a request submission form in React. - Create a system in Django to automatically assign requests to employees based on their roles and the type of request. - Update the role-based access control to include permissions for managing different types of requests.	4/12/2024	5/1/2024	Sprint_5	- Ensure that a condo owner can submit different types of requests through the form (moving in/out (date for reserving elevators), intercom changes, requesting access (fobs, keys), reporting a violation, reporting deficiency found in common areas, or asking a question.). - Verify that each submitted request is automatically assigned to the correct employee role. - Confirm that employees can see and respond to assigned requests according to their roles.
19	US 23: employee (Paul) accessing assigned requests from condo owners	As Paul (employee), I want to view and manage requests assigned to me, so that I can efficiently address condo owners' needs.	User Story	M	9	P2	- Develop a dashboard in React for employees to view their assigned requests. - Implement filtering and sorting of requests in the dashboard based on type, date, and status. - Create Django views and API endpoints for updating the status of requests. - Integrate notification functionality in Django to alert employees of new assignments. - Adjust MongoDB schema to track request status updates and notifications.	4/12/2024	5/1/2024	Sprint_5	- Check that employees can access their dashboard and view requests assigned to them. - Verify that employees can update the status of requests and those updates are visible to the condo owner. - Ensure that employees receive notifications for new requests assigned to them, respecting the role-based access controls previously set up.
20	US24: a user notification pag	As Eric (owner), I want to view a notification page for updates on submitted or assigned requests, so that I can stay informed about the latest activities.	User Story	L	12	P2	- Design and implement a notifications schema in MongoDB to store notification data related to user activities. - Develop a notification system in Django that triggers updates for activities such as new requests, status changes, and assignments. - Create a React component for the notification page that displays a list of notifications, sorted by the most recent. - Integrate the React notification page with Django backend to fetch and display notifications. - Update the role-based access control system to ensure users only receive notifications relevant to their role and activities.	4/12/2024	5/1/2024	Sprint_5	- Verify that users can access the notification page and see a list of recent activities related to their submitted or assigned requests. - Ensure that notifications are accurately reflecting the activity's status, such as "Request Assigned," "Status Updated," or "New Request Submitted." - Confirm that the notification list is updated in real-time or near-real-time as activities occur. - Check that the notifications displayed are appropriate to the user's role, with condo owners seeing updates on their requests and employees seeing notifications about assignments. - Test the functionality where users can clear notifications or mark them as read, and these changes persist upon the next login.

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria

Issue #	Title	Description	Labels	Size	Estimate	Priority	Tasks	Start Date	End Date	iteration	Acceptance Criteria