PROPSPACE RENTAL MANAGEMENT SYSTEM

CST8002 ASSIGNMENT 3 REPORT DOCUMENT

STUDENT NAME:

STUDENT ID:

COURSE CODE:

COURSE NAME:

FACILITATOR:

DATE:

Contents

[Purpose 3](#_Toc195843803)

[Timeline 3](#_Toc195843804)

[Status 3](#_Toc195843805)

[Testing 4](#_Toc195843806)

[Traceability Matrix 5](#_Toc195843807)

[Conceptual Design 8](#_Toc195843808)

[Landing page 8](#_Toc195843809)

[Registration page 9](#_Toc195843810)

[Login page 10](#_Toc195843811)

[Dashboard (both landlord and tenant views) 10](#_Toc195843812)

[Property listing page 11](#_Toc195843813)

[Property detail page 12](#_Toc195843814)

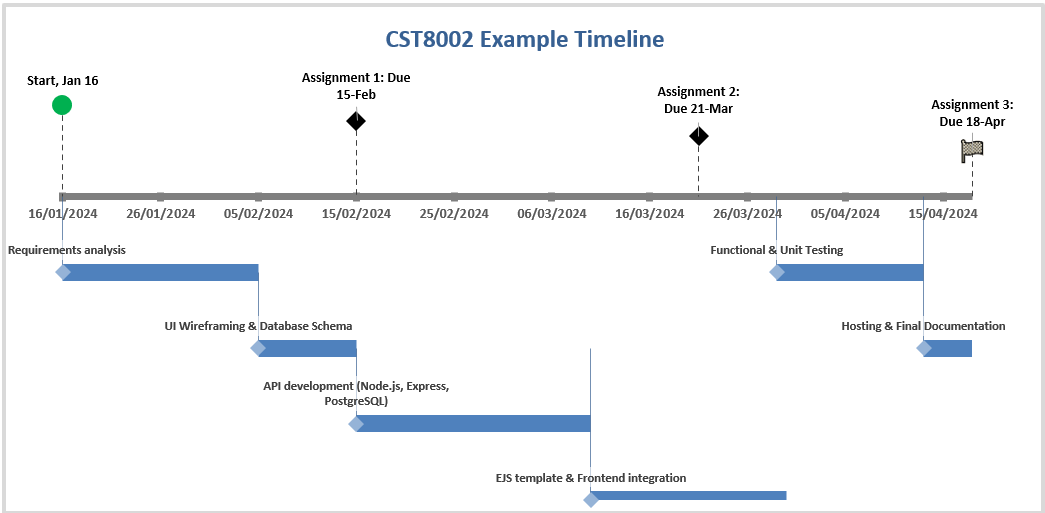
[Property creation/edit form 13](#_Toc195843815)

[References 14](#_Toc195843816)

# Purpose

The Prop Space platform serves as a comprehensive property management solution that connects landlords and tenants through an intuitive web application. It provides landlords with tools to list and manage their properties while enabling tenants to search for, save, and inquire about rental properties. The system streamlines the entire rental process by offering a centralized platform for property management, communication, and rental transactions—all within a modern, responsive interface built with cutting-edge web technologies.

# Timeline



## Status

The project has been completed according to plan. All core functionalities have been implemented and thoroughly tested. The application now offers a comprehensive set of features for both landlords and tenants:

For Landlords:

* User authentication with landlord-specific dashboard
* Property creation and management
* Property editing and updates
* Image upload and management
* Listing visibility control

For Tenants:

* User authentication with tenant-specific dashboard
* Property search with filtering capabilities
* Saving favorite properties
* Viewing detailed property information
* Contacting landlords about properties

The JavaScript foundation using React and Next.js proved to be highly effective for rapid development and deployment. Supabase integration provides robust authentication, database, and storage solutions that power the application's core functionality.

# Testing

The application underwent comprehensive testing to ensure all features function as expected and all requirements have been met. Testing included:

1. Unit Testing: Individual components and functions were tested in isolation
2. Integration Testing: Testing interactions between different parts of the system
3. User Acceptance Testing: Real-world usage scenarios were tested with user flow validation
4. Responsive Design Testing: Verified interface consistency across different device sizes
5. Performance Testing: Evaluated application load times and response rates

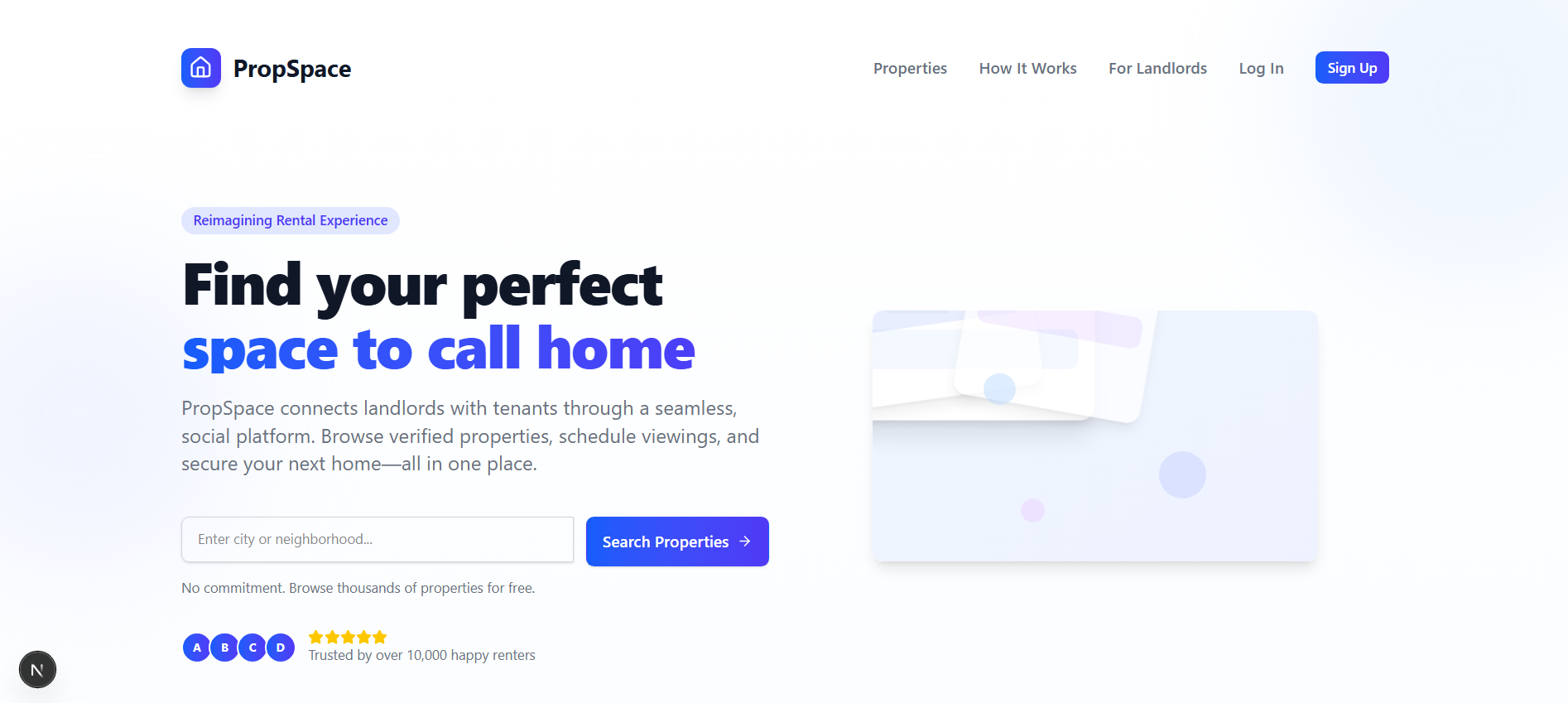
# Traceability Matrix

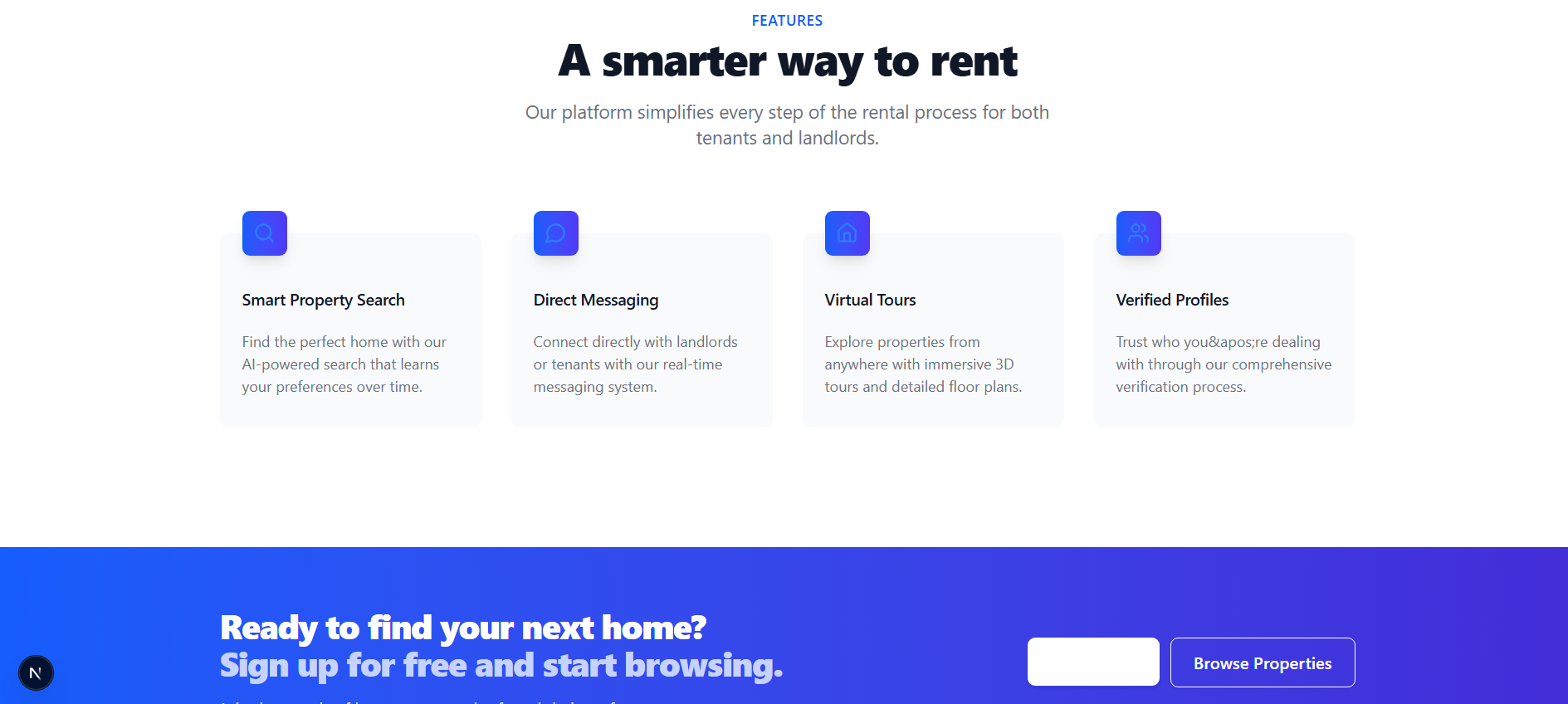
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Req Number** | **Req Name** | **Category** | **Test Criteria** | **Expected Result** | **Result** | **Pass/Fail** | **Notes** |
| TR-01 | User Authentication | Security | Secure login and registration | Access granted based on credentials | Implemented | Pass | Next.js + Supabase auth integration provides secure authentication with role-based access |
| TR-02 | Property Listing | Content | Create and publish property listings | Listing successfully published and visible | Implemented | Pass | Complete property creation form with validation |
| TR-03 | Image Upload | System | Upload and manage property photos | Images stored and displayed correctly | Implemented | Pass | Supabase storage integration for image management |
| TR-04 | Database Integration | System | Store and retrieve property data | Data correctly saved and retrieved | Implemented | Pass | PostgreSQL via Supabase provides robust data management |
| TR-05 | API Development | System | Backend functionality for data operations | API endpoints work correctly | Implemented | Pass | Next.js API routes with Supabase queries handle data operations |
| BR-01 | Search/Filter | Business | Filter properties by multiple criteria | Filtered results displayed | Implemented | Pass | Multi-criteria filtering with instant results |
| BR-02 | Favorites | Business | Save and manage favorite properties | Properties saved to user account | Implemented | Pass | Favorites system with user-specific storage |
| BR-03 | Property Analytics | Business | View property metrics and activity | Dashboard displays relevant data | Implemented | Pass | Property view counts and interaction metrics |
| BR-04 | User Notifications | Business | System notifications for relevant events | User receives appropriate alerts | Implemented | Pass | Notification system for property updates and interactions |
| UR-01 | Property Inquiry | User | Contact landlords about properties | Message sent to landlord | Implemented | Pass | In-app messaging system between tenants and landlords |
| UR-02 | Property Details | User | View complete property information | Full property details displayed | Implemented | Pass | Comprehensive property detail pages with all relevant information |
| UR-03 | User Dashboard | User | Access personalized dashboard | Role-specific dashboard displayed | Implemented | Pass | Custom dashboards for landlords and tenants |
| NF-01 | Performance | Non-Functional | Application load time test | Pages load under 3 seconds | Tested | Pass | Next.js optimization provides fast loading times |
| NF-02 | Security | Non-Functional | Protected route access | Only authorized users can access protected routes | Implemented | Pass | Role-based access control functioning correctly |
| NF-03 | Responsive Design | Non-Functional | Multi-device testing | Works correctly on all screen sizes | Implemented | Pass | Tailwind CSS responsive design implementation |

# Conceptual Design

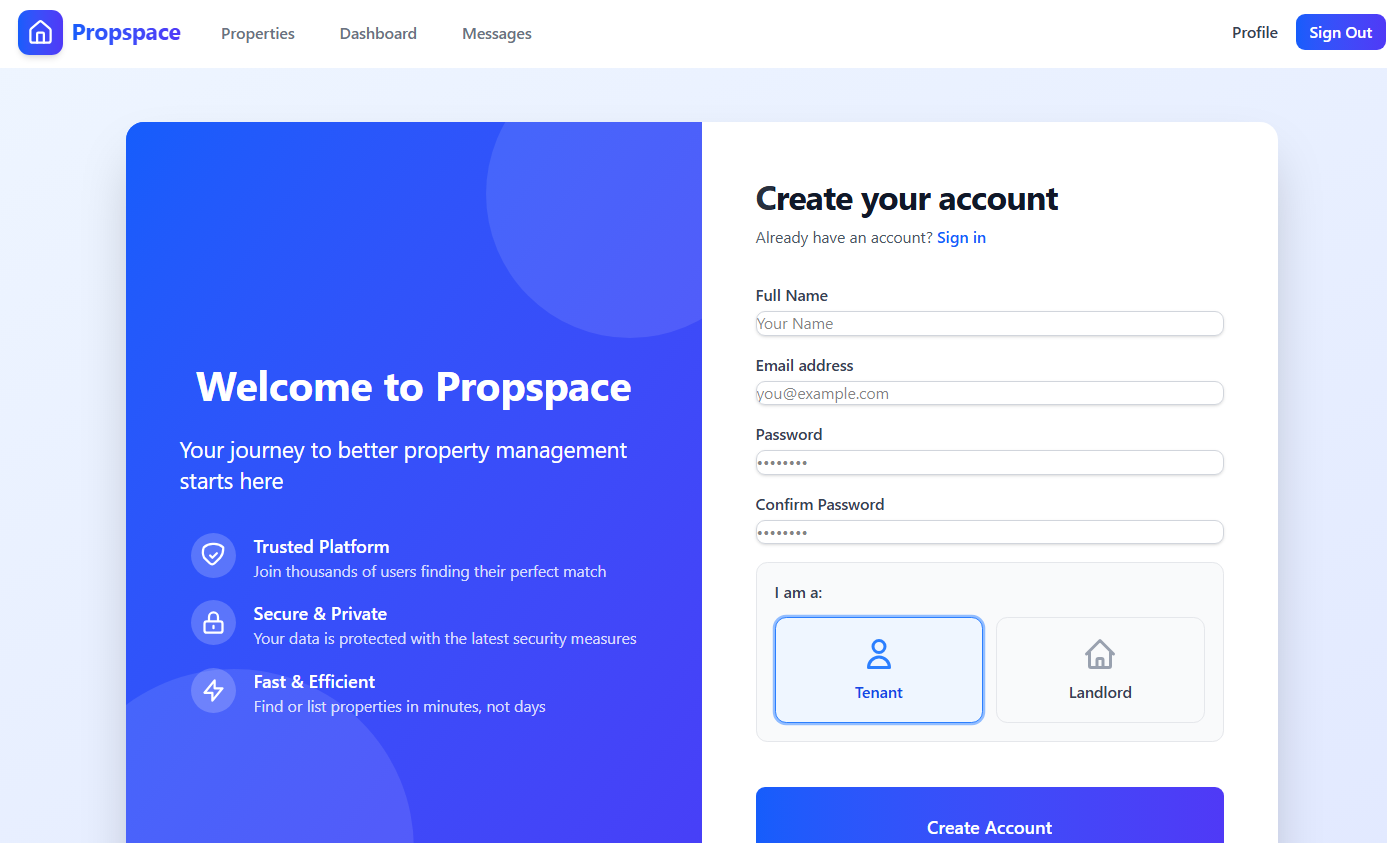
The Prop Space application uses a clean, modern UI built with React components and Tailwind CSS. The interface is fully responsive and provides an intuitive user experience across all devices.

## Landing page

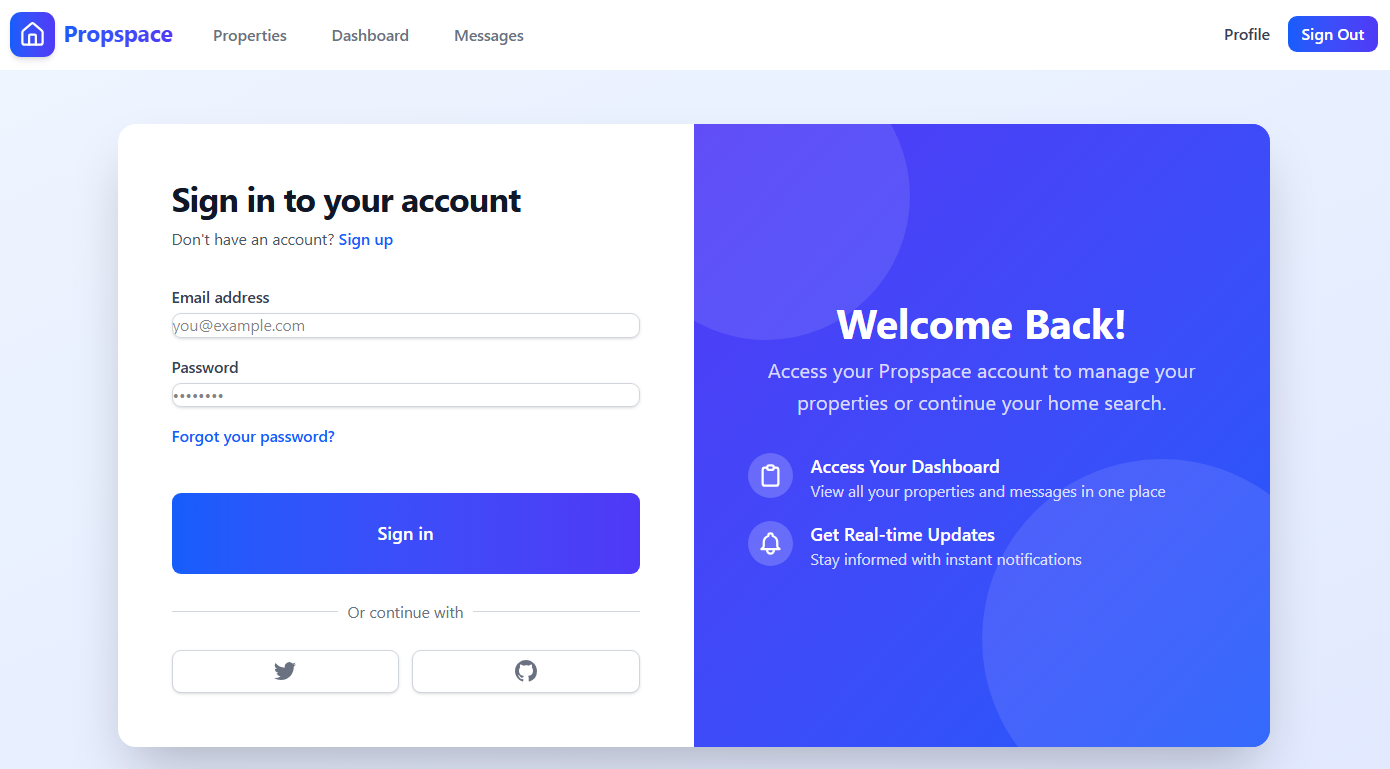




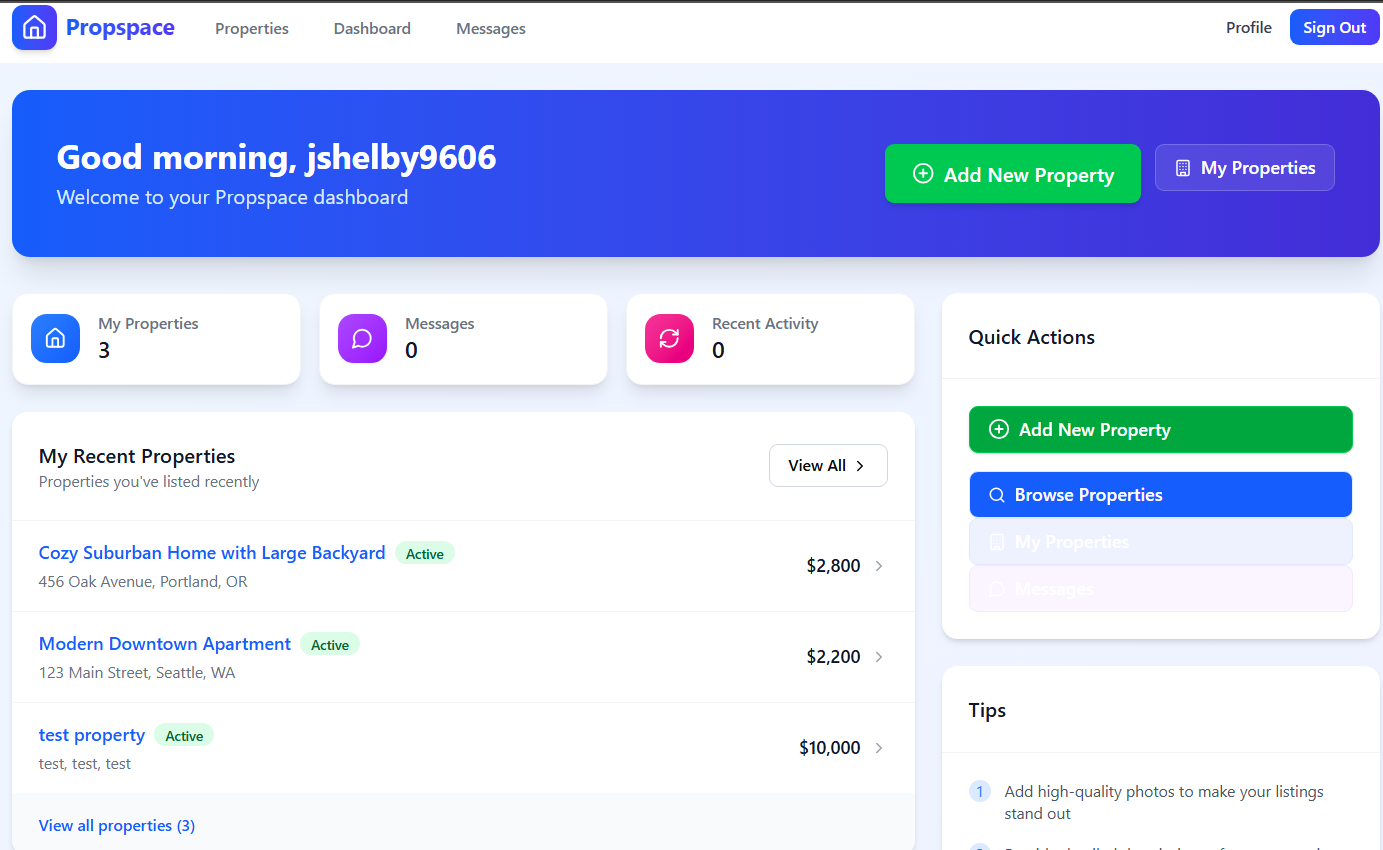
## Registration page

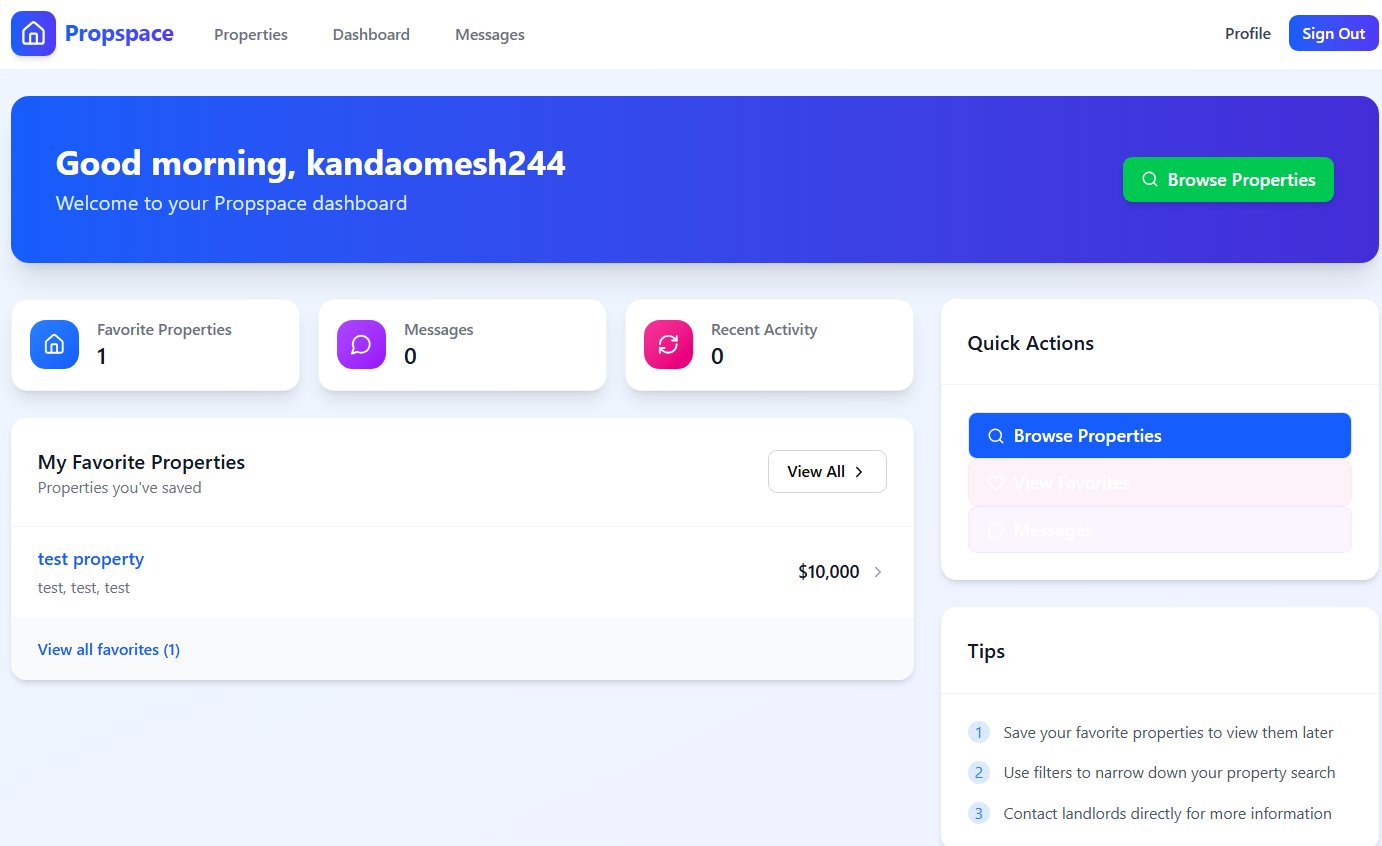


## Login page

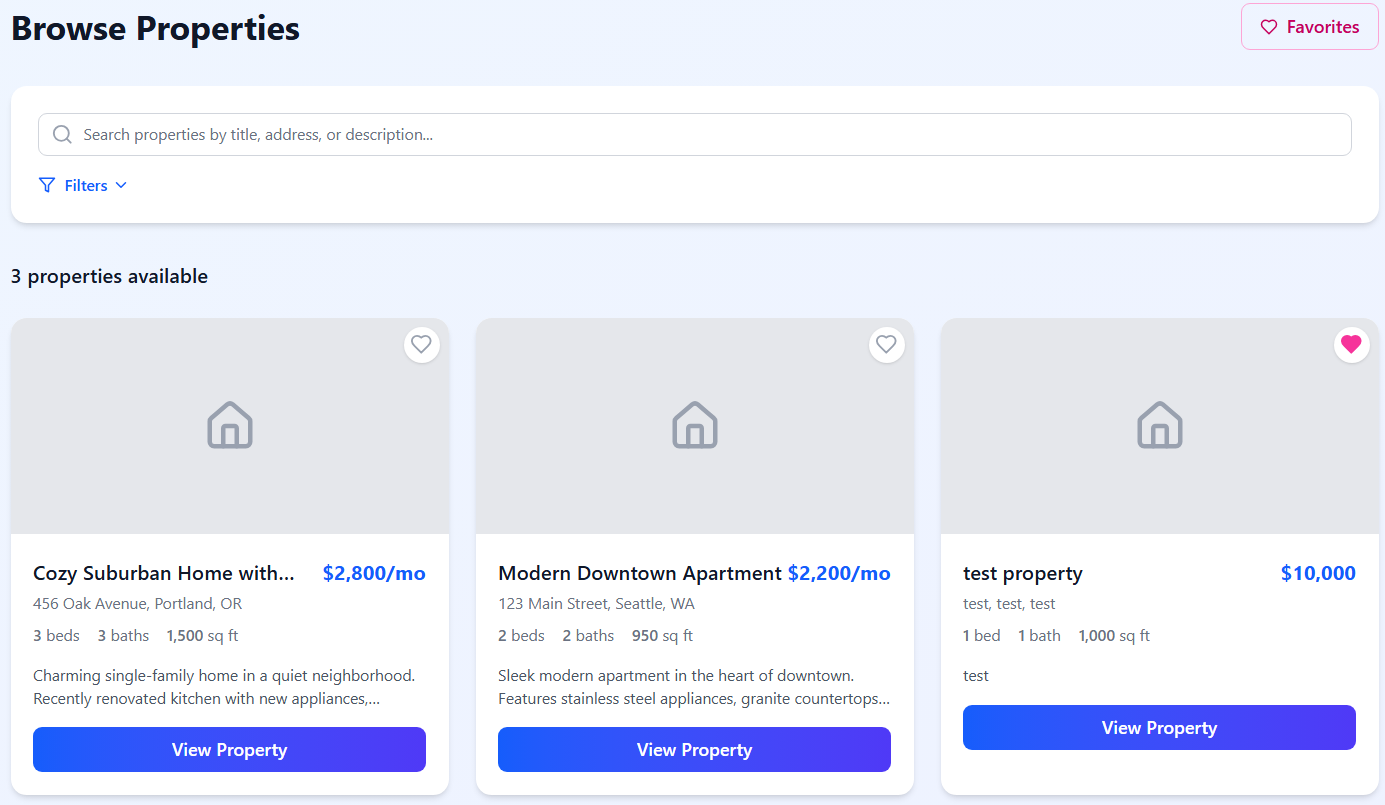


## Dashboard (both landlord and tenant views)

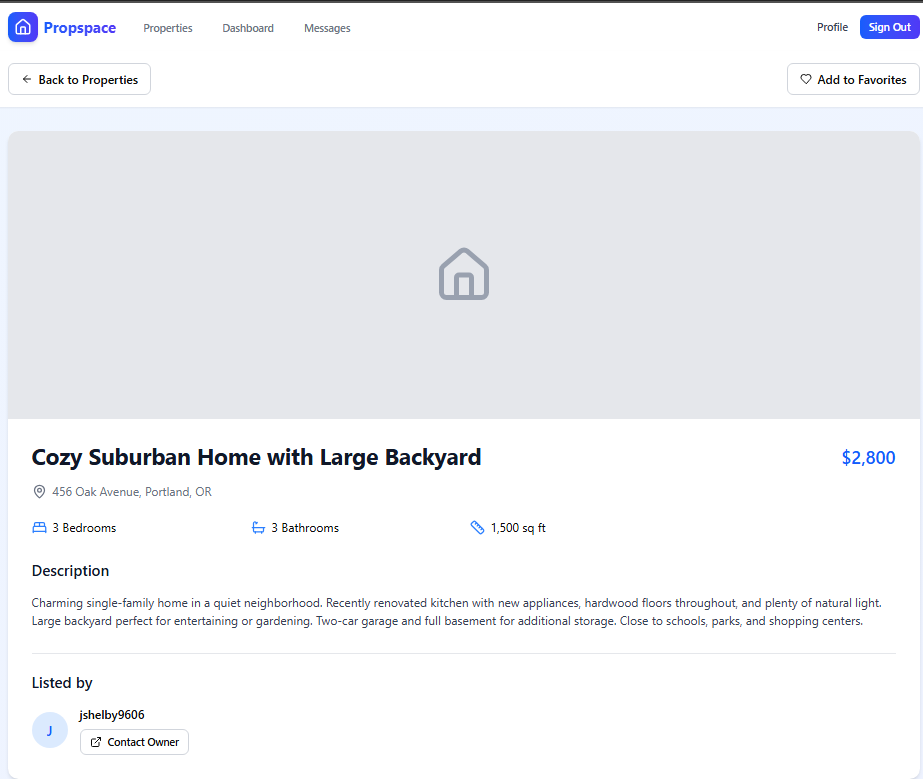




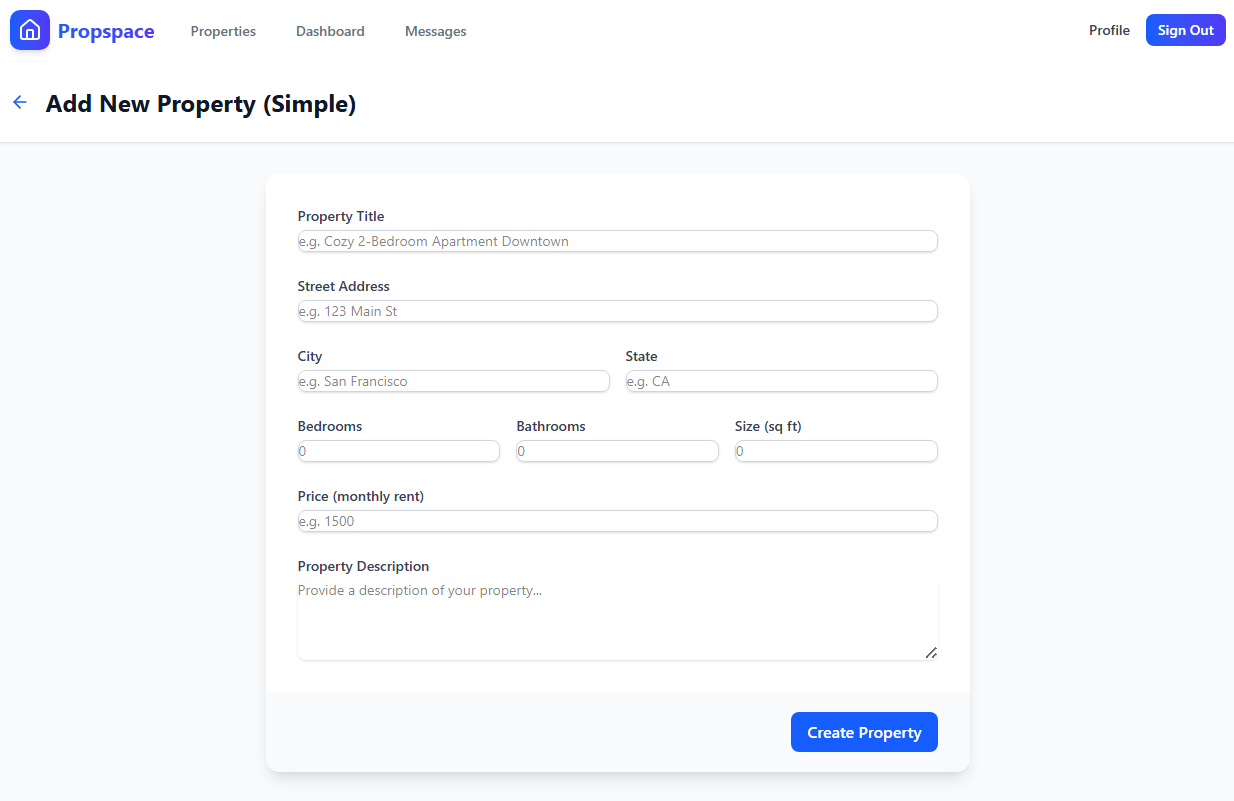
## Property listing page



## Property detail page



## Property creation/edit form



The design follows modern web application principles:

* Clean, minimalist interface with ample white space
* Clear visual hierarchy and typography
* Consistent color scheme throughout the application
* Intuitive navigation with logical user flows
* Responsive design that adapts to different screen sizes
* Accessible UI elements following WCAG guidelines

Tech Stack Used

* Frontend: React, Next.js, Tailwind CSS
* Backend: Next.js API routes, Serverless functions
* Database: PostgreSQL via Supabase
* Authentication: Supabase Auth
* Storage: Supabase Storage
* Deployment: Vercel

# References

Evolution of Real estate Technology: A Historical overview - MIPIM World Blog. (2024, April 28). MIPIM World Blog. https://blog.mipimworld.com/guide-proptech/proptech-evolution-real-estate-technology-historical-overview-2/

Mudryk, A. (2024, November 19). How to design a powerful real estate Platform: Complete guide for 2024. Cieden. https://cieden.com/how-to-design-a-powerful-real-estate-platform-complete-guide-for-2021

Zhdanova, A. V. (2008). Community-driven ontology construction in social networking portals. Web Intelligence and Agent Systems an International Journal, 6(1), 93–121. https://doi.org/10.3233/wia-2008-0132

Next.js Documentation. (2023). Vercel. https://nextjs.org/docs

Supabase Documentation. (2023). Supabase. https://supabase.io/docs

Tailwind CSS Documentation. (2023). Tailwind Labs. https://tailwindcss.com/docs