Al-Najah National University

FACULTY OF ENGINEERING AND INFORMATION TECHNOLOGY

Computer Engineering Department

Software Graduation Project

SHAQATI

Students:

Abd albaset Rajab	12042362
Ahmad hanani	12043081

Supervised by:

Dr. Khaled daud

Acknowledgements

First and foremost, we extend our deepest gratitude to Almighty God, whose divine guidance, strength, and mercy have been our source of perseverance throughout this journey. Without His blessings, this achievement would not have been possible.

To our beloved families, especially our parents, we express our sincere thanks for their endless love, sacrifices, and unwavering belief in us. Their encouragement was the light that guided us through every challenge.

With heartfelt admiration, we offer our special thanks to Dr. Khaled Daud, a true pillar of inspiration and excellence throughout our academic journey. His profound knowledge, passionate teaching, and unwavering dedication have left an indelible mark on us. He was not only an educator, but also a mentor who guided us toward excellence and critical thinking—and for that, we are forever grateful.

We would also like to extend our sincere appreciation to all the professors of the Computer Engineering Department at An-Najah National University for equipping us with knowledge and values that extend far beyond the classroom. Their dedication has shaped the engineers we have become.

Last but not least, we would like to thank our dear friends and classmates. Their support, shared struggles, and joyful moments made these five years unforgettable. This journey was never walked alone—they made it meaningful, memorable, and worth every challenge.

Together, we reached the summit. Thank you.

Disclaimer Statement

This report was written by students at the Abd albaset Rajab& Ahmad hanani Engineering Department, Faculty of Engineering, An-Najah National University. It has not been altered or corrected, other than editorial corrections, as a result of assessment and it may contain language as well as content errors. The views expressed in it together with any outcomes and recommendations are solely those of the students. An-Najah National University accepts no responsibility or liability for the consequences of this report being used for a purpose other than the purpose for which it was commissioned.

Table of Contents

Table of Contents 4

List of Figures 7

1 Chapter 1 10

1.1 Introduction 10

1.1.1 Problem 10

1.1.2 Objective 10

1.1.3 Scope of the Work 11

1.1.4 Importance 11

2 Chapter 2 12

2.1 Theoretical Background and Previous Work 12

2.1.1 Practical Challenges and Limitations 12

2.1.2 Adopted Development Standards 12

2.1.2.1 MVC Architecture 12

3 Chapter 3 14

3.1 Literature review 14

4 Chapter 4 15

4.1.1 Methodology 15

4.1.2 Tools, Programming Languages, APIs Technologies 15

4.1.3 client Side (User & Driver Interfaces): 15

4.1.4 Website Side for admin 16

4 1.5 Server Side: 16

4.1.6 IDEs and Code Editors 17

4.1.7 Database Structure 18

4.1.8 Database - NoSQL 18

4.1.9 Features of The Application 19

4.1.10 Implementation 19

4.1.11 Mobile Application 19

1. **landlord dashboard** 19

2. Charts and Revenue Analytics Page 20

5. **Landlord Properties Management Screens** 23

4.1.12 Database and API Server 85

5 Chapter 5 86

5.1 Result and Analysis 86

6 Chapter 6 87

6.1 Discussion 87

6.1.1 **Learning Outcomes** 87
6.1.2 **Key Challenges** 87
7 Chapter 7 88
7.1 Conclusion and Recommendation 88
7.1.1 Summary 88
7.1.2 Things We Learned 88
7.1.3 Recommendations 88
7.1.4 Future Work 89
8 Chapter 8 90
8.1 References 90

List of Figures

Figure1 landlord dashboard 19
Figure2 Charts and Revenue Analytics Page 20
Figure3 Landlord Important Notifications Popup 21
Figure4 Landlord Messaging and Chat Screens 22
Figure5 Landlord Properties Management Screens 23
Figure6 Landlord Property Analytics and Performance Dashboard 24
Figure7 Landlord Property Creation and Management Workflow 24
Figure8 Contract Lifecycle, Payments, and Official Lease Document 25
Figure9 Landlord Maintenance & Complaints Management Dashboard 26
Figure10 Payments Dashboard and Transaction Details 27
Figure11 Expenses Management Dashboard, Entry Form, List View, and Analytics Charts 28
Figure12 Security Deposits Tracking, Refund Actions, and Analytics 31
Figure13 Invoices Management, Status Tracking, and Collection Analytics 32
Figure14 User Registration Screen (sign up) 33
Figure15 User Authentication Flow & Reset password 34
Figure16 Email Alert for New Login Detection 35
Figure17 User Not Found - Email Verification Prompt 37
Figure18 Admin Dashboard - System Overview and Analytics 38
Figure19 Admin Navigation Menu 39
Figure20 Admin- User Management Screen 40
Figure21 Statistics Charts 41
Figure22 Admin - Property Management screens 42
Figure23 Admin-Edit Property 43
Figure24 Admin Property Types Management – list view, add type form, and statistics chart 44
Figure25 Admin Property Types – create, edit, and delete property categories 45
Figure26 Admin Property Types – edit category details 46
Figure27 Admin Contract Details – manage lease status, payments, renewals 47
Figure28 Admin Payments – overview, filtering, and approval of rent transactions 48
Figure29 Admin Payments – export options, advanced filters, detailed receipt view, and revenue analytics 50
Figure30 Admin Maintenance & Complaints – request list, actions, analytics 50
Figure31 Admin Reviews – search, filter by rating, sorting, and statistics 51
Figure32 Admin Notifications 52
Figure33 Admin Expenses – overview, detailed expense list, and analytics 53
Figure34 Admin Invoices – overview of invoice totals, detailed list, and analytics 54
Figure35 Admin System Settings 55
Figure36 Tenant home page: search, browse, chat. 56
Figure37 Artificial Intelligence - Investment Tips in Real Estate 57
Figure38 tenants - dashboard 58
Figure39 Contract management and payment tracking 60
Figure40 Payment tracking and history management 68
Figure41 Maintenance request and tracking system 74
Figure42 Expenses tracking and budget management 79
Figure43 Deposit tracking and refund management 81
Figure44 Tenant web pages 82
Figure45 Landlord web pages 83
Figure46 Admin web pages 83

Abstract

This project proposes the design and development of a comprehensive Smart Real Estate Management System, implemented as both a web and mobile application, aimed at transforming the management, rental, and sale of residential properties. The system provides a unified digital platform that seamlessly connects property owners, tenants, and administrators, enabling streamlined, efficient, and transparent execution of all real estate operations.

Property owners can create and manage profiles on the site, as well as add and update listings with all the necessary information, such as location, price, area, photos, amenities, and descriptive data. Applications for rentals are received and processed, requests are approved or denied, contracts are issued and digitally signed, payment transactions are tracked and managed, and maintenance issues are handled effectively thanks to the system. Property owners are guaranteed to stay up to date on important events and outstanding duties pertaining to their properties thanks to automated notifications and reminders.

Tenants can visit the site to search for available apartments with advanced searching and filtering features, read in-depth property descriptions, submit rental applications online, and e-sign leases. Tenants can safely pay rent and utilities online, see payment history, and submit maintenance requests or report problems in real-time through the portal. The site also includes personalized dashboards and notification alerts to enhance user engagement and satisfaction.

The administrative board provides system administrators full authority over every activity on the platform, including user administration, verification of property ownership, complaint management, tracing financial transactions, and creating detailed analytical reports as well as financial reports. These reports assist in evaluating the performance of the platform, tracking the level of engagement among the users, and making sound decisions to optimize operational efficiency.

In order to guarantee scalability, cross-platform compatibility, data privacy, and strong security, the system was created utilizing contemporary online and mobile technologies. In order to provide a completely digital and efficient approach to real estate administration, a focus was focused on developing an intuitive user interface, automating conventional procedures, and cutting down on paperwork. This project offers a clever, effective, and easily accessible solution that improves communication, saves time, and enhances the overall experience for all parties engaged in property management by fusing technological innovation with useful functionality.

1 Chapter 1

1.1 Introduction

1.1.1 Problem

The property management sector in Palestine faces significant challenges due to reliance on manual processes, paper-based documentation, and fragmented communication. Many landlords and administrators manage properties using spreadsheets and emails, leading to data inconsistency, errors, and delays in handling contracts, payments, and maintenance requests.

Communication gaps between tenants and landlords make it difficult to track maintenance issues and receive timely updates, resulting in slow responses and reduced service quality. Contract and payment management are often manual and error-prone, while tenants struggle to find suitable properties due to limited search features and incomplete listings.

Additionally, the absence of real-time notifications and centralized digital systems reduces efficiency and scalability, especially as property portfolios grow. These challenges are further influenced by the unique local context in Palestine, where accessibility and infrastructure limitations must be considered.

This project proposes a Property Management System (PMS) that offers a unified digital platform with automated workflows, centralized data management, real-time communication, and multi-platform accessibility, designed to meet the specific needs of the Palestinian property market.

1.1.2 Objective

The objective of this project is to develop a Property Management System (PMS) that digitizes and streamlines property management processes in Palestine. The system replaces manual and fragmented workflows with a unified digital platform that supports landlords, tenants, and administrators, improving efficiency, accuracy, and accessibility.

The proposed PMS enables centralized data management, real-time communication, and automated handling of maintenance requests, contracts, and payments. It also provides advanced property search features, real-time notifications, secure role-based access, and multi-platform support (web and mobile). Overall, the system aims to enhance tenant satisfaction, increase landlord productivity, reduce operational errors, and deliver a scalable solution tailored to the Palestinian property market.

1.1.3 Scope of the Work

This project focuses on the design and implementation of a Property Management System (PMS) with web and mobile support using Flutter. It includes a Node.js/Express backend with MongoDB, secure authentication, and role-based access for tenants, landlords, and administrators.

The system provides core modules for property listings and search, maintenance request management, digital contracts, invoice and payment tracking, real-time notifications via FCM, and administrative dashboards. It emphasizes essential workflows and responsive user interfaces, while advanced features such as payment gateways and third-party integrations are considered out of scope for future work.

1.1.4 Importance

This project addresses a real need in property management by replacing inefficient manual processes with a unified digital system that reduces delays, errors, and communication gaps. By streamlining property listings, maintenance handling, contracts, and payments, the system improves overall operational efficiency and transparency.

The platform benefits landlords through centralized management and scalability, enhances tenant satisfaction with faster responses and clearer information, and supports administrators with automated workflows and data-driven oversight. At the market level, it promotes professional standards, supports growth, and helps modernize property management in Palestine to better adapt to local challenges.

1 Chapter 2

1.1 Theoretical Background and Previous Work

1.1.1 Practical Challenges and Limitations

Throughout developing this Property Management System, we faced several significant challenges:

1. Time Constraints and Learning Curve: This was our first full mobile and web application. We learned Dart/Flutter and Node.js/MongoDB alongside coursework. Integrating Firebase for notifications and authentication added complexity. Balancing learning, other courses, and project deadlines required extra time. We spent additional hours to understand the technologies and align the system with our vision.
2. Notification System Complexity: Implementing push notifications with FCM across iOS, Android, and Web involved platform-specific setup, token management, and background handling. Ensuring reliable delivery across devices and handling edge cases added development time and complexity.
3. Role-Based Access Control and Security: Implementing secure authentication, role-based permissions (tenant, landlord, admin), and authorization checks across features required careful design to prevent unauthorized access while maintaining usability. This needed iterative refinement.
4. Data Synchronization and State Management: Managing data consistency across multiple screens, real-time updates, and offline handling in Flutter required careful state management and caching strategies, sometimes leading to temporary inconsistencies during development.

These challenges tested our technical and problem-solving skills and improved our collaboration, adaptability, and ability to learn and adjust quickly.

1.1.2 Adopted Development Standards

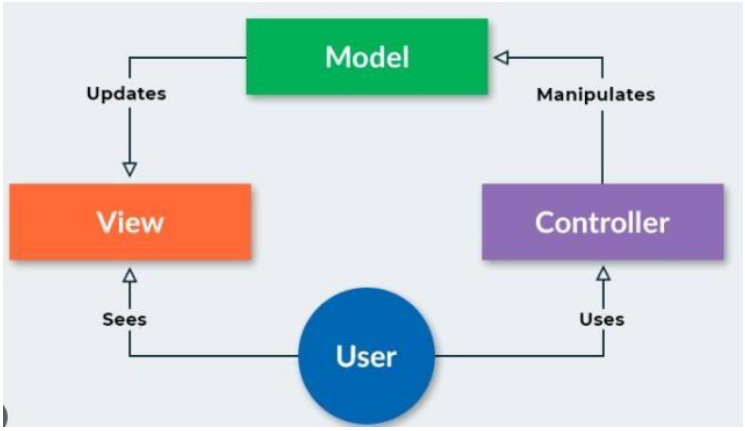
1.1.2.1 MVC Architecture

We use the MVC pattern, extended with a Router component to organize application flow and improve scalability. This helps separate concerns, manage complexity, and speed up development. The main components are:

- **Model:** Represents the data layer, managed through MongoDB. Handles storing and retrieving user, property, maintenance request, contract, invoice, and admin data. Ensures consistency by responding to controller and view requests.

- **View:** The UI for mobile and web, built with Flutter. Allows tenants, landlords, and administrators to interact with the system—browsing properties, submitting maintenance requests, managing contracts, viewing invoices, and accessing dashboards. The UI adapts across platforms.
- **Controller:** The logic layer implemented with Node.js and Express. Processes client requests, applies business logic (authentication, authorization, notifications, data validation), and coordinates between the view and the model.

This separation enables independent development and testing of components, resulting in a more maintainable and scalable codebase.



2 Chapter 3

2.1 Literature review

The development of this Property Management System uses a combination of modern technologies and methodologies from open-source tools, commercial platforms, and real-time communication standards. This review highlights relevant literature and technical resources that shaped the architecture and feature design.

GitHub was used for version control and collaboration . Flutter served as the frontend framework for cross-platform development, supporting responsive UIs for mobile and web from a single codebase. This choice was influenced by Flutter's flexibility in UI design and its ability to target multiple platforms efficiently.

For backend development, Node.js and Express were selected for their non-blocking, event-driven architecture, which supports concurrent requests and real-time features. MongoDB was chosen as the NoSQL database for its flexibility, document-oriented structure, and scalability, suited for property, user, contract, and maintenance data.

Firebase Cloud Messaging (FCM) was integrated for real-time push notifications, enabling instant updates to tenants, landlords, and administrators about maintenance requests, contract reminders, and payment due dates. This improved communication and responsiveness across the system.

For image management, Cloudinary was used for uploading and storing property images and 3D model assets, offering reliable cloud-based storage and optimization. This supported efficient image handling for property listings.

For hosting and deployment, platforms like Render were utilized for backend deployment, streamlining CI/CD and supporting Node.js environments. This enabled reliable backend hosting with minimal configuration.

For authentication and authorization, JSON Web Tokens (JWT) were implemented for secure user authentication and role-based access control, ensuring secure access to system resources. This provided a stateless authentication mechanism suitable for distributed systems.

For 3D model visualization, ModelViewer was explored for displaying 3D property models, though implementation faced performance and compatibility challenges. This supported enhanced property visualization for tenants.

Socket.IO was considered for real-time features like chat and live updates , though the current implementation primarily relies on RESTful APIs and push notifications.
Together, these technologies form the foundation for a scalable, reliable Property Management System that meets modern expectations for efficiency, real-time communication, and cross-platform accessibility.

3 Chapter 4

4.1.1 Methodology

4.1.2 Tools, Programming Languages, APIs Technologies

After evaluating different technologies, a modern and scalable stack was selected for the Property Management System. The frontend is built using Flutter and Dart to support cross-platform web and mobile applications, while the backend uses Node.js with Express.js to provide RESTful APIs. MongoDB with Mongoose is used for flexible data storage, and authentication is handled using JWT and bcryptjs.

Additional services include Firebase Cloud Messaging for real-time notifications and Cloudinary for image storage. Development and deployment rely on Git/GitHub, Postman, VS Code, Android Studio, and cloud hosting platforms such as Render or Heroku. This technology stack ensures scalability, real-time communication, and efficient multi-platform development.

4.1.3 client Side (User & Driver Interfaces):

1. **Design:** The Property Management System mobile and web interfaces are designed to be clear and easy to use. The interface lets tenants, landlords, and administrators navigate features like property browsing, maintenance requests, contract management, invoicing, and notifications. The color scheme balances professionalism and readability.

   Before finalizing the design, we reviewed interfaces from property management platforms and mobile apps. We adapted common UI patterns to fit our goals and local context. We focused on accessibility, responsive layouts, and consistent behavior across Android, iOS, and web using Flutter.

2. **Frameworks:** We chose Flutter to build the application. Flutter is an open-source UI toolkit for mobile (iOS, Android), web, and desktop from a single codebase. This enables a consistent, performant experience across platforms.

Top motivators for using Flutter:

(a) Simple to learn.
(b) Excellent performance.
(c) Reusable widgets.
(d) Quick time to market.
(e) Hot reload for fast iteration.
(f) Suitable for MVP development.
(g) Strong design capabilities.
(h) Lower development costs.
(i) Strong documentation and community.
(j) Good developer experience.

3. **Programming languages:** Flutter uses Dart, which targets web and mobile developers. It supports iOS, Android, and web, and runs on modern browsers, mobile devices, and servers. It goes without saying that we will utilize Dart as we will be utilizing the Flutter framework. There are numerous parallels between Dart and other
programming languages including Java, C, Swift, and Kotlin. Since Java is the most familiar language we know, we found that Dart is very comparable to it. This makes learning a new language easier and more straightforward.

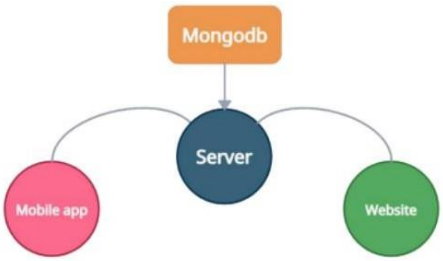
4.1.4 Website Side for admin:

1. **Design:** A dedicated web interface was built for administrators. It enables full system management: creating user accounts (tenants, landlords, admins), sending login credentials via email, monitoring property listings, reviewing and approving property submissions, managing maintenance requests, overseeing contracts and invoices, viewing analytics and reports, managing complaints, and controlling access permissions. The interface prioritizes clarity and efficiency for effective administrative oversight.

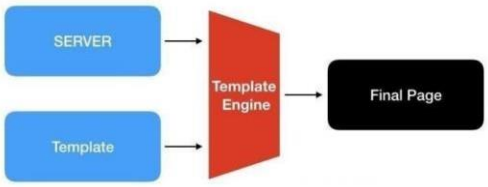
2. **Programming Languages and Frameworks:** The website was built using Flutter, a UI toolkit developed by Google, in combination with the Dart programming language. Dart was chosen for its compatibility with Flutter and its similarity to familiar languages such as Java, which made it easier for the team to learn and adopt. The web application supports modern browsers and is fully responsive, allowing the admin to manage the system effectively across various devices. The backend logic and database connectivity are handled using Node.js and MongoDB, enabling real-time operations, secure role-based access control, and fast data processing.

4.1.5 Server Side:

1. **Frameworks:** On the server side, we used Node.js, a fast, scalable, cross-platform runtime for building high-performance server-side and networking applications. Node.js uses a single-threaded, event-driven, non-blocking I/O model powered by the V8 JavaScript engine, making it suitable for real-time features like maintenance request updates, live notifications, and concurrent property management operations.



To build our API, we used Express.js, which simplifies routing and handles HTTP requests efficiently. Express.js supports modular routing and middleware integration, helping us structure the application logic cleanly and efficiently. This enabled us to implement RESTful endpoints for property management, user authentication, maintenance requests, contract handling, invoice processing, and notification delivery, while maintaining code organization and scalability.



For real-time communication, especially in-app notifications and system updates between tenants, landlords, and administrators, we integrated Firebase Cloud Messaging (FCM). This cloud-hosted service enables real-time push notifications across connected clients, ensuring immediate delivery of updates about maintenance requests, contract status changes, payment reminders, and administrative alerts across platforms. This ensures all stakeholders receive timely notifications about property-related activities and system changes.

2. **Programming Languages:** The backend is written in JavaScript, leveraging compatibility with Node.js and supporting libraries. JavaScript's flexibility supports both frontend and backend work. Its asynchronous capabilities and real-time support suit our needs, including live maintenance request updates, status notifications, property management, contract tracking, and user management.

4.1.6 IDEs and Code Editors

We developed the Property Management System using Visual Studio Code (VS Code), a free, cross-platform editor from Microsoft. VS Code provided syntax highlighting, integrated terminal, Git support, and extensions for Dart, JavaScript, Node.js, and MongoDB, improving our workflow across mobile and web.

- **Version Control:** Since we worked as a team, version control was essential. We used GitHub to manage code versions, track issues, and merge features. We also used Google Drive to share documentation and media assets.

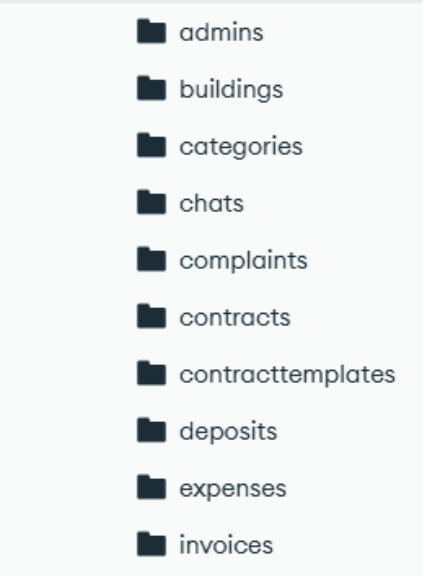
- **API Endpoints Testing:** Our application relies on RESTful APIs, so we verified endpoints before client integration using Postman. Postman helped test GET, POST, PUT, PATCH, and DELETE, debug responses, monitor performance, and ensure reliability across functionalities including authentication, property management, maintenance requests, contract handling, invoice processing, and notifications.

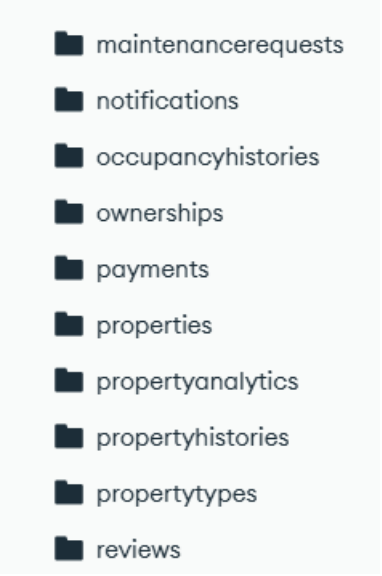
4.1.7 Database Structure

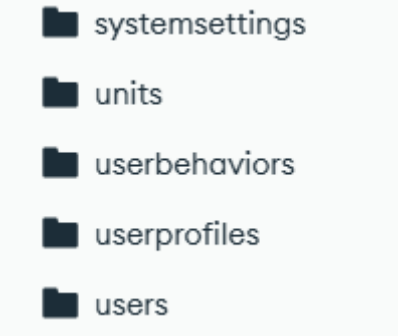
In most applications, the database is essential for reading, editing, and removing data. It gives the application purpose and enables data backup.Our application depends directly on the database for displaying data. We chose MongoDB, a NoSQL document database, because our data model fits a document structure, it's simple to use with our API, and it provides flexibility for evolving property, user, maintenance, contract, and invoice schemas. This makes it well-suited for storing and managing the varied data types in a property management system.

4.1.8 Database- NoSQL:

We chose a non-relational schema for document-based development. Unlike XML, MongoDB stores documents as JSON (JavaScript-based), which fits our backend on Node.js. MongoDB suits Node.js prototyping and keeps data as JSON objects. Unlike SQL databases that require joins across tables, related data is typically in one document. MongoDB is a NoSQL database that handles JSON-like documents with optional schemas, giving flexibility for property, user, maintenance request, contract, and invoice data. This simplifies development and improves performance for our property management workflows.







4.1.9 Features of The Application

4.1.10 Implementation

In this section, we'll dive into the details of each part of the system. All the tools, methods and libraries used are discussed in detail.

The project has three roles: Admin, **Landlord**, and tenant.

Let's start with the Landlord pages.

4.1.11 Mobile Application

landlord dashboard: The SHAQATI mobile application provides landlords with a clean, dashboard-style interface that summarizes the most important information as soon as they log in. At the top of the home screen, the app greets the user by name and displays a prominent notification banner for important events, such as successful activation of a rental contract. Below this, the Quick Actions section offers direct buttons like Add Property and View Reports, enabling the landlord to perform frequent tasks with a single tap. The Overview area then presents key business indicators in separate cards, including Total Revenue, the number of Active Contracts, and the total number of Properties, followed by additional KPIs such as Occupancy Rate, Average Monthly Rent, and a count of open Maintenance requests. At the bottom of the dashboard, a Recent Transactions list shows the latest payments with the tenant's name, date, and amount, helping the landlord quickly review financial activity. Overall, the mobile application is designed to be simple, informative, and action-oriented so that landlords can manage their properties efficiently from their phones.

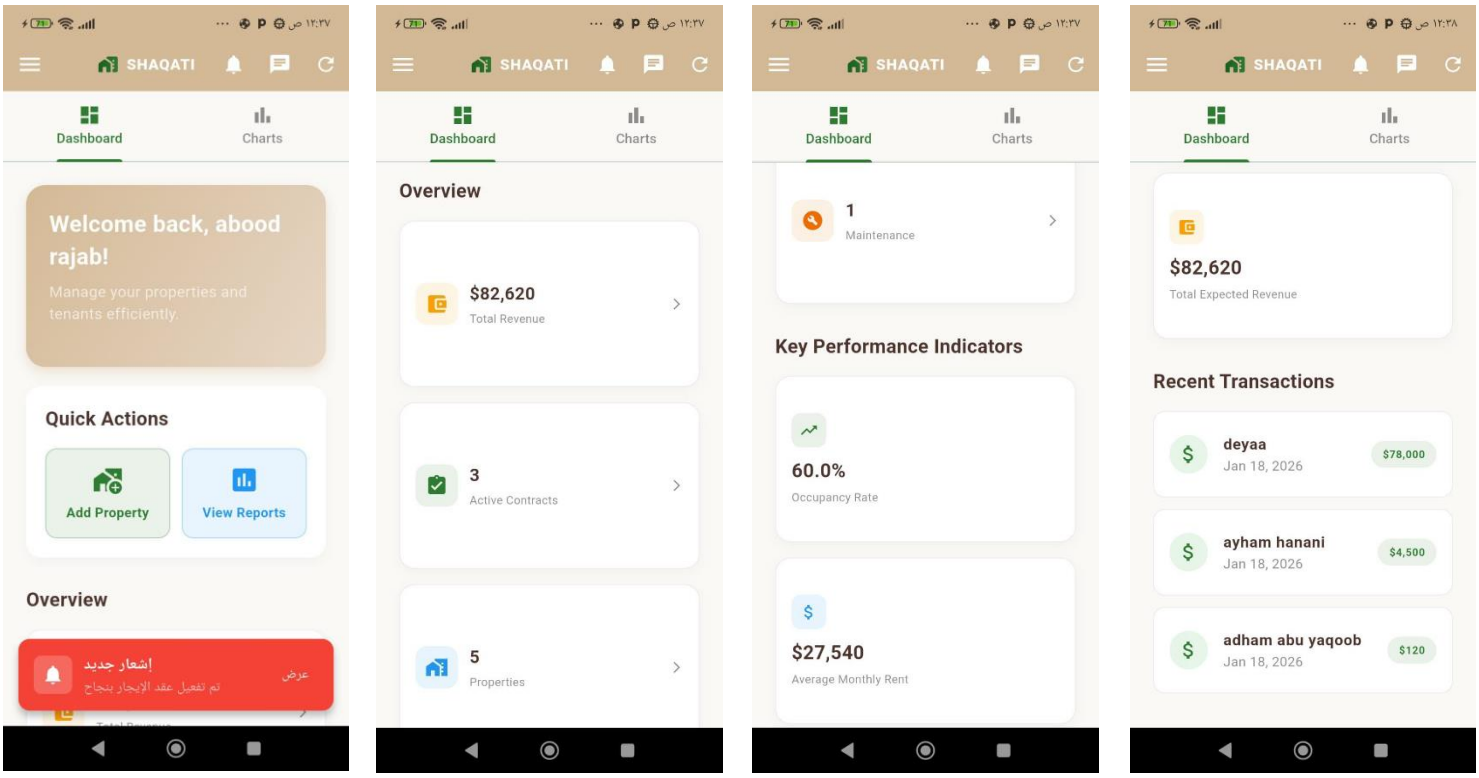


Figure1 landlord dashboard

Charts and Revenue Analytics Page: On the Charts tab, the landlord can see visual analytics about their portfolio.

At the top, the Property Distribution by Status donut chart shows the percentage of properties that are Rented, Available, or Pending, helping the owner understand occupancy at a glance.

Below it, the Monthly Revenue Trend line chart tracks total revenue over recent months, so the landlord can see growth or drops in income over time.

Finally, the Property Comparison by Revenue bar chart compares how much revenue each property generates, making it easy to identify the best- and worst-performing units.

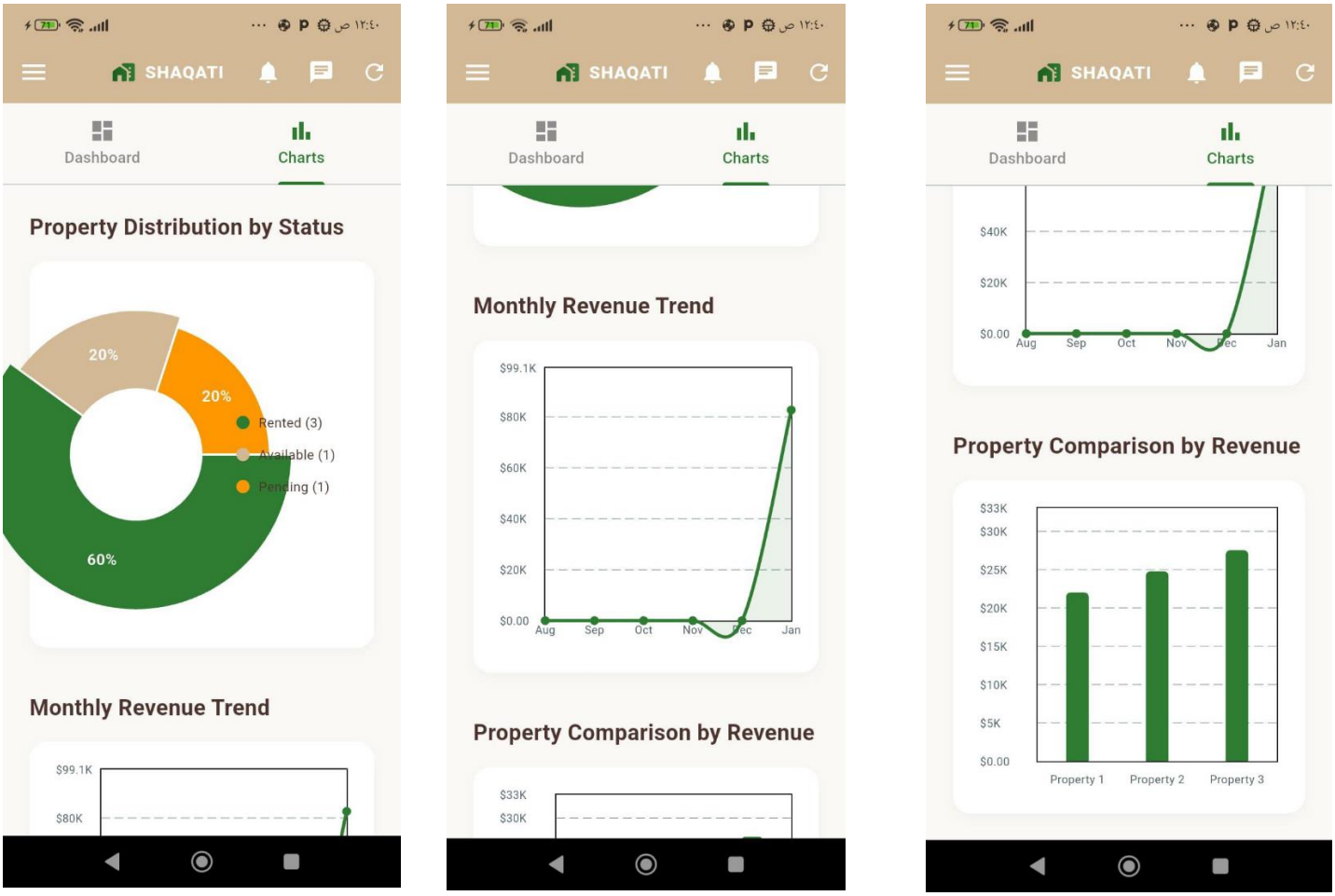


Figure2 Charts and Revenue Analytics Page

This screen shows a popup for the landlord with a list of important notifications.

At the top, the badge displays the total number of unread alerts. Each card contains a bell icon, the word “Notification” in red, the message text (for example, “The rental contract has been activated successfully”), and the time the notification was received. At the bottom, there are two actions: Close to dismiss the popup and Refresh to reload the latest notifications from the server.

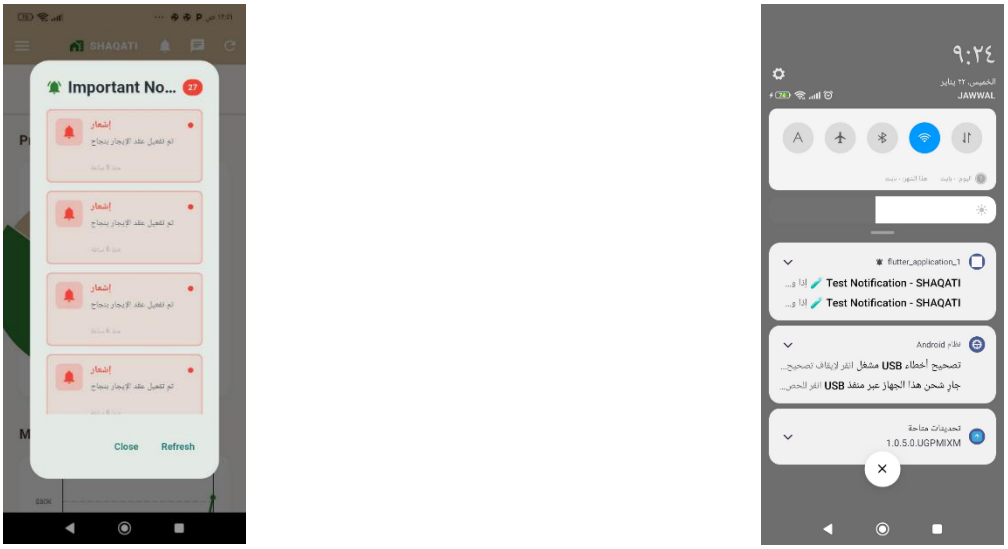


Figure3: Landlord Important Notifications Popup

Landlord Messaging and Chat Screens: The first screen shows the Messages list for the landlord, where all existing conversations are displayed with the user’s name and role (admin, tenant, landlord). Each row acts as a chat preview and can be tapped to open the full conversation. The second screen is the chat detail view with a selected user, and a simple messaging area at the bottom where the landlord can type text, send voice messages, or attach images. This real-time messaging feature allows landlords to communicate directly with tenants and admins inside the app.

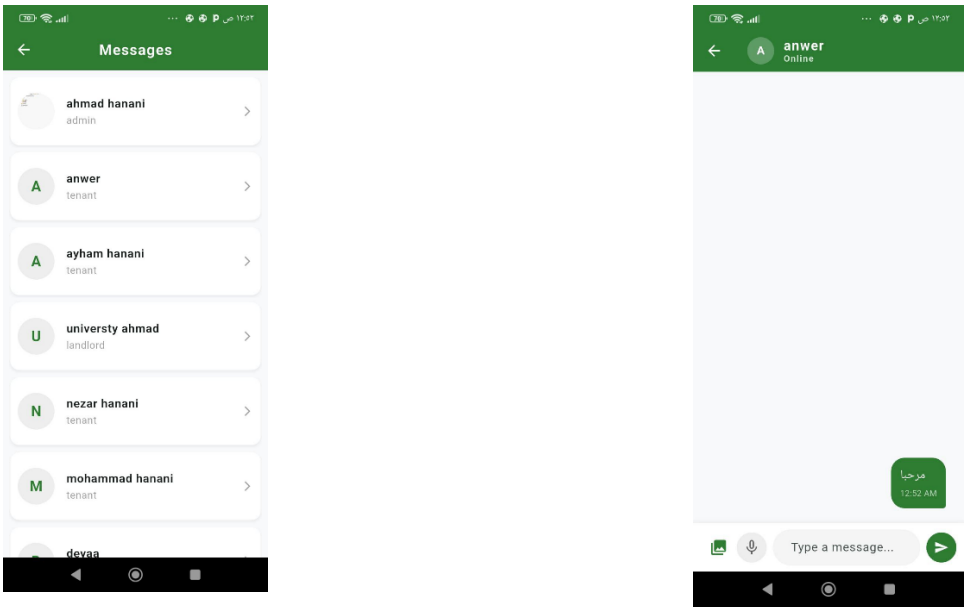


Figure4: Landlord Messaging and Chat Screens

Landlord Properties Management Screens: The first screen displays the owner's navigation menu (sidebar), with the user profile (name and email) at the top and a list of options below, including Control Panel, Properties, Contracts, Maintenance, Payments, Expenses, Deposits, Invoices, and Reports. The second screen displays the "My Properties" list, where each property card shows a photo, address, status badge (Rented, Pending Approval) by the administrator, location, price, and type (Villa, Office, Garden), with the ability to edit and delete. A search bar and filter button are located at the top, and a floating "Add Property" button is located in the lower right. The third screen displays the filter dialog box, which allows the owner to filter properties by status (Available, Rented, Pending), type (Apartment, Office, Villa, Garden), city, and price range, with "Clear All," "Cancel," and "Apply" buttons at the bottom.

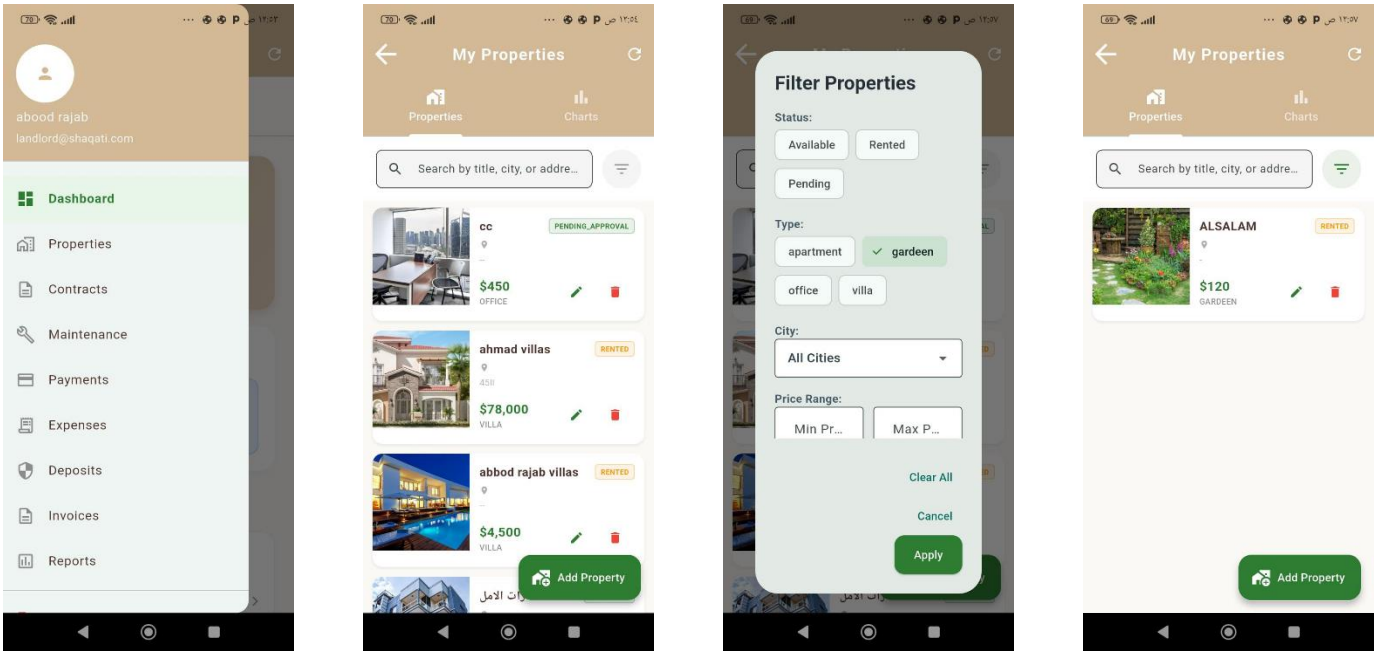


Figure5: Landlord Properties Management Screens

Landlord Property Analytics and Performance Dashboard: These screens show the analytics (Charts) for the landlord inside the My Properties section.
The first screen displays a bar chart labeled "Properties by City/Region", summarizing how many properties the landlord owns in each city, followed by a Property Locations Map section to visualize their positions on the map.
The second screen presents a donut chart "Property Distribution by Type", breaking down the portfolio into office, villa, apartment, and garden units, while the section below again summarizes properties by city/region.
The third screen shows per-property KPIs, where each property card lists its occupancy rate, revenue, and costs, helping the landlord compare performance for every building.

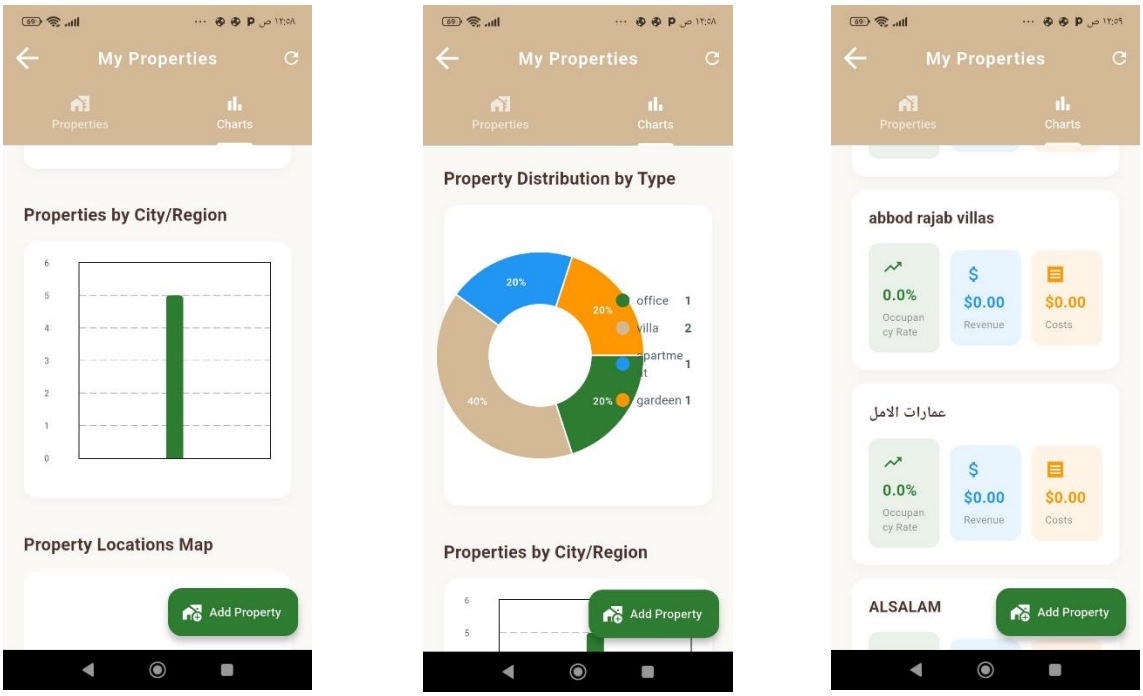


Figure6: Landlord Property Analytics and Performance Dashboard

Landlord Property Creation and Management Workflow: The Property Management feature lets landlords create, edit, and manage property listings. They can choose listing type (rent or sale), property type (apartment, villa, office, house), and enter details like title, price, area, description, bedrooms, bathrooms, amenities (WiFi, parking, pool, gym), location (map or address), photos, and optional 3D model links. After creation, properties appear in the landlord's My Properties list with status badges (PENDING_APPROVAL, RENTED, AVAILABLE) and can be edited or deleted. Once approved, these listings become visible to tenants in Home page , where they can search, filter, and view details before submitting rental requests. This connects landlords (who create and manage properties) with tenants (who browse and apply), with admin oversight for approval and system management.

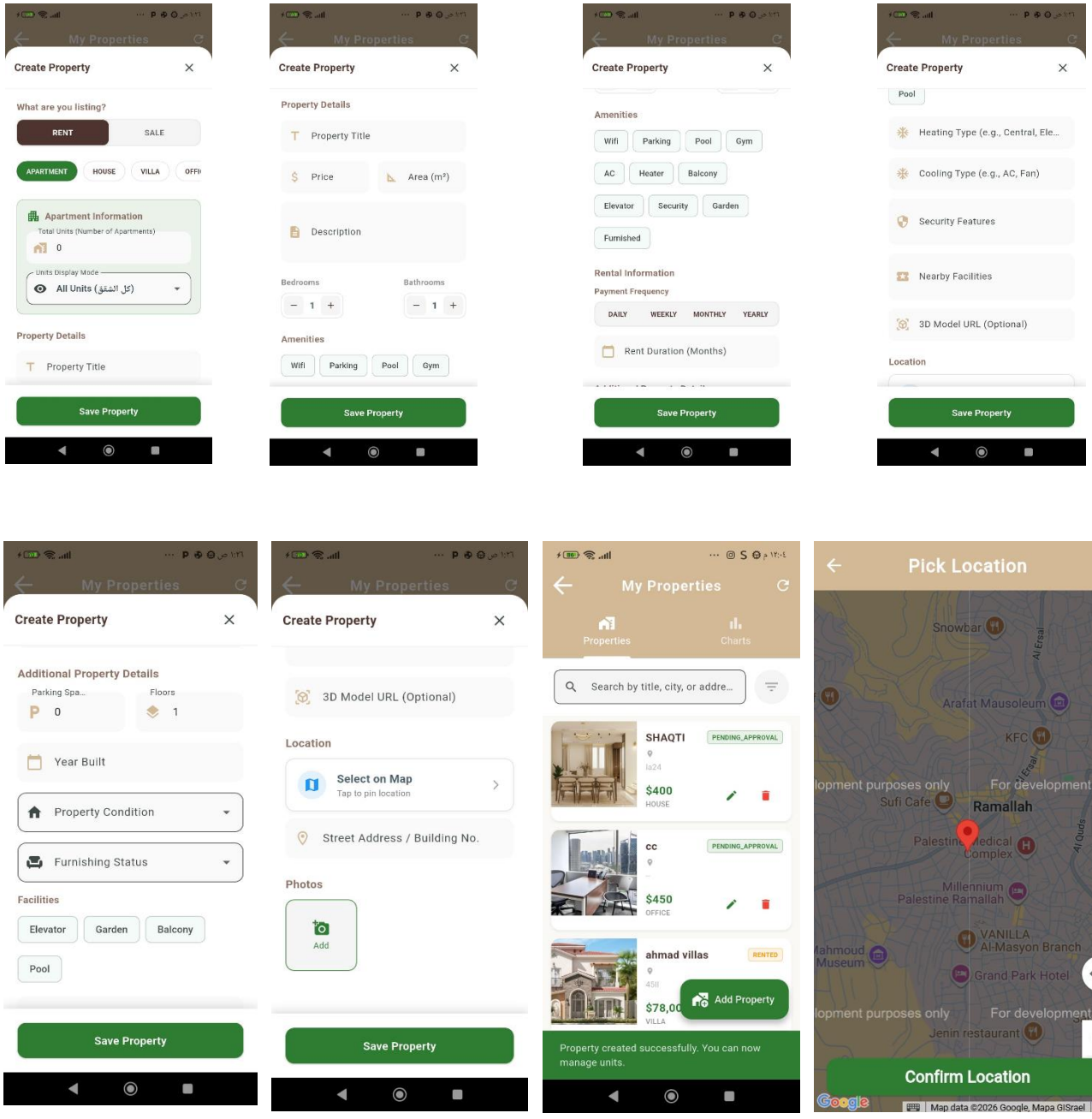


Figure7: Landlord Property Creation and Management Workflow

Contract Lifecycle, Payments, and Official Lease Document: These screens show the Contract Management feature, where each active lease between a landlord and tenant is tracked end-to-end. For a selected contract, the system displays the property, parties, duration, monthly rent, and automatically generates an official PDF lease that both sides can view, download, and share. The landlord can monitor contract progress, see how many payments have been made, view a financial summary (total paid, remaining balance, average payment), manage invoices and upload attachments (e.g., ID, signed papers). From the same view, they can open maintenance requests, update contract status, renew the lease, chat with the tenant, and see aggregate contract statistics (total, active, expired, pending) with sorting options such as newest, oldest, highest price, or by status. Tenants see their own contracts and payment schedule, while admins can oversee all contracts across the system for auditing and reporting.

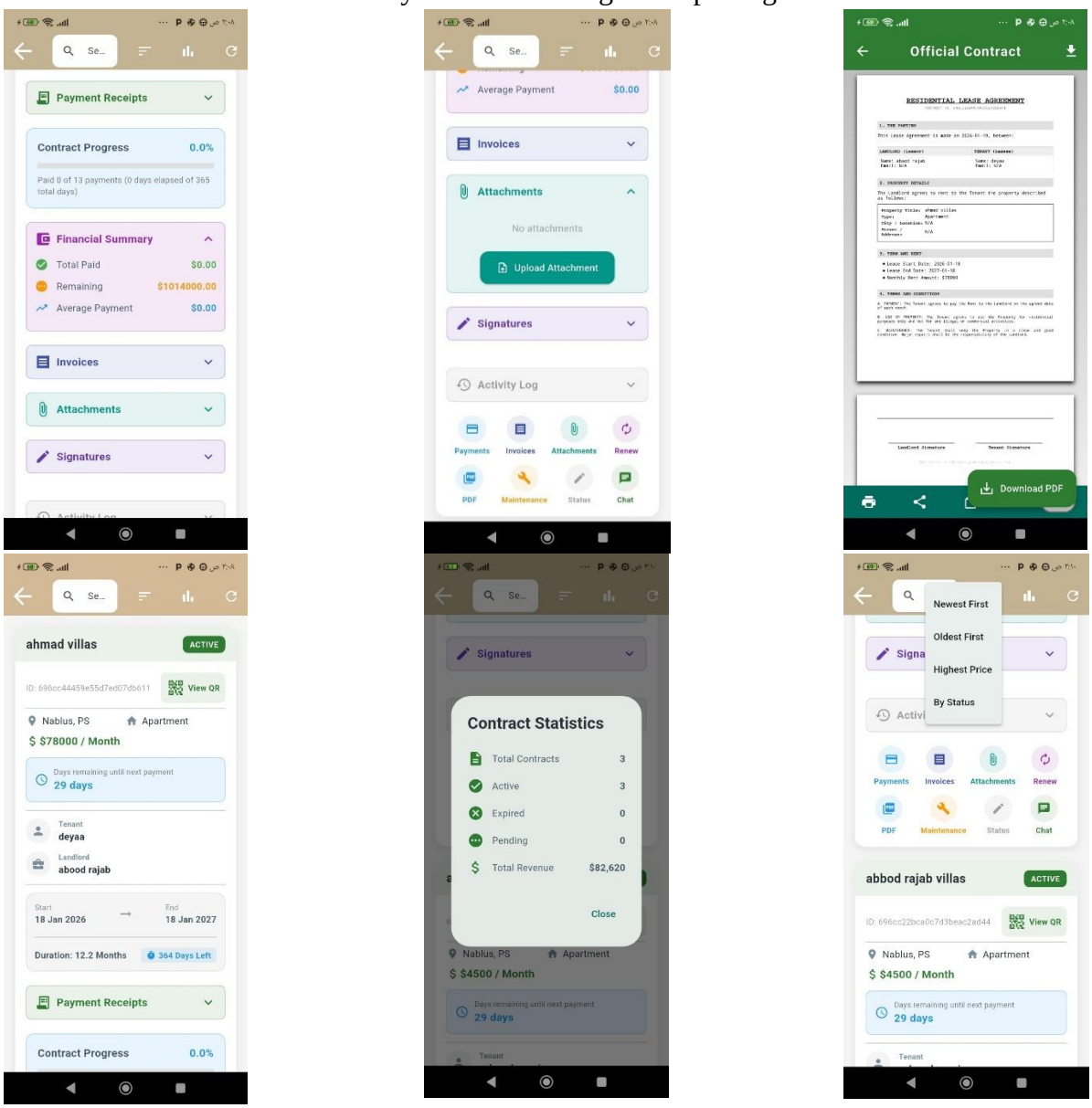


Figure8: Contract Lifecycle, Payments, and Official Lease Document

Landlord Maintenance & Complaints Management Dashboard: This screen allows the owner to track and manage all maintenance requests and complaints related to their property in one place. These requests originate from the tenant. At the top, summary cards display the total number of requests, the number of pending, ongoing, and completed requests, and the total cost, providing a quick overview of the building's status. The owner can search by apartment and open a filter window to narrow down requests by status (pending, ongoing, completed) and priority (low, medium, high).

Each request card displays the apartment name, image, and description, the requester's name and contact information, the creation date, and visual badges indicating the request type (complaint/maintenance), priority, and current status. From the card itself, the owner can update the request: change the status, modify the priority, and assign a technician (optional). The dashboard statistics and badges are then immediately updated to reflect the new status.

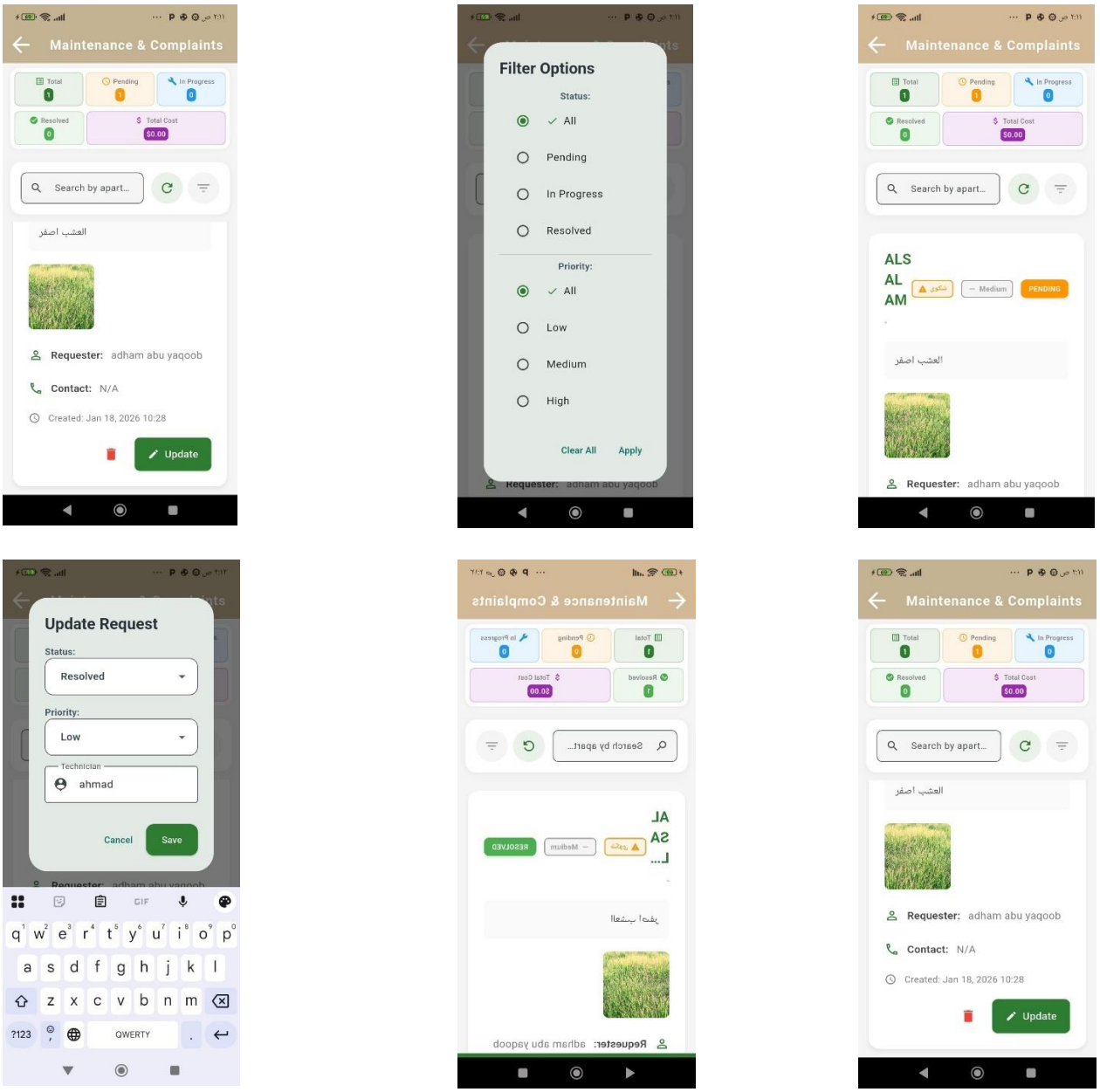


Figure9: Landlord Maintenance & Complaints Management Dashboard

Payments Dashboard and Transaction Details: The Payments & Transactions dashboard gives landlords and admins a financial overview. Summary cards show Total Payments (count), Total Revenue (sum), Paid (completed), and Pending (outstanding), providing a quick financial snapshot.

The list shows individual transactions with the amount, status badge (PAID/PENDING), tenant name, property name, payment method (e.g., TEST_VISA, Cash, Bank Transfer), date, and Payment ID. Users can search by tenant or property and filter by Status (Paid, Pending, Refunded) and Method (Visa, Cash, etc.).

Tapping "View Details" opens a detailed modal with the full payment record: amount, status badge, tenant, property, method, date, and Payment ID. This helps track payments, verify transactions, and audit financial records. Tenants see their own payments, landlords see payments for their properties, and admins see all transactions.

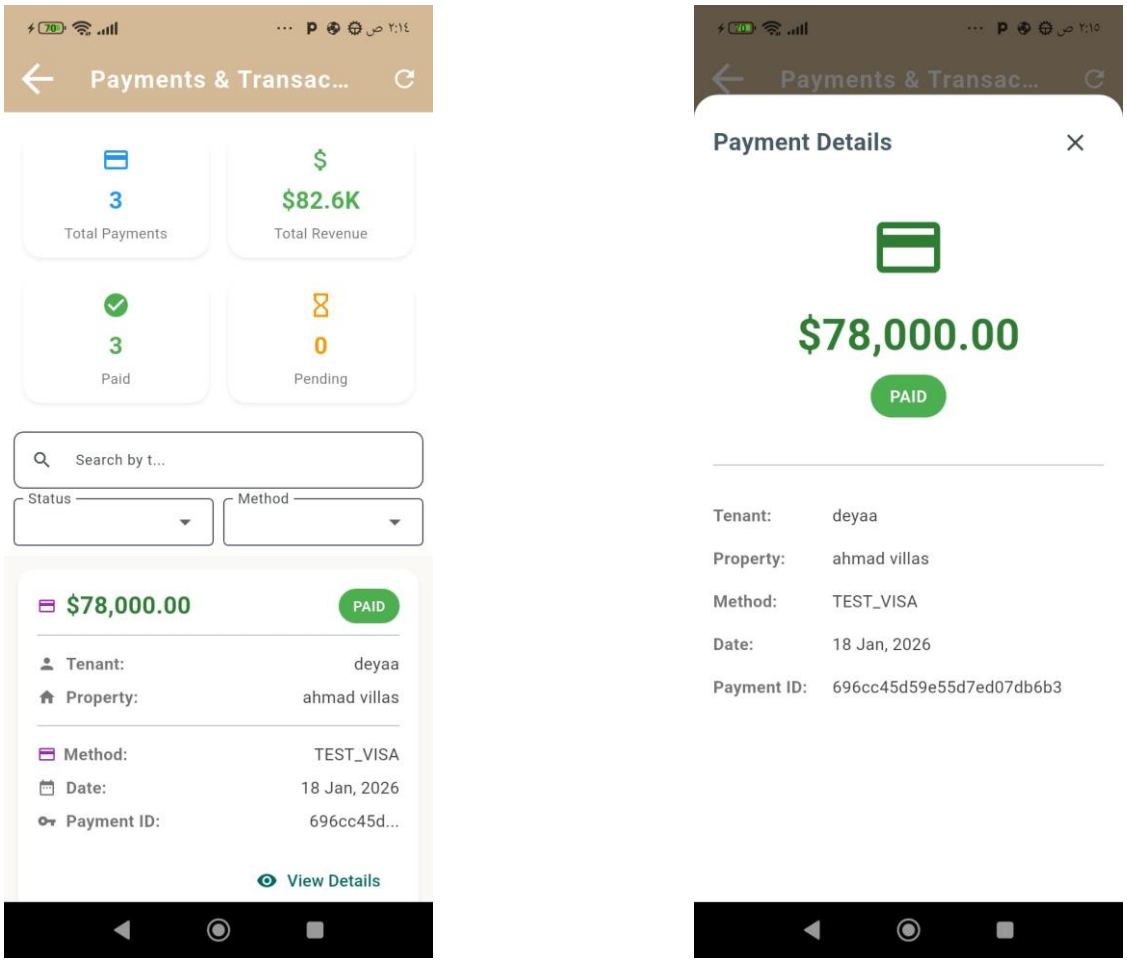


Figure10: Payments Dashboard and Transaction Details

Expenses Management Dashboard, Entry Form, List View, and Analytics Charts: The Expenses Management module lets landlords record and analyze all property costs. They can set a monthly budget, add expenses with type, amount, date, description, and receipt, then view a list of all entries. Summary cards and charts show total and highest expenses, budget vs. spent, and distribution by type and by property, so landlords quickly see where money is going and if they are over budget.

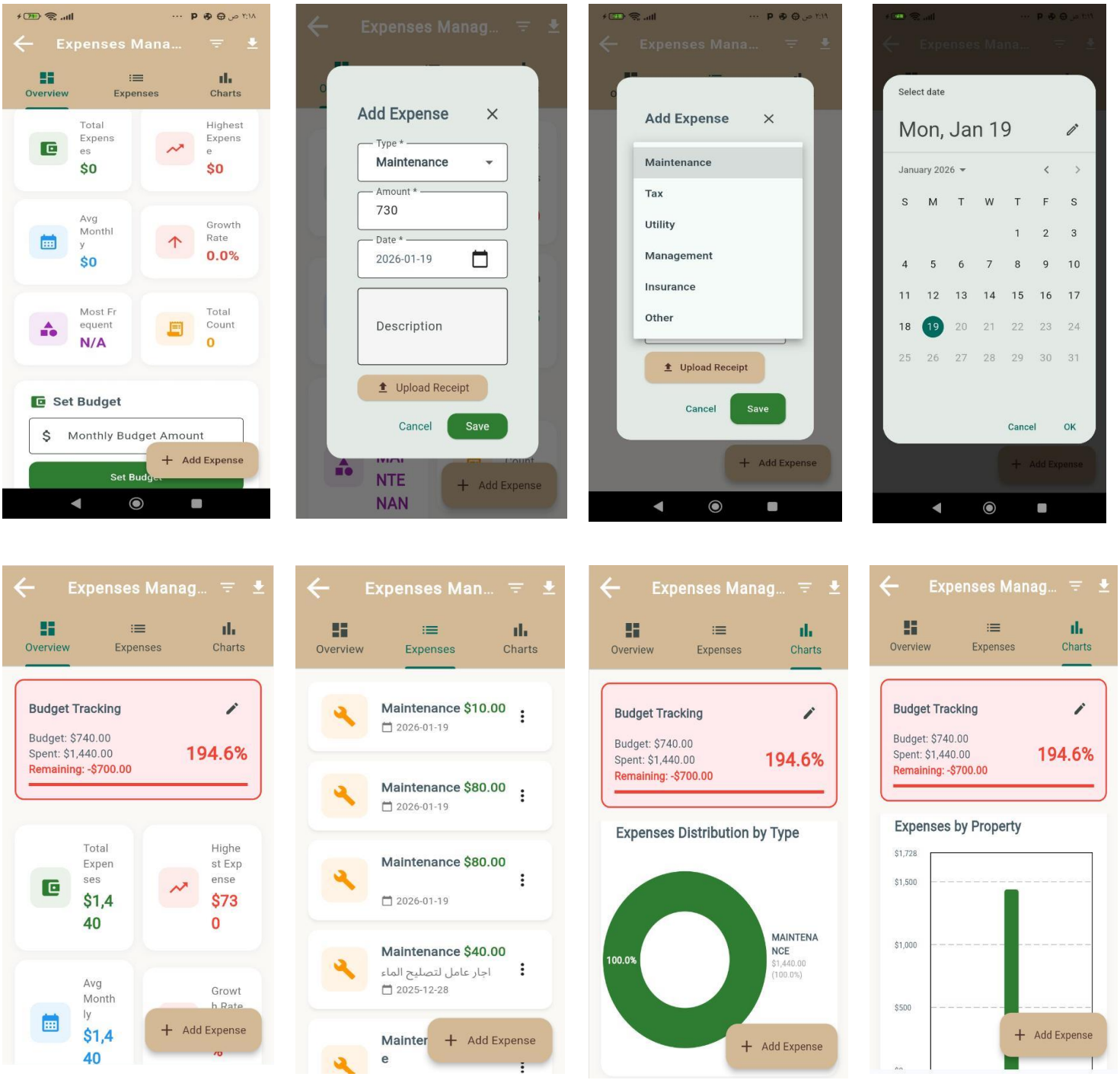


Figure11: Expenses Management Dashboard, Entry Form, List View, and Analytics Charts

Security Deposits Tracking, Refund Actions, and Analytics: The Deposits Management module tracks tenants' security deposits and their refund status. The Overview tab shows totals for deposits held, refunded, pending refunds, average deposit amount, and counts, plus analytics like full refund rate and average hold duration, with a recent activity feed. In the Deposits tab, each deposit card shows the tenant, amount, and a breakdown of deducted, refunded, and available balance, and the landlord can process full or partial refunds with one action. The Charts tab visualizes deposits by status and by property, helping landlords quickly see how much money is still held versus already refunded.

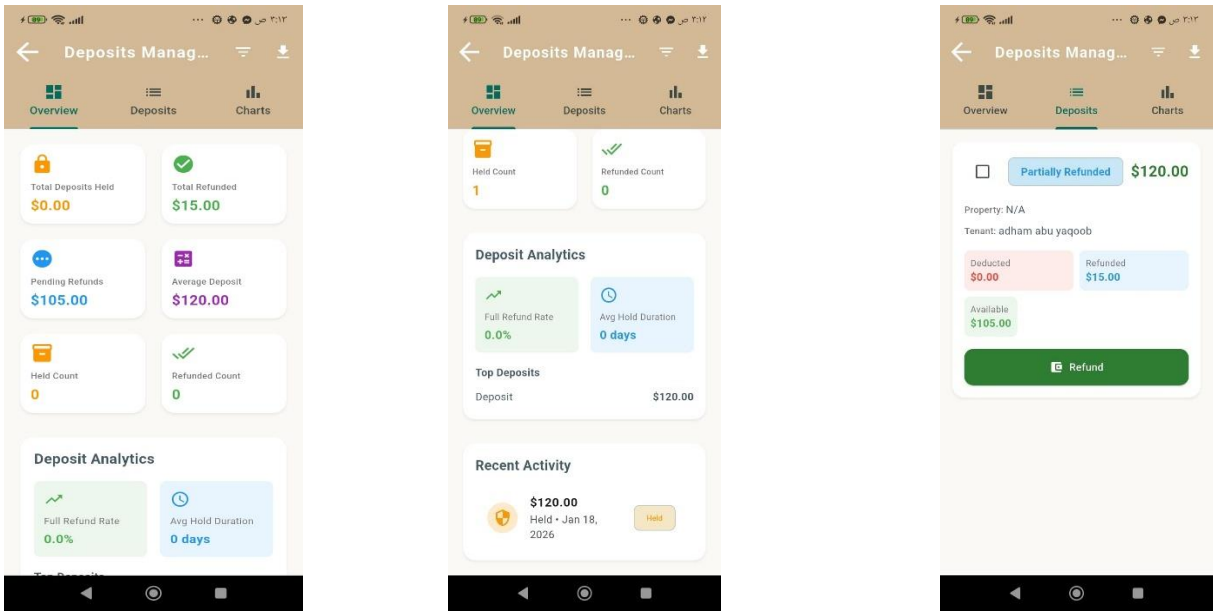


Figure12: Security Deposits Tracking, Refund Actions, and Analytics

Invoices Management, Status Tracking, and Collection Analytics: The Invoices Management module tracks all rent and fee invoices for the landlord. The Overview tab shows KPIs like total invoice value, paid, pending, overdue, average invoice amount, and collection rate, plus analytics for overdue amount and average payment time, and a recent activity list. In the Invoices tab, each invoice shows status, amount, tenant, property, issue date, due date, and the landlord can search by invoice number and send the invoice by email. The Charts tab visualizes invoices by status, by tenant, and monthly trends, helping owners see who pays on time and how revenue is distributed.

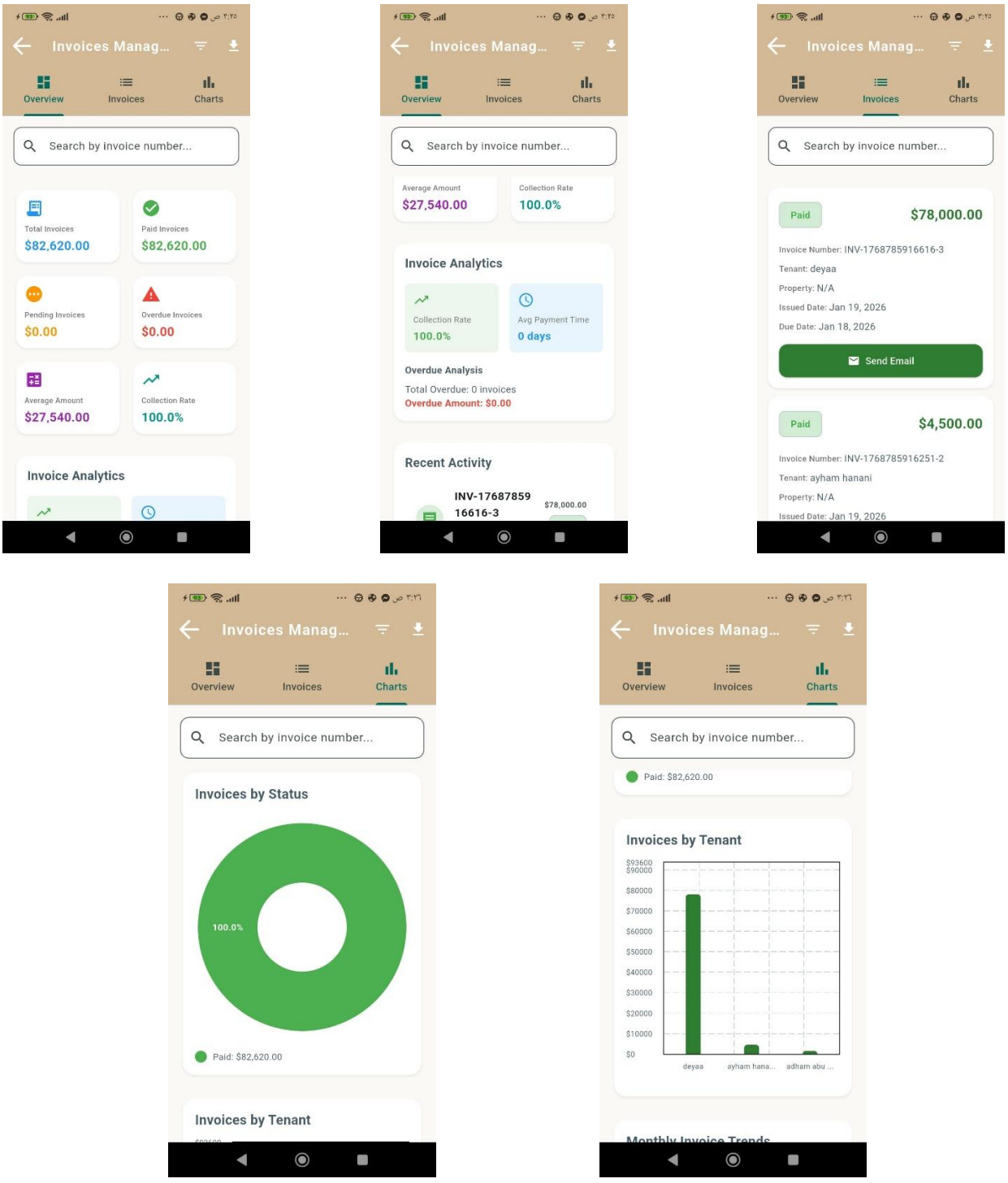


Figure13: Invoices Management, Status Tracking, and Collection Analytics

User Registration Screen: The Create Account screen lets new users register. The form includes Full Name, Email, Phone Number (optional), Password, and Confirm Password. After filling the fields, users tap "Create Account" to register. A link at the bottom navigates to the login screen for existing users. The system validates inputs and creates the account.

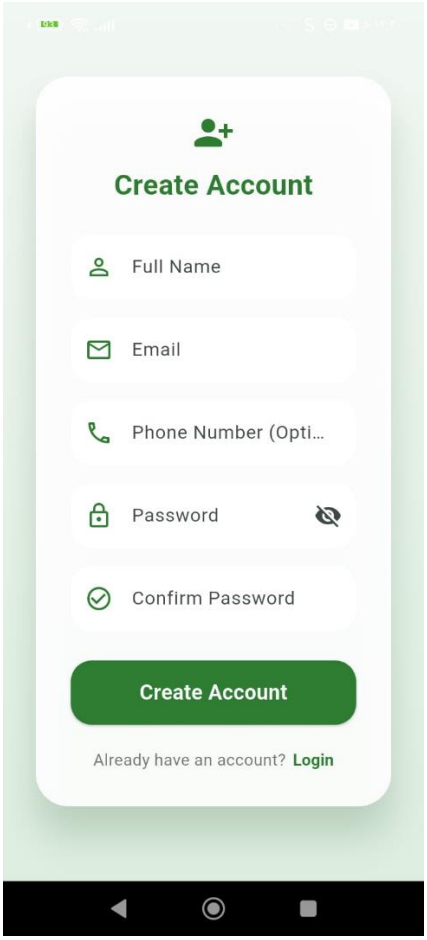
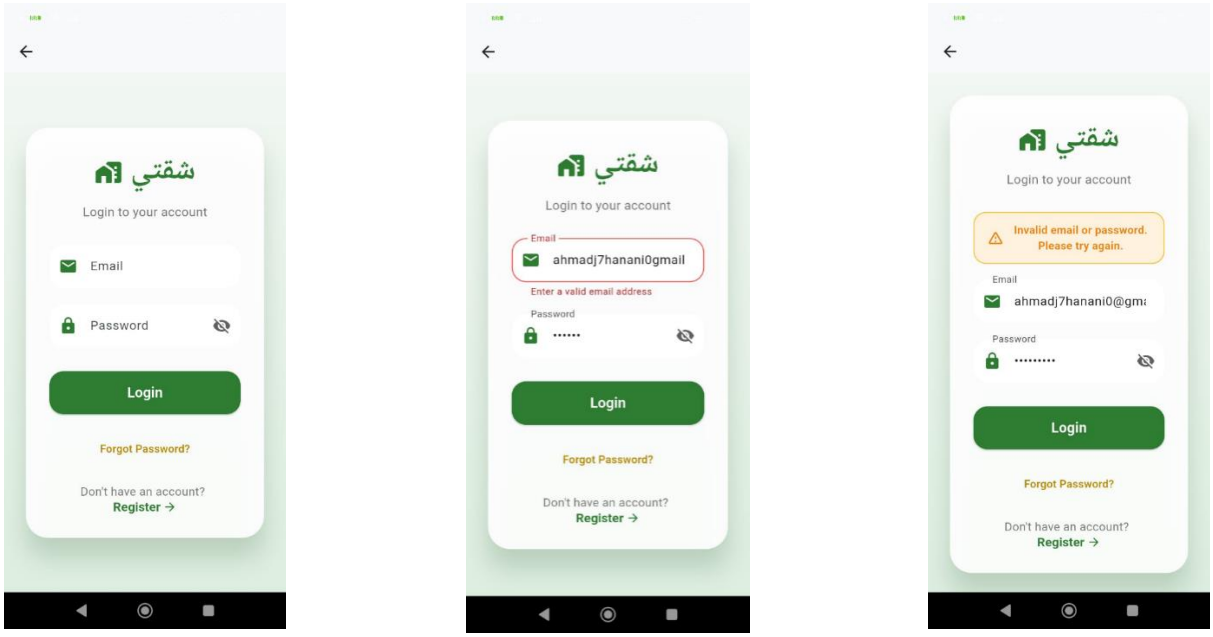


Figure14: User Registration Screen (sign up)

User Authentication Flow: The login screen allows users to sign in using their email address and password. Error messages such as "User not found" or "Incorrect email address or password" will appear if the credentials are incorrect.



This screen completes the password reset process. Users can use the "Forgot password?" option to reset their password by receiving a 6-digit random code via email. The interface displays the registered email address for confirmation and provides secure fields for code entry and new password creation, ensuring only authorized users can reset their accounts.

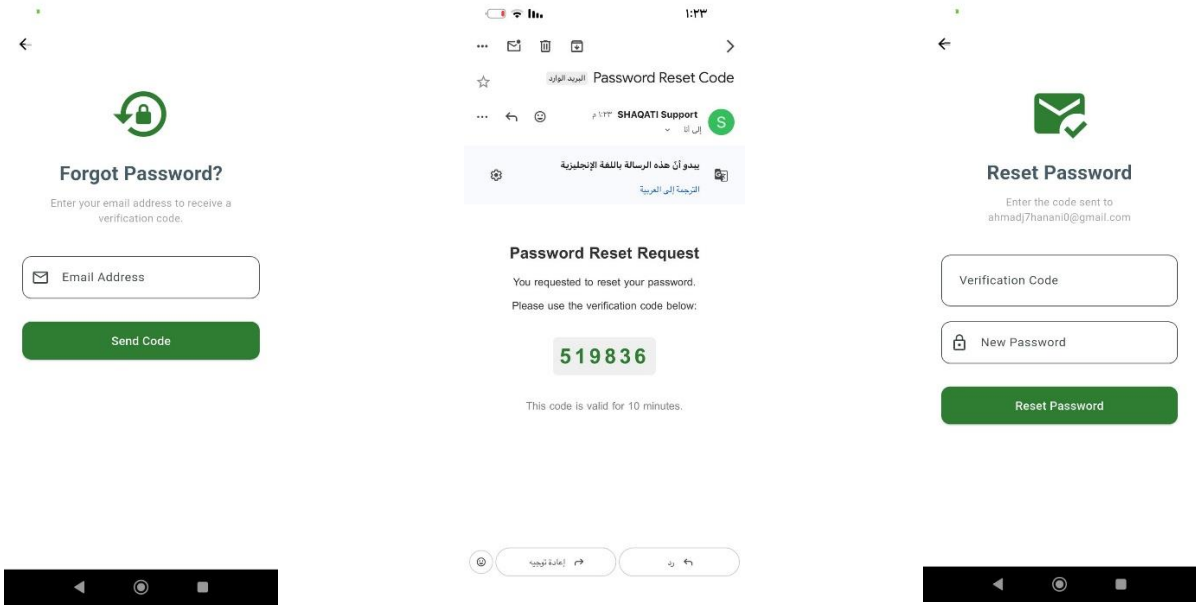


Figure15: User Authentication Flow & Reset password

** **Note** → Security enhancement: After every successful login, the system automatically sends a login notification email to the account owner. This lets users immediately detect any suspicious access and greatly improves account security and transparency

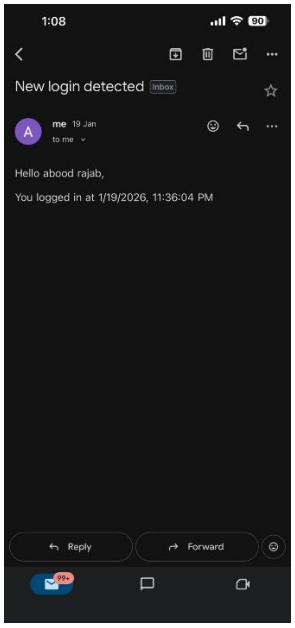


Figure16: Email Alert for New Login Detection

"**User not found**" → When the message "User not found" appears, it usually means that the email address has not been fully verified yet. The user must open their email inbox, find the verification message sent by the system, and confirm the account before they can log in successfully.

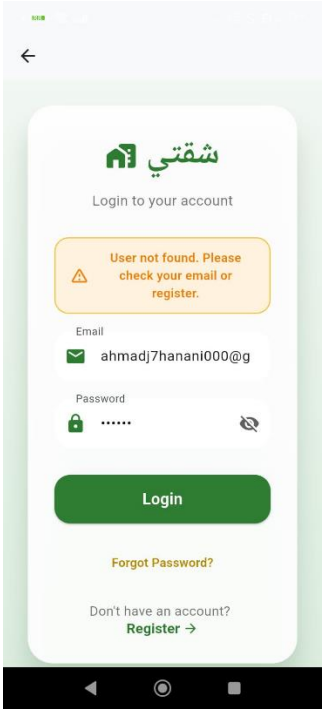


Figure17: User Not Found - Email Verification Prompt

Now we move on to the **admin pages and learn about their permissions and features.

Admin Dashboard – Overview

Admin Dashboard - System Overview and Analytics: The Admin Dashboard is a central control panel that gives the Administrator (Admin) a system-wide overview.

Main features

"Manage System" button for quick access to system settings

System summary

Total Users: Total registered users

Landlords: Number of property owners

Tenants: Number of tenants

Properties: Total properties

Contracts: Number of contracts

Payments: Number of payments

Additional statistics

Maintenance: Maintenance requests

Complaints: Complaints

Reviews: Reviews

Notifications: Notifications

Latest activities

Latest Users: Recently registered users (name, email, registration date)

Latest Properties: Recently added properties (name, price, status, date)

Latest Contracts: Recent contracts (start date)

Latest Payments: Recent payments (amount, status, date)

Performance analytics

Total Revenue: Total revenue

User Distribution by Role: Distribution of users by role (Tenant, Landlord, Admin) via a donut chart

Statistics by status

Properties by Status: Properties (rented, available, pending_approval)

Payments by Status: Payments (paid, pending) with amounts

Contracts by Status: Contracts (active)

Maintenance by Status: Maintenance requests (resolved, in_progress)

Permissions and functions

View all data: Access to system-wide statistics

Monitor system: Track users, properties, contracts, and payments

Manage users: View recently registered users

Monitor payments: Track payments and their statuses

Performance analysis: Review revenue and user distribution

System notifications: Receive notifications about activities (e.g., new property added)

Export data: PDF button to export reports

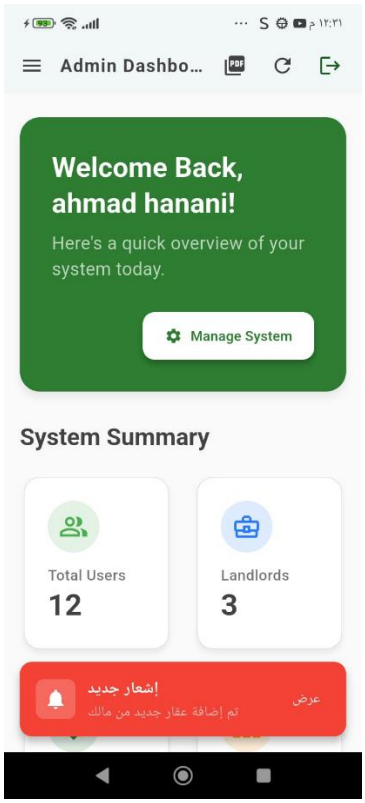
Refresh data: Refresh button to update information

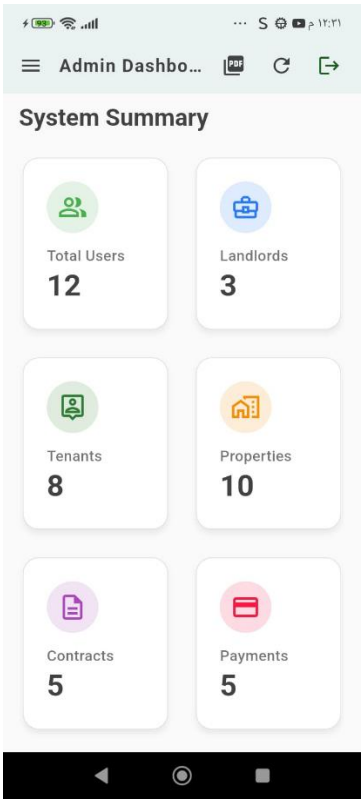
Design and navigation

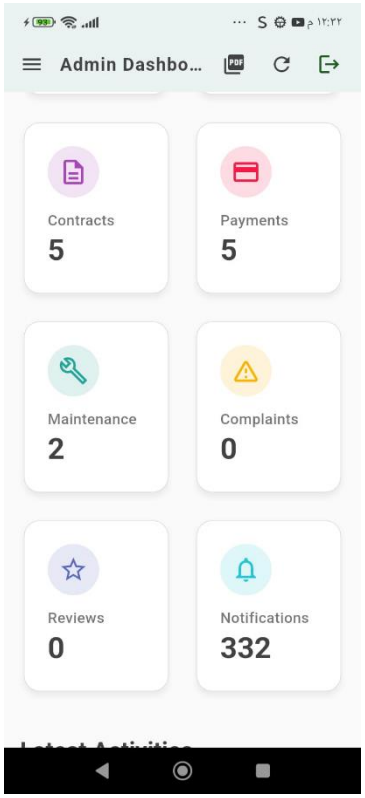
Side menu (Hamburger Menu): Access to other sections

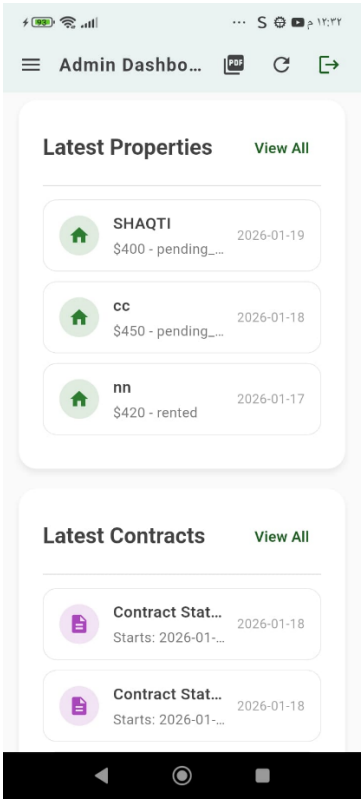
Quick action buttons: PDF, Refresh, Logout

"View All" links: Navigate to detailed pages


Admin Dashbo...
Welcome Back, ahmad hanani!
Here's a quick overview of your system today.
Manage System
System Summary
Total Users
12
Landlords
3
إشعار جديد
تم إضافة عقار جديد من مالك
عرض


Admin Dashbo...
System Summary
Total Users
12
Landlords
3
Tenants
8
Properties
10
Contracts
5
Payments
5


Admin Dashbo...
Contracts
5
Payments
5
Maintenance
2
Complaints
0
Reviews
0
Notifications
332


Admin Dashbo...
Latest Properties
View All
SHAQTI
$400 - pending_...
2026-01-19
cc
$450 - pending_...
2026-01-18
nn
$420 - rented
2026-01-17
Latest Contracts
View All
Contract Stat...
Starts: 2026-01-...
2026-01-18
Contract Stat...
Starts: 2026-01-...
2026-01-18

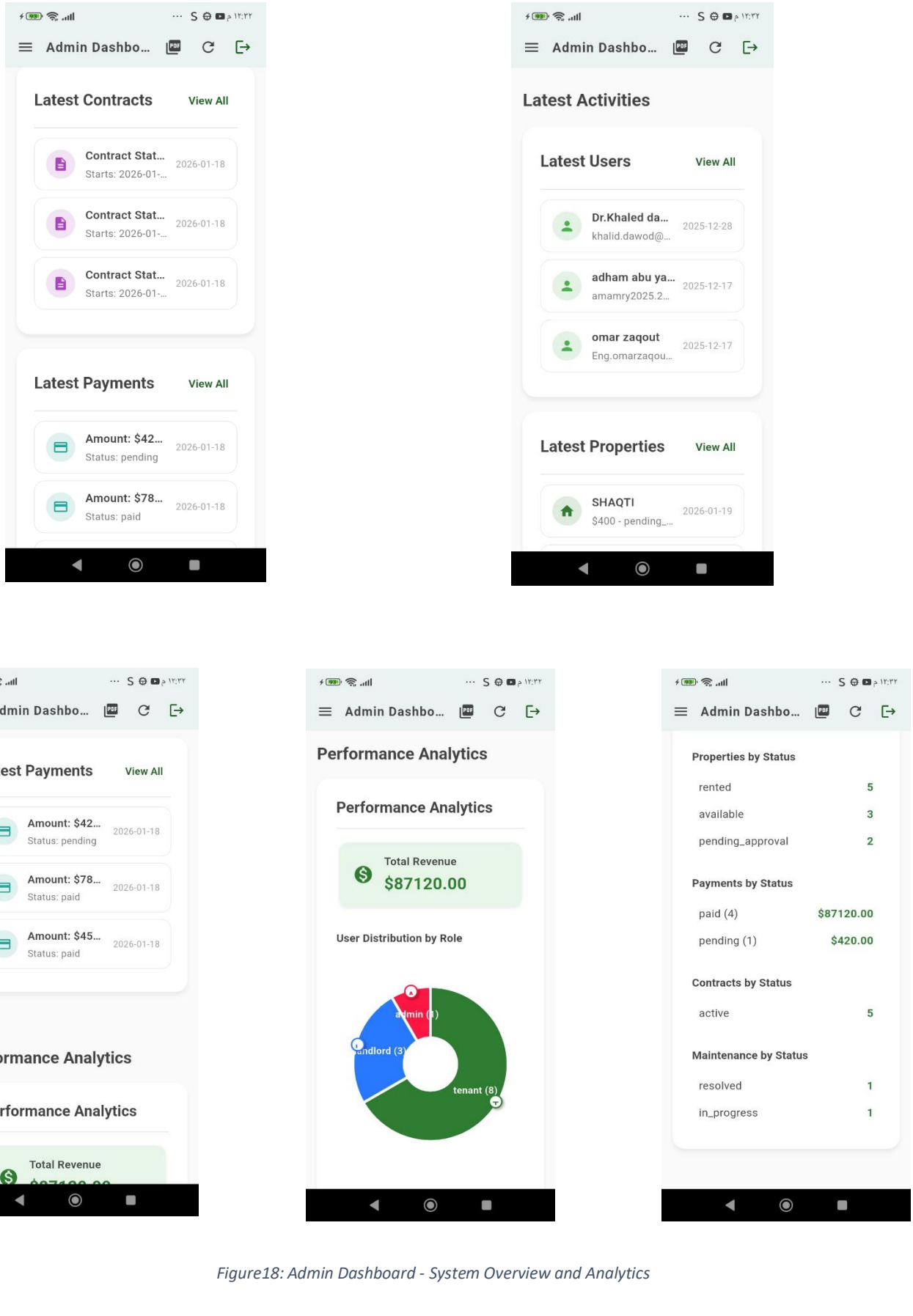


Figure18: Admin Dashboard - System Overview and Analytics

The **Admin Navigation Drawer** provides quick access to all management modules in the system. At the top it shows the admin's profile (name, email) with shortcuts to settings and logout, and below it lists the main sections: Dashboard Home, User Management, Property Management, Property Types, Contract Management, Contract Templates, Payments & Transactions, Maintenance & Complaints, Reviews Management, and Notifications Management. Red badges highlight modules with pending items, such as new contracts, open maintenance/complaint requests, and unread notifications, so the admin can immediately jump to what needs attention.

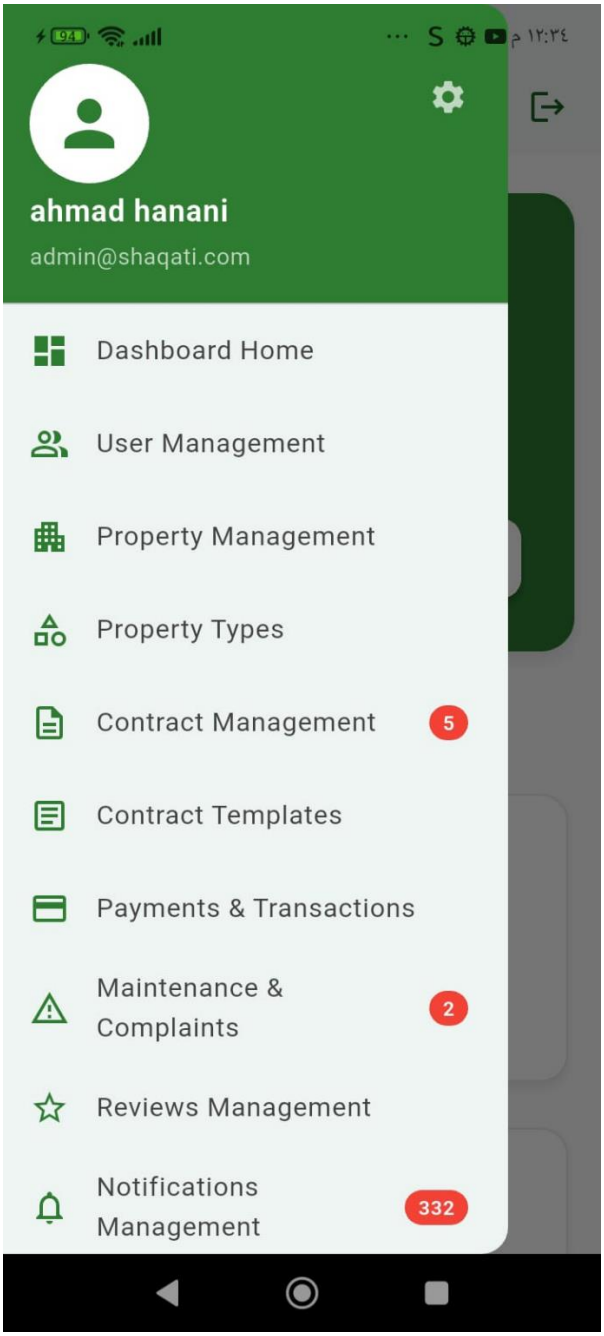
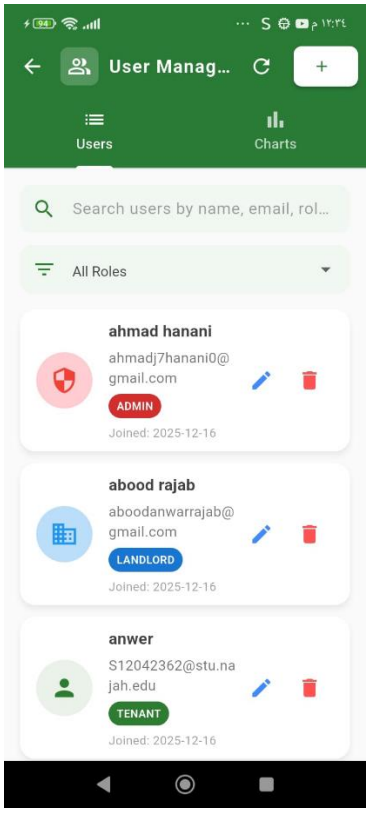
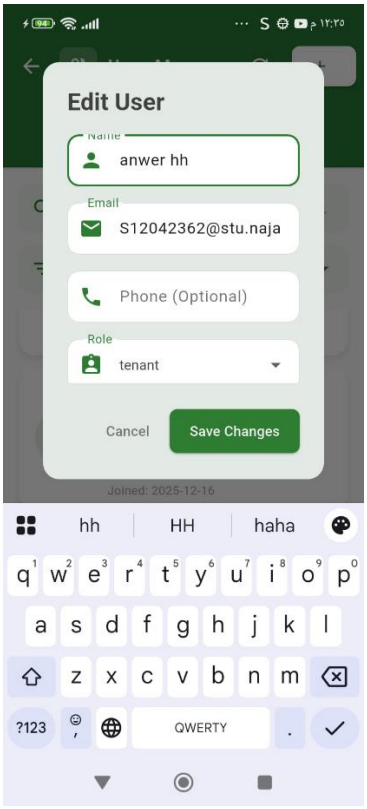


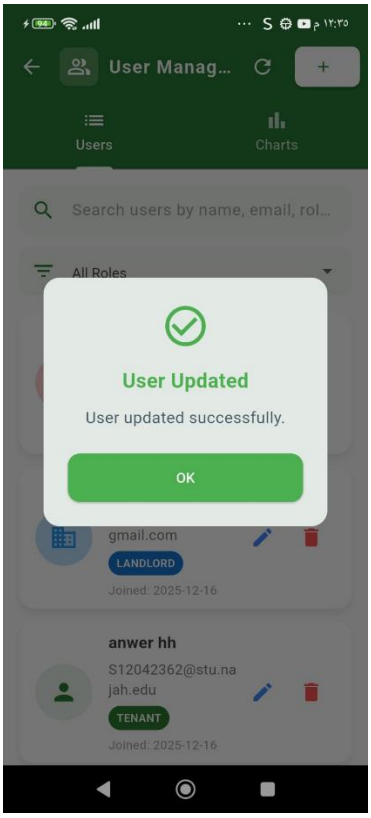
Figure19: Admin Navigation Menu

Admin - User Management Screen:

The **User Management Screen** enables administrators to manage user accounts. It displays users with names, emails, roles (Admin, Landlord, Tenant), phone numbers, and join dates. Each user has a color-coded avatar and role badge. A search bar filters by name, email, or role, and a dropdown filters by role. Administrators can create, edit, and delete users via dialog forms. Deletion requires confirmation. The screen includes two tabs: "Users" lists all accounts with management options, and "Charts" shows a pie chart of users by role. Success notifications appear after operations, and the list refreshes automatically.







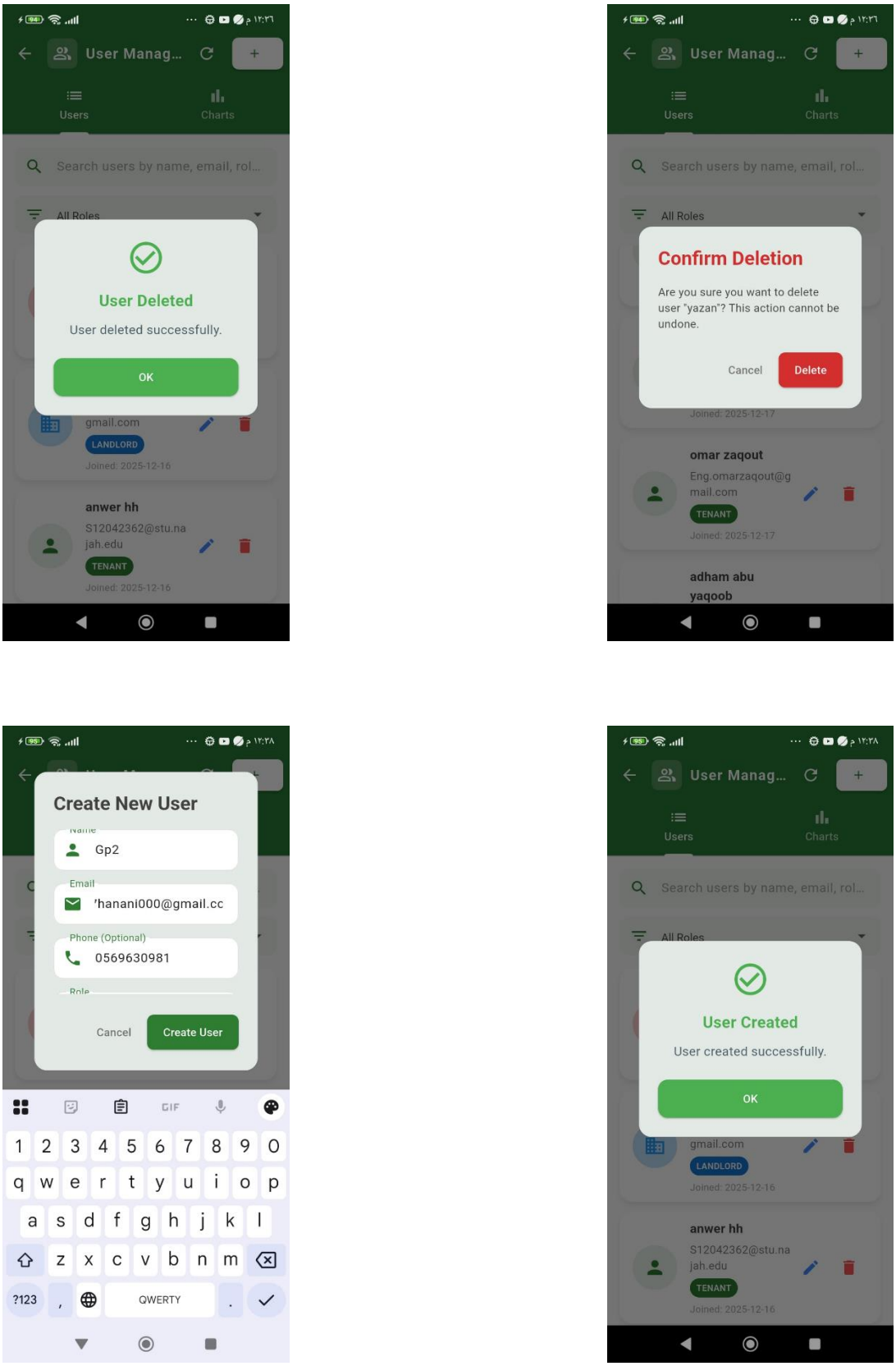


Figure20: Admin- User Management Screen

The Charts tab displays statistics through visualizations. In User Management, a donut chart shows the distribution of users by role (Admin, Landlord, Tenant) with color-coded segments and counts in a legend. In Property Management, two donut charts are shown: one for properties by status (available, occupied, maintenance) and another for properties by type (apartment, villa, etc.), allowing administrators to quickly view data distribution and trends.

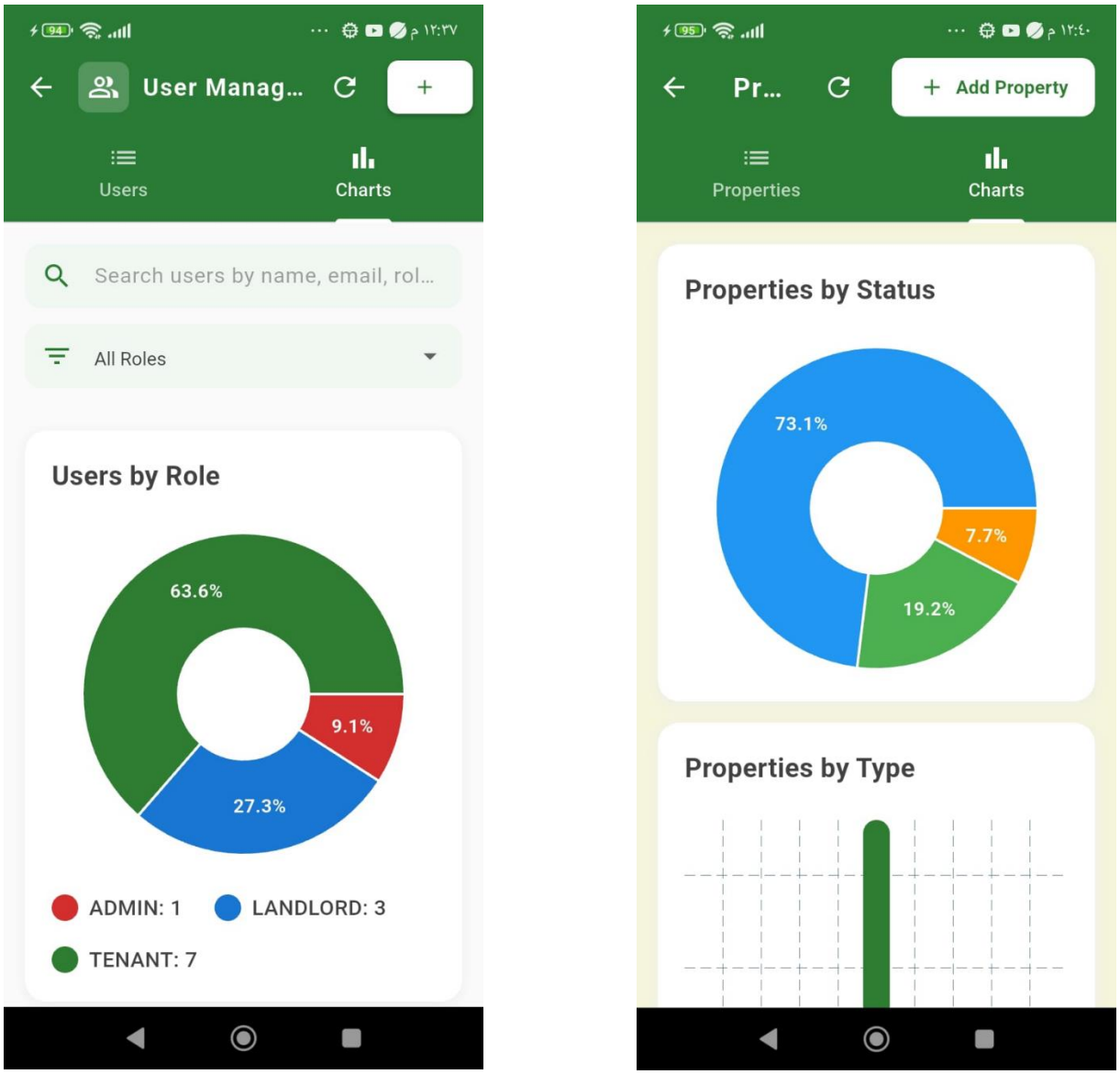


Figure21: Statistics Charts

The **Property Management Screen** enables landlords to manage properties. It displays a scrollable list of property cards showing an image, name, price, and status badge (AVAILABLE, RENTED, or PENDING_APPROVAL). Each card includes edit and delete buttons. The screen includes a search bar for properties and a status filter dropdown (ALL STATUSES, PENDING APPROVAL, AVAILABLE, RENTED). Properties awaiting approval show an "Approve" button. Administrators can approve pending properties. The "Add Property" button opens a form to create new listings. After creating or updating, success notifications appear. The screen includes two tabs: "Properties" for the list and "Charts" for visual statistics.

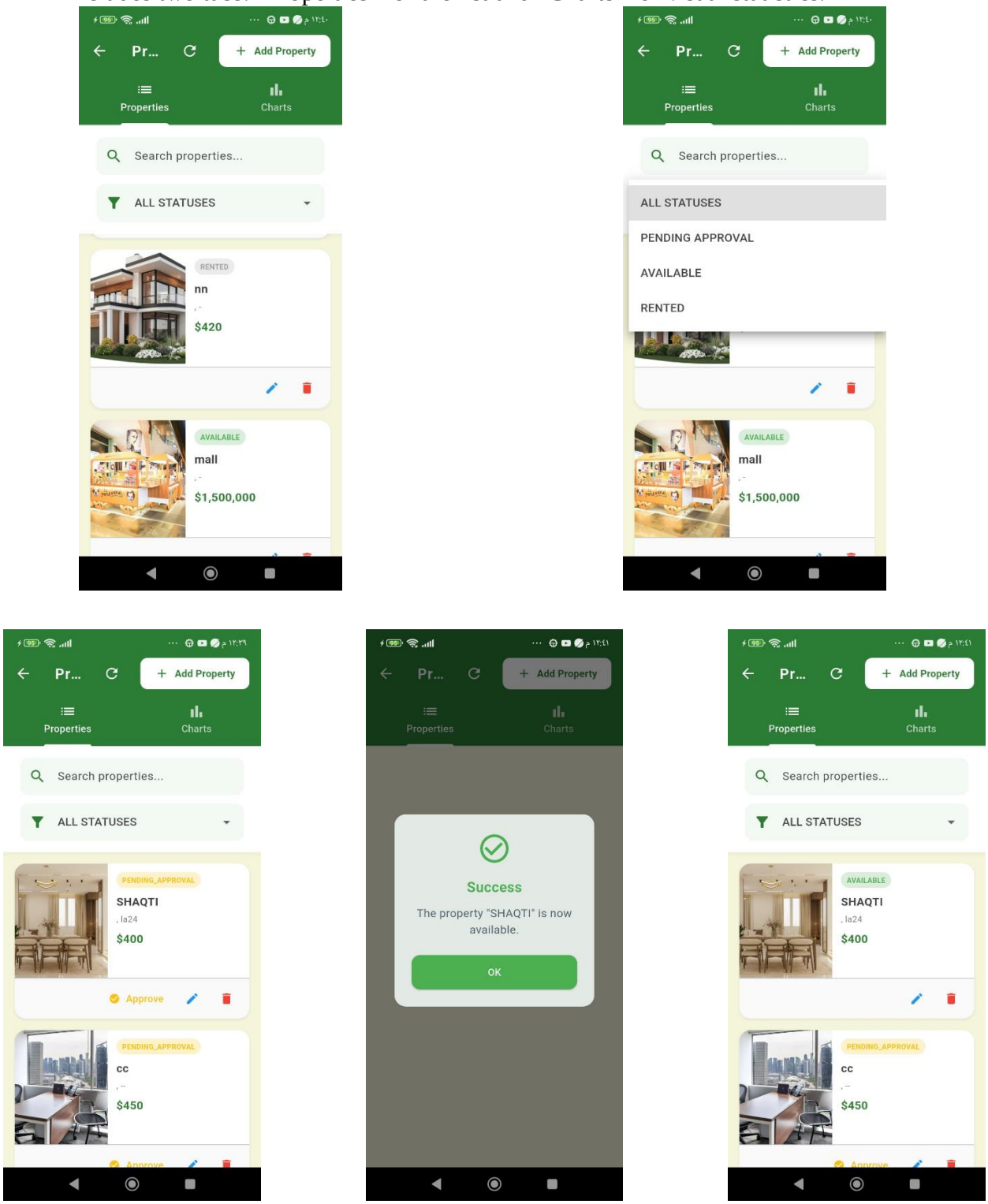


Figure22: Admin - Property Management screens

The Property Management Screen enables landlords to edit existing property listings through a form. **The Edit Property** form allows updates to property details, including status (AVAILABLE, RENTED, PENDING_APPROVAL), operation type (RENT or SALE), property type (APARTMENT, HOUSE, VILLA, etc.), title, price, area, and description. After saving changes, a success notification appears confirming the update. The screen also displays a scrollable list of property cards showing images, names, prices, and status badges, each with edit, delete, and (for pending properties) approve buttons.

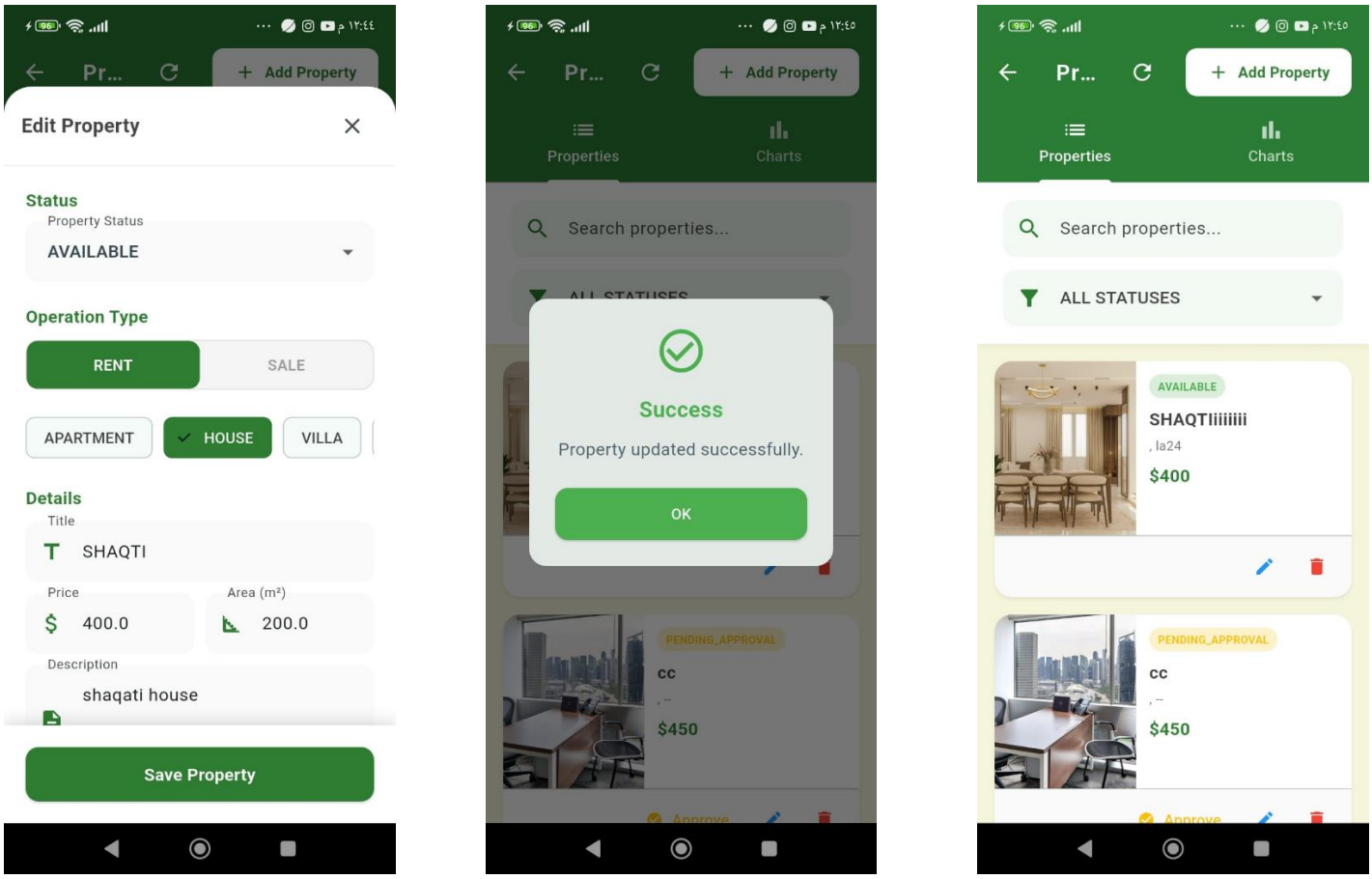


Figure23: Admin-Edit Property

This screens allow the admin to define and manage reusable **property types** that are used across the SHAQATI system, such as apartment, house, villa, office, shop, and custom categories. The list view shows each type with its internal name, display name, display order, and active status, while the three-dot menu lets the admin edit or deactivate a type without losing historical data. Through the “Add Property Type” dialog, the admin can create new types by specifying a lowercase key, a user-friendly display name, an icon, an ordering number, an optional description, and whether the type is active. The charts tab provides a quick donut chart overview of how many property types are active or inactive, helping the admin keep the catalog organized and consistent for all landlords and tenants.

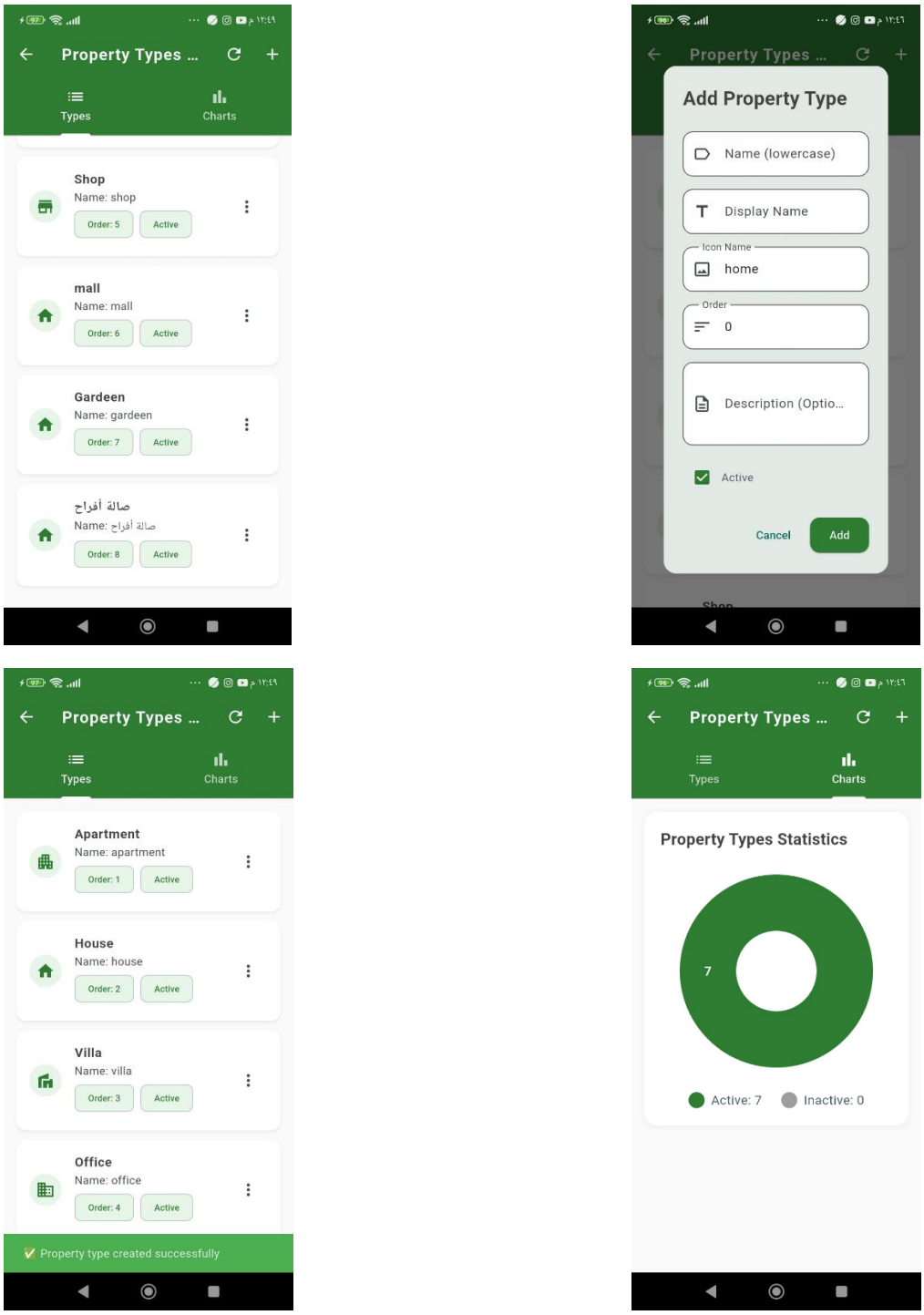


Figure24: Admin Property Types Management – list view, add type form, and statistics chart

This set of screens shows how SHAQATI lets the **admin manage property types** that are used when landlords create new properties. From the Create Property form, the landlord chooses an operation type (rent or sale) and then selects one of the predefined property types (such as mall, garden, or event hall) that were configured by the admin. In the Property Types list, the admin can open the menu for any type to edit its details, deactivate it, or permanently delete it; a confirmation dialog appears before deletion to prevent mistakes. After a type is removed, the list is refreshed and a success message confirms that the property type was deleted successfully.

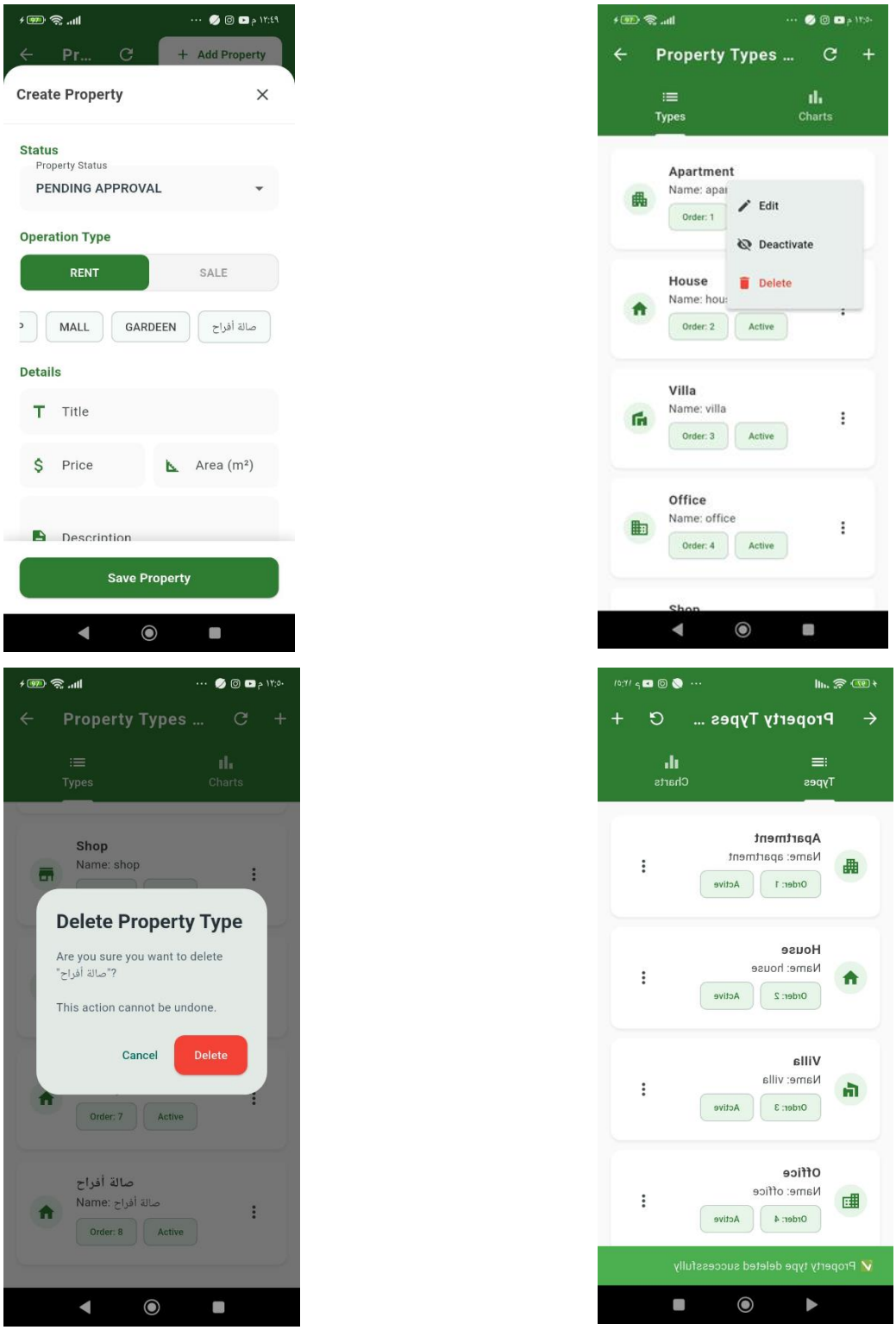


Figure25: Admin Property Types – create, edit, and delete property categories

In the **Property Types management** screen, the admin can edit an existing property category such as “Villa” by updating its technical name, display name, icon name, display order, and optional description while also choosing whether the type is active. After saving, the list of types is refreshed to show the updated information and a success message confirms that the property type was updated correctly, ensuring that all landlords use a clean and consistent set of categories when creating properties.

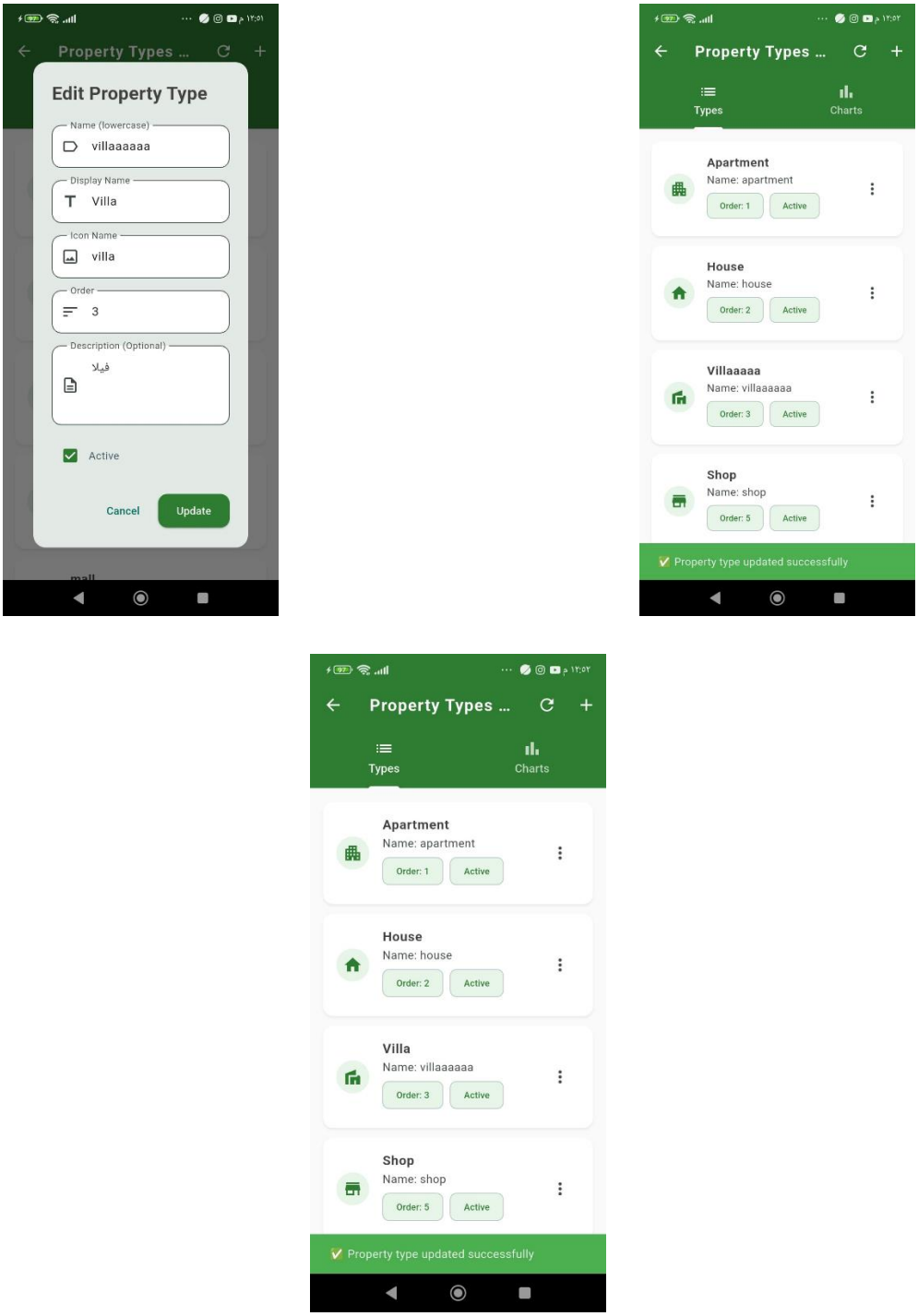
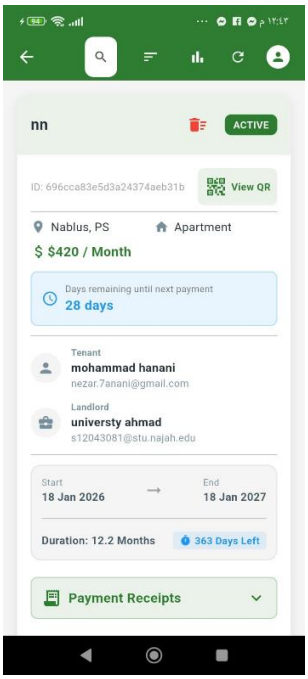
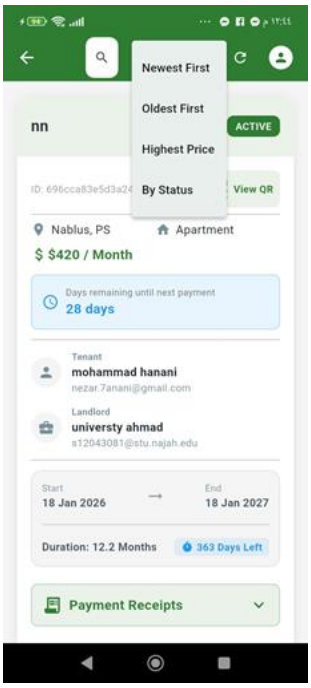
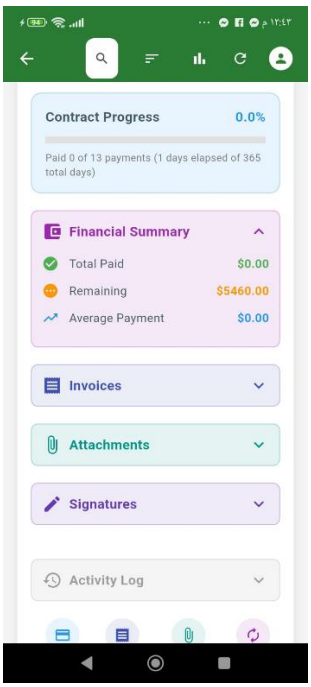


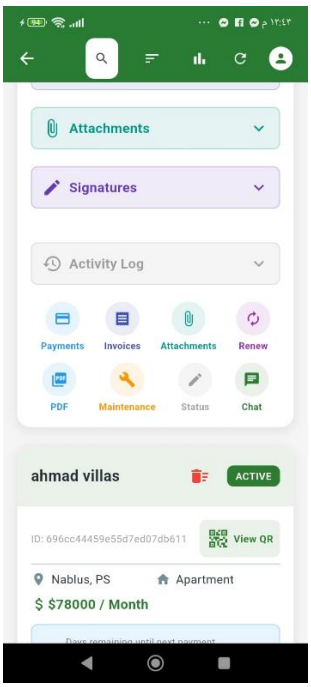
Figure26: Admin Property Types – edit category details

In the **admin contract** screen, SHAQATI shows the full details of an active lease agreement, including property location and type, monthly rent, days remaining until the next payment, and the linked tenant and landlord accounts. The admin can quickly sort contracts, review contract progress, and open expandable sections for financial summary, invoices, attachments, digital signatures, and the activity log, as well as access shortcuts for payments, maintenance, status changes, chat, PDF export, and viewing other contracts. Additional dialogs allow the admin to see quick statistics for the portfolio (number of active or pending contracts and estimated revenue), renew a contract by selecting new start and end dates, and change the contract status (draft, pending, active, expiring soon, expired, terminated, or rejected). From this screen the admin can also open the automatically generated official lease document in PDF format, which includes both parties' information, property details, rent terms, and signature fields ready for printing or digital signing.









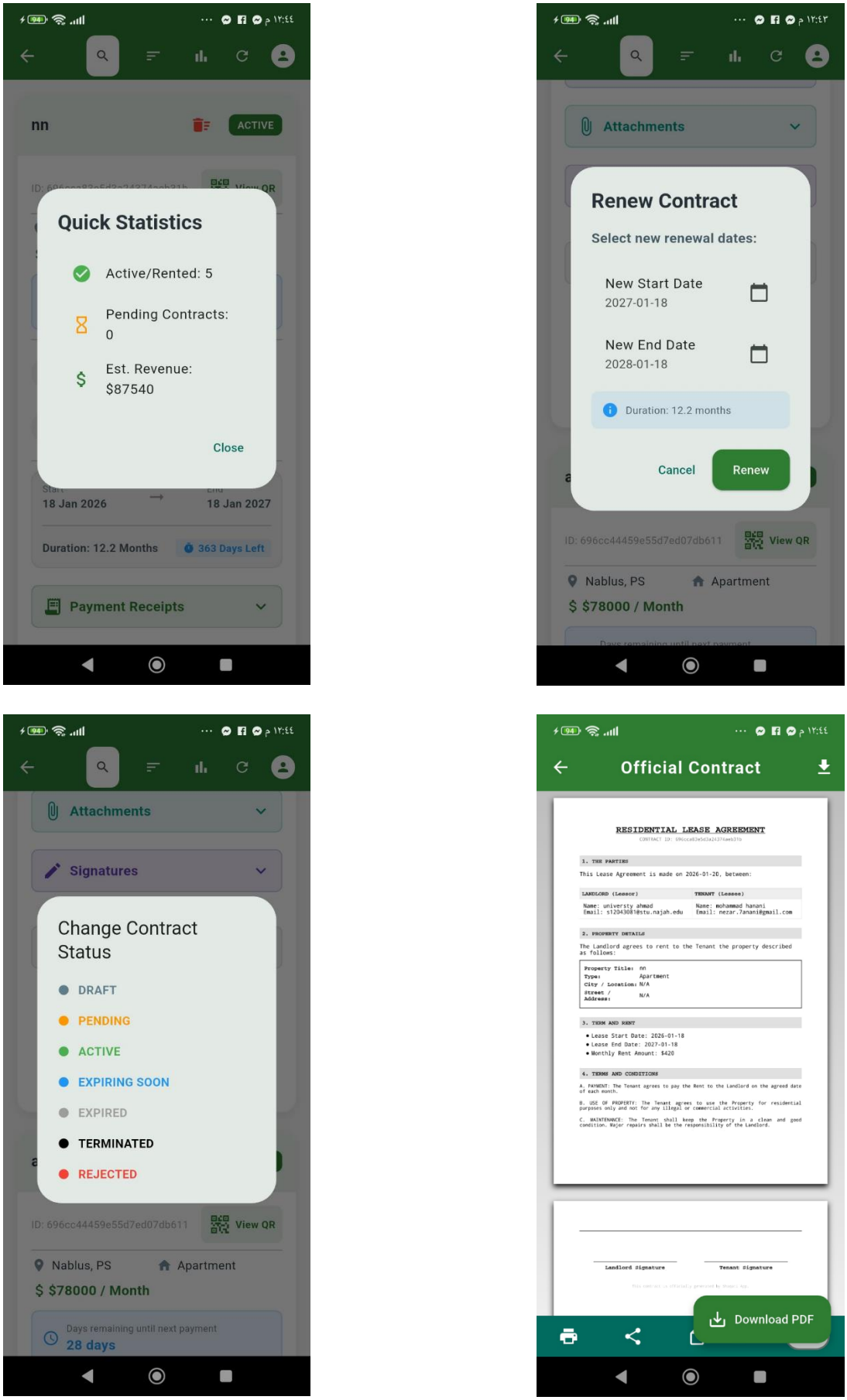


Figure27: Admin Contract Details – manage lease status, payments, renewals

The **payments** screen gives the admin a complete overview of all rent payments processed, showing total payments, total revenue, and how many payments are paid or still pending. Each payment record lists the tenant, related property, amount, payment method (such as TEST_VISA), date, and status, and the admin can sort the list by newest, oldest, highest amount, or lowest amount, as well as filter by status and method. For large or important payments, visual badges highlight the amount, and the admin can open an individual payment to review its details and either approve it or mark it as failed, ensuring accurate financial tracking and control over online transactions.

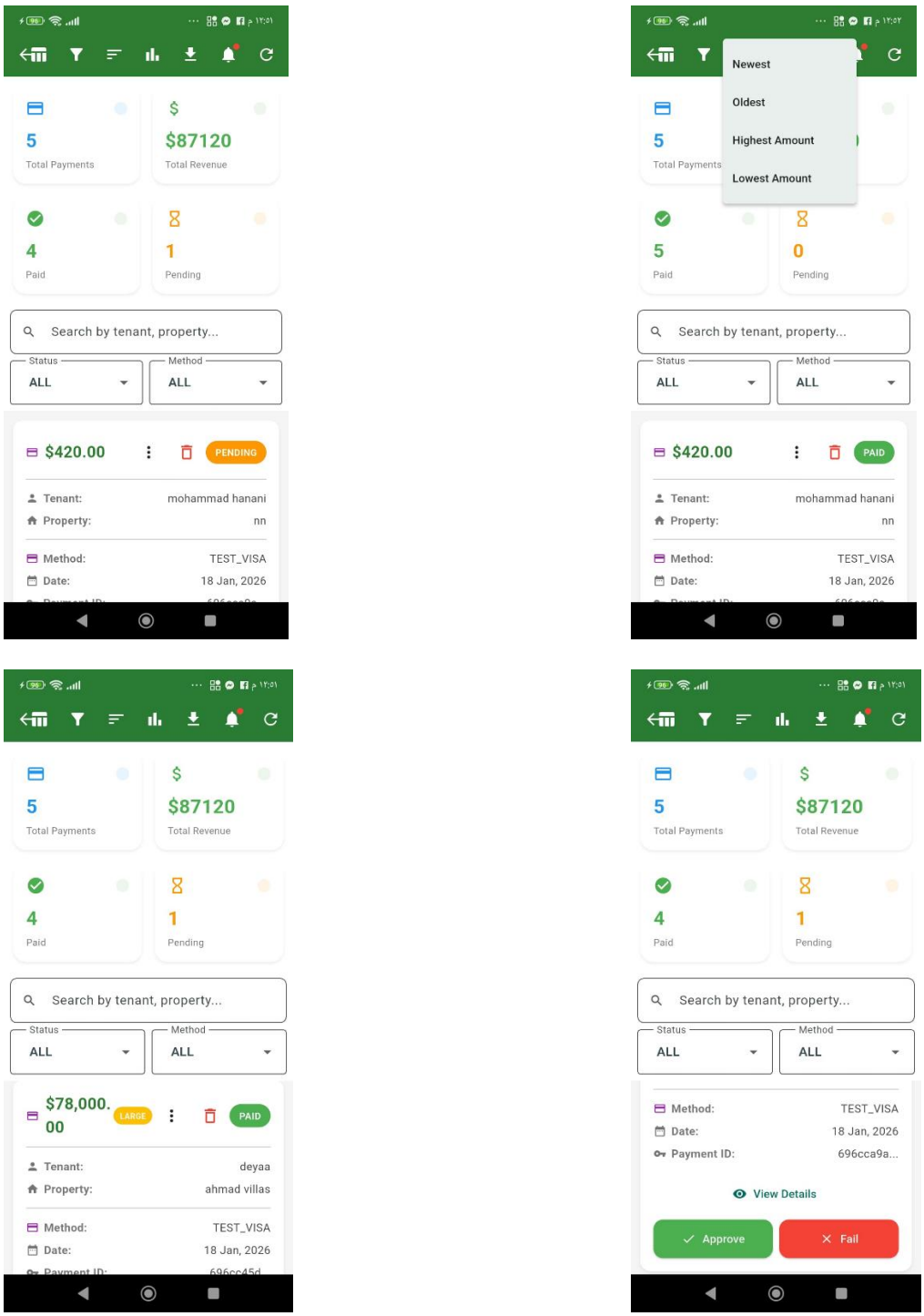


Figure28: Admin Payments – overview, filtering, and approval of rent transactions

In addition to viewing and approving individual payments, the admin can export all payment records as a CSV file for Excel analysis or as a formatted PDF report. Advanced filters allow narrowing results by date range, amount range, specific property, or tenant, and these filter sets can be saved for reuse. From the payment details dialog, the admin can review the full transaction, including method, status, exact time, receipt number, tenant information, and related property. A separate analytics popup summarizes all payments visually, showing how many payments are paid and the total revenue over time using charts.

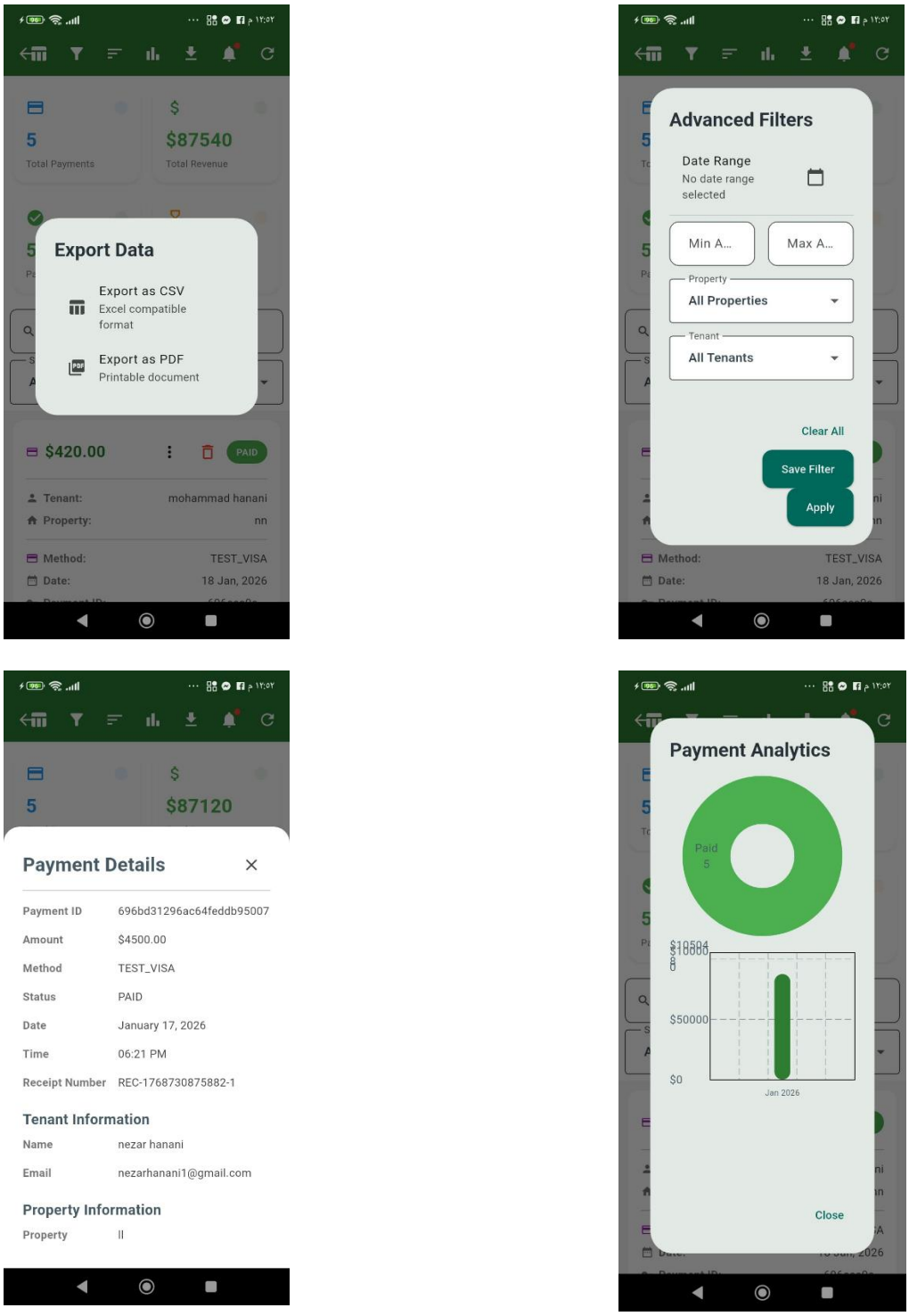


Figure29: Admin Payments – export options, advanced filters, detailed receipt view, and revenue analytics

The **Maintenance & Complaints** screen lets the admin monitor all requests submitted for the landlord's properties, showing quick counters for total, pending, in-progress, resolved tickets, and total cost. Each request card displays the apartment, priority level, current status (such as IN PROGRESS or RESOLVED), description, requester name, contact details, creation time, and action buttons to update or delete the ticket. From the Charts tab, the admin can view visual analytics, including a donut chart of requests by status and a bar chart of requests by priority, which helps identify workload distribution and service quality at a glance.

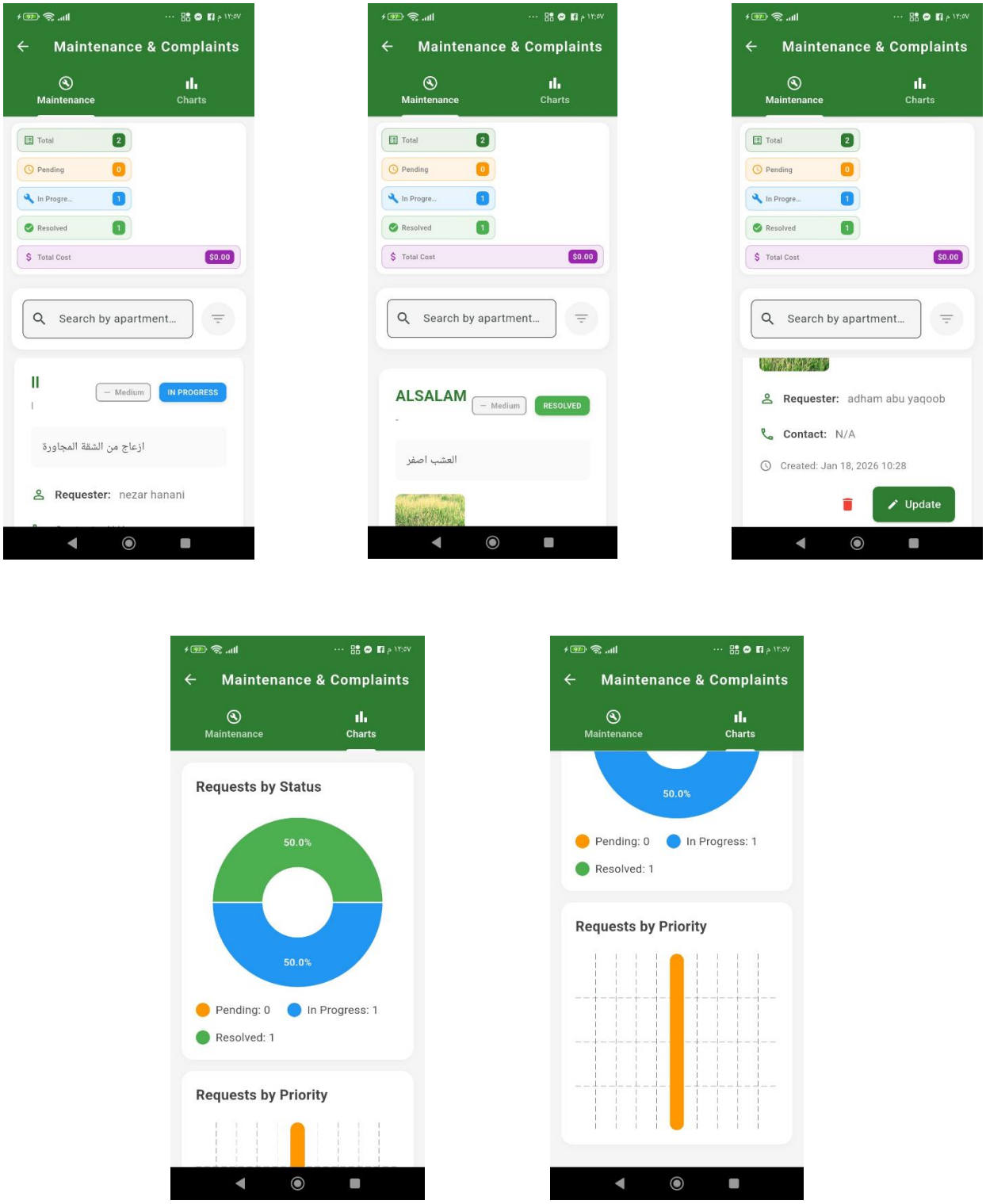


Figure30: Admin Maintenance & Complaints – request list, actions, analytics

The **Reviews** screen allows the admin to manage tenant feedback on properties by searching through reviews, filtering them by star rating, and sorting them by newest, oldest, highest, or lowest rating. When no reviews have been submitted yet, the screen clearly shows a "No Reviews" message, while a statistics popup summarizes the total number of reviews, the average rating, and how many reviews are visible or hidden, helping the admin quickly understand user satisfaction.

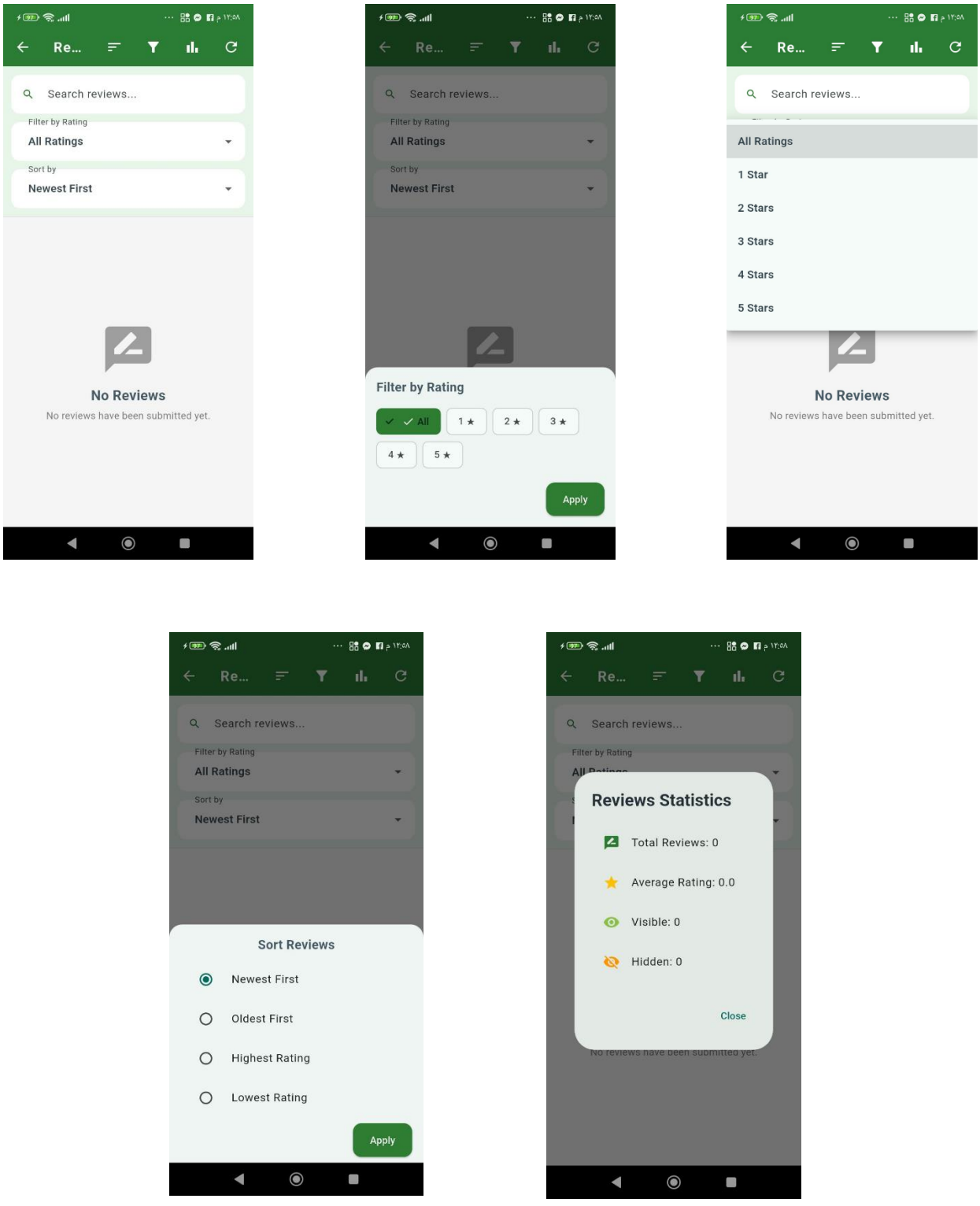


Figure31: Admin Reviews – search, filter by rating, sorting, and statistics

The **Notifications Management** screen allows the admin to review all system messages that were sent to users, such as alerts about new properties, payment status updates, or changes made by the admin. Each entry in the sent log shows the notification text, target user, and timestamp, giving a clear history of communication. From the same screen, the admin can create a new broadcast by choosing the audience (all users, tenants only, or landlords only) and writing a custom message that will be delivered instantly through the app.

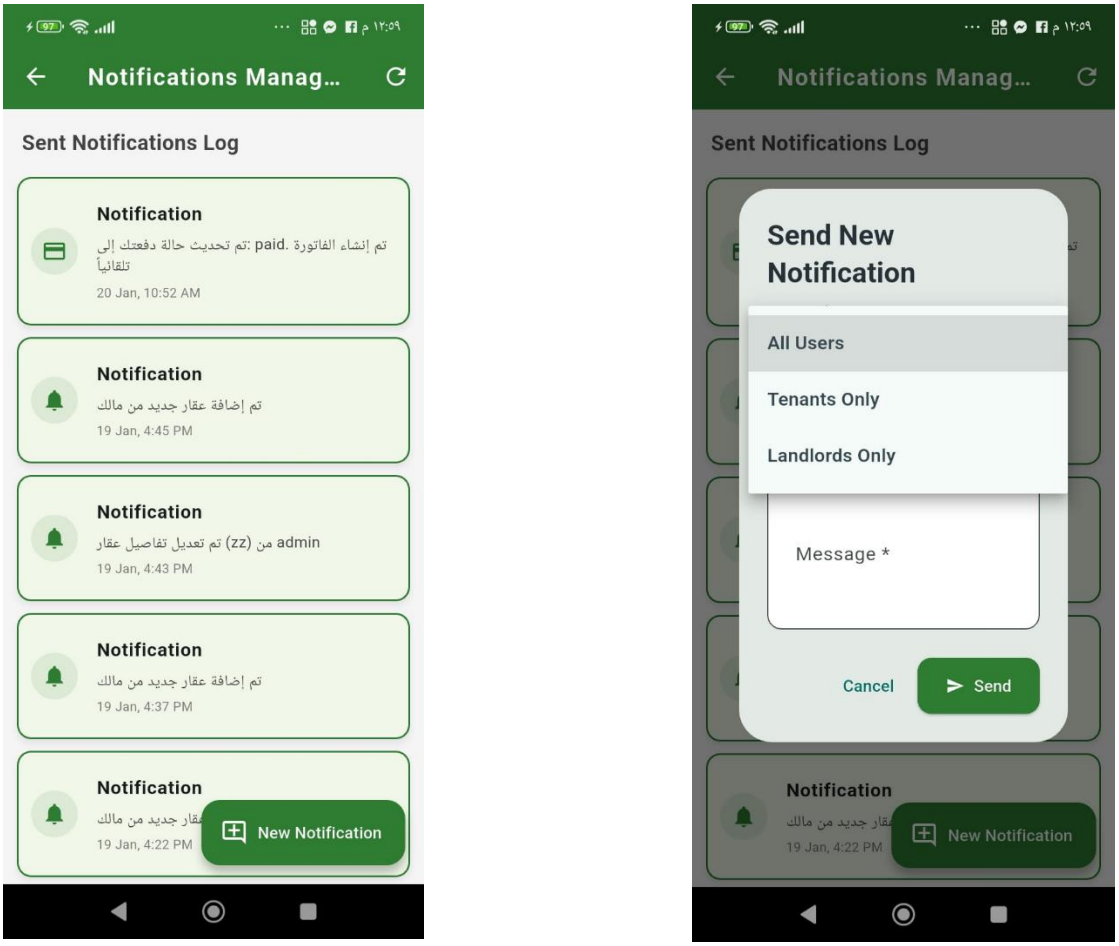


Figure32: Admin Notifications

The **Expenses Management** module helps the admin track all spending related to the properties in SHAQATI. The overview tab summarizes key indicators such as total expenses, highest single expense, average monthly spending, growth rate over time, most frequent expense type (for example, maintenance), and total number of expense entries, with an option to set a monthly budget target. The expenses tab lists each expense record with its type, date, and amount so the admin can review and audit detailed transactions. The charts tab visualizes expenses by property, trends over time, and distribution by type, making it easy to see where money is going and how costs are evolving month by month.

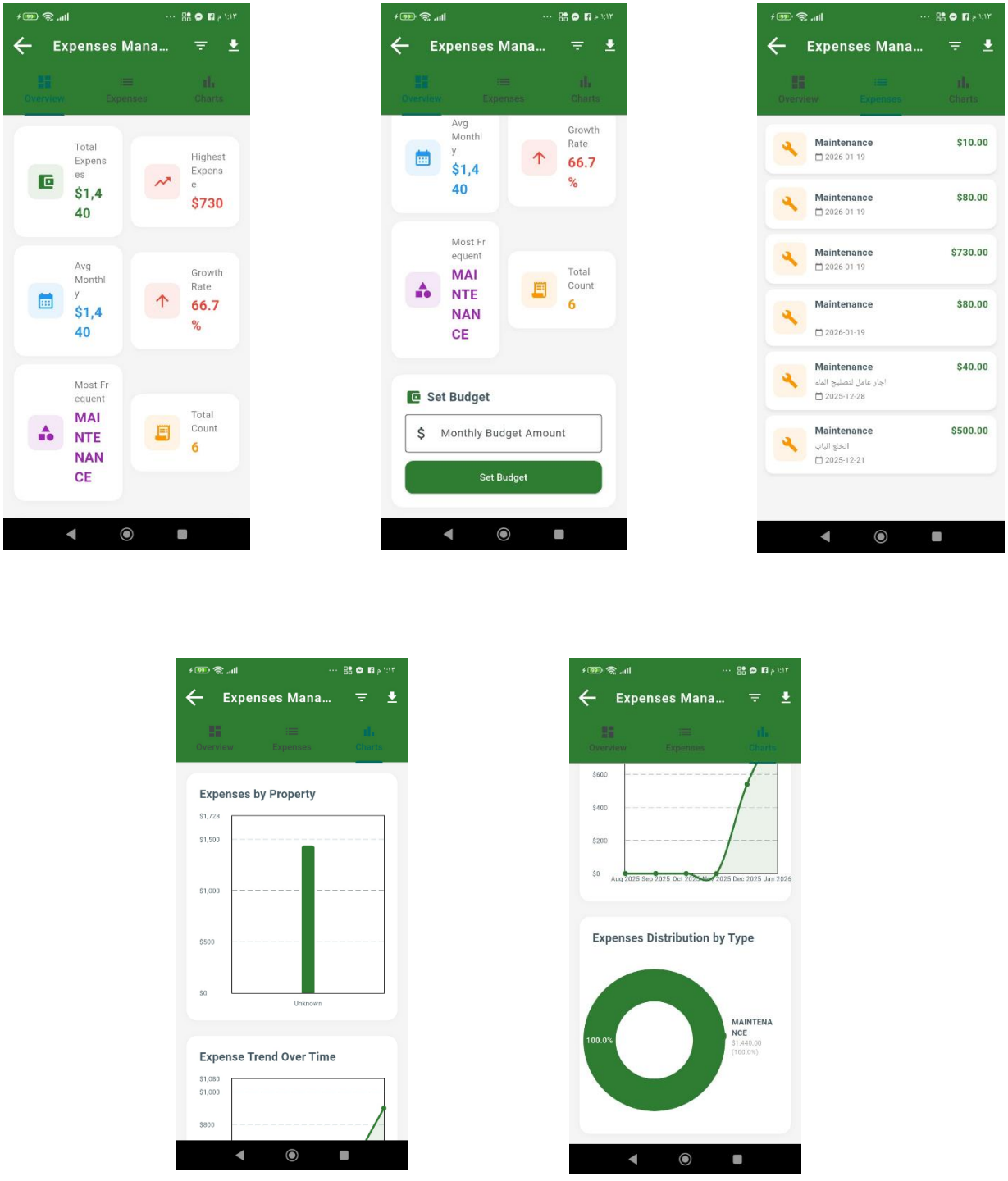


Figure33: Admin Expenses – overview, detailed expense list, and analytics

The **Invoices Management** screen gives the admin a complete picture of all invoices . The overview tab shows key KPIs such as total and paid invoice amounts, pending and overdue totals, average invoice value, and overall collection rate, along with recent invoice activity. In the invoices tab, the admin can search by invoice number and review each invoice with its amount, tenant, property, issue date, due date, and a quick action to send the invoice by email. The charts tab visualizes invoices by status (paid, pending, overdue), by tenant, and over time, helping the admin analyze cash flow and collection performance for the entire portfolio.

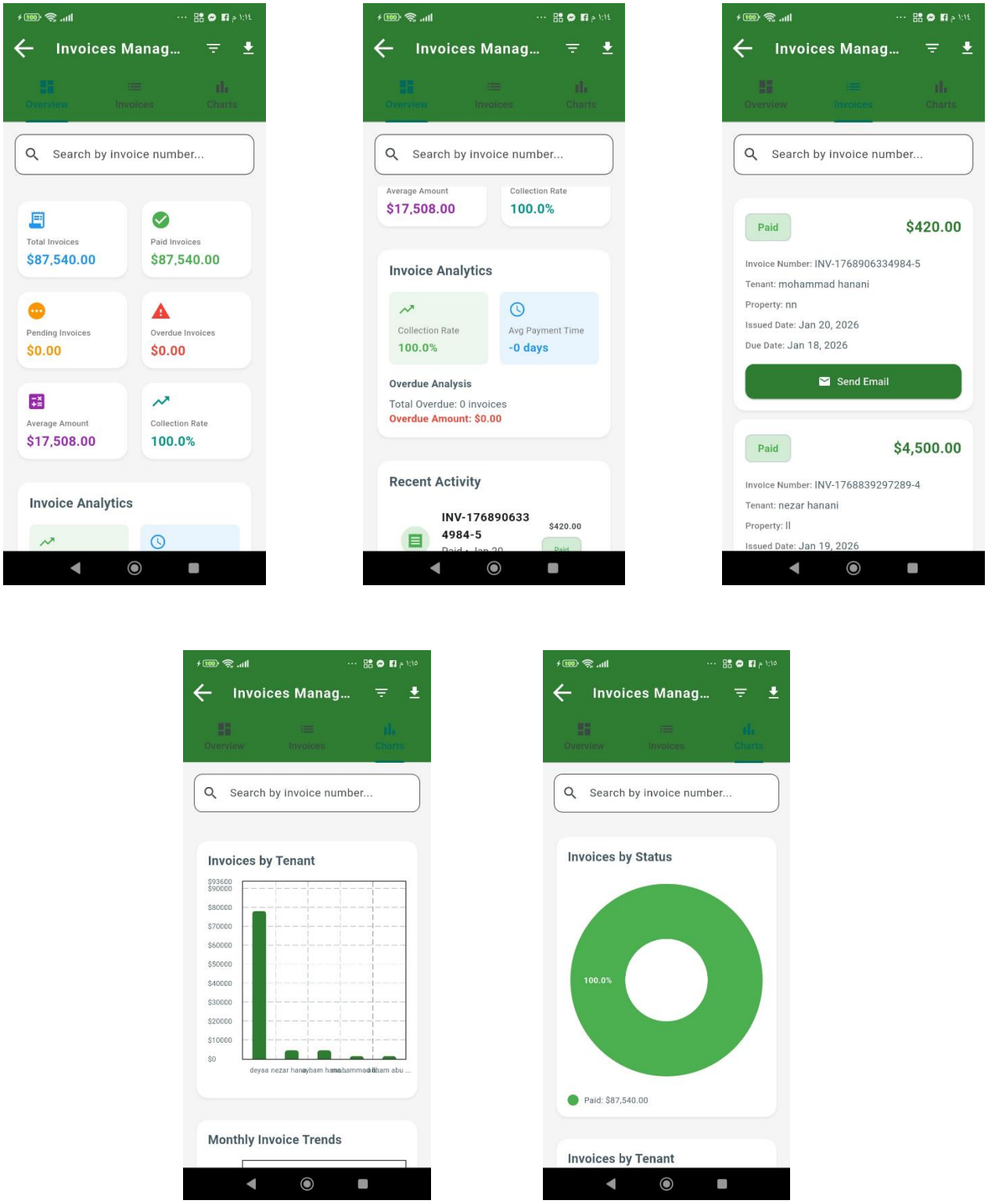


Figure34: Admin Invoices – overview of invoice totals, detailed list, and analytics

The **System Settings** screen allows the admin to configure global platform options such as application title, currency, pagination, date format, and logo. It also controls notification behavior, property defaults, and user management rules to adapt the system easily to different business needs.

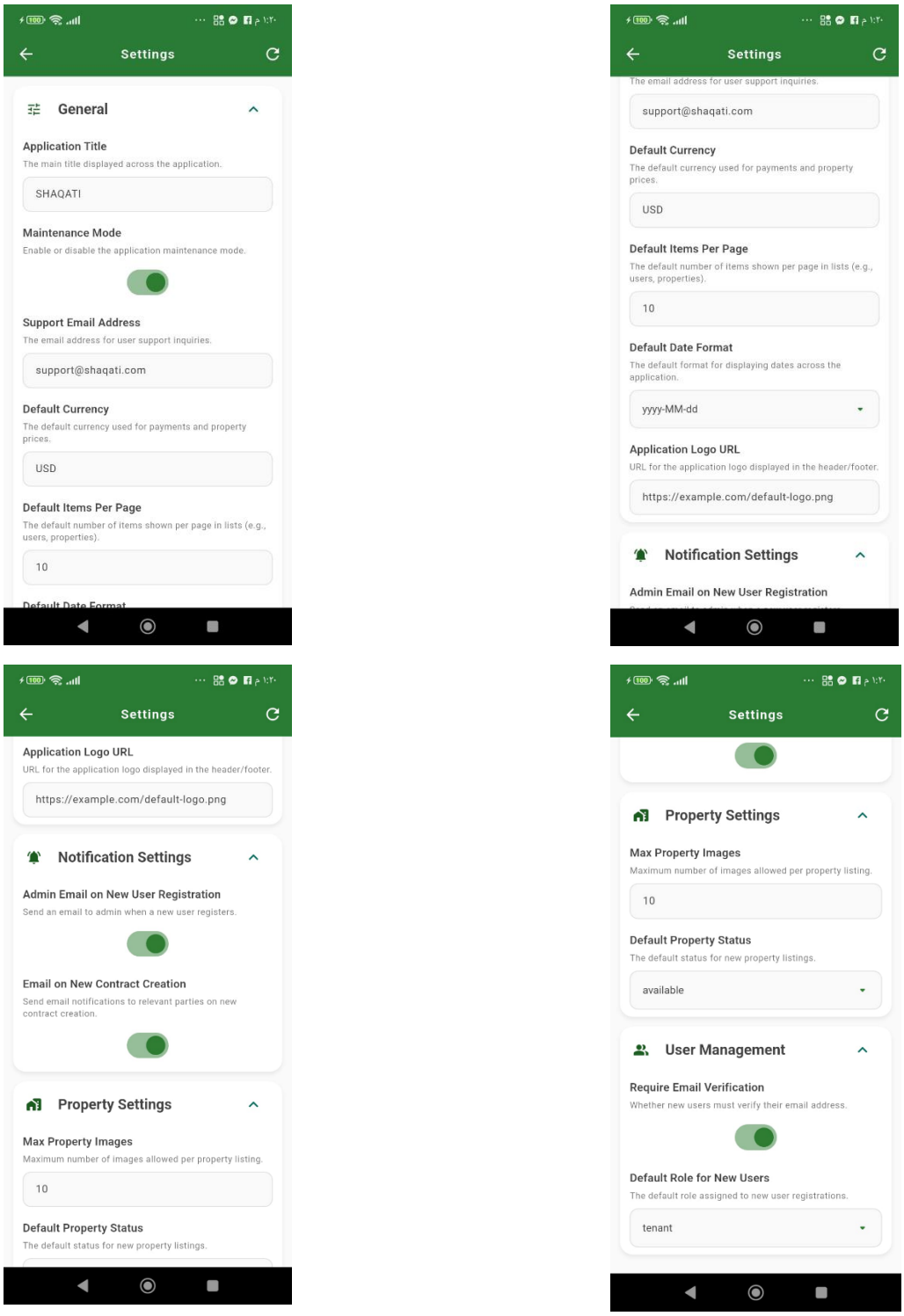
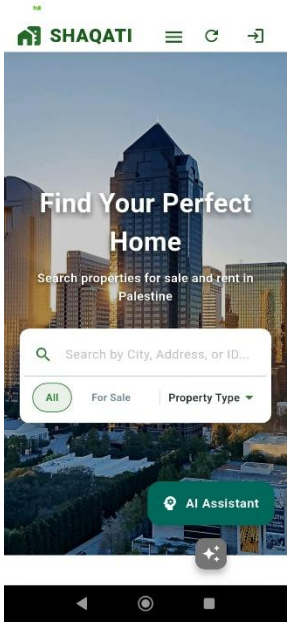
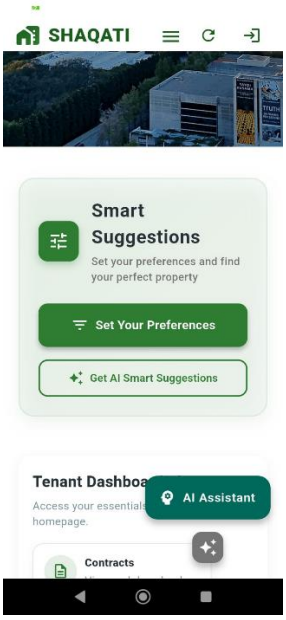


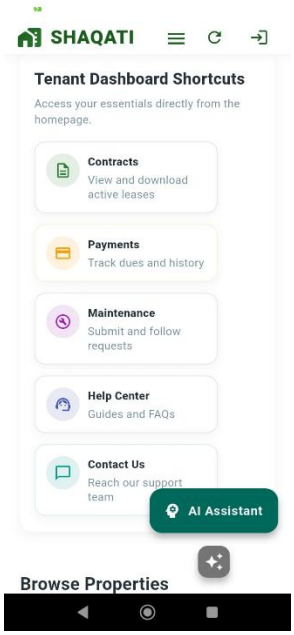
Figure35: Admin System Settings

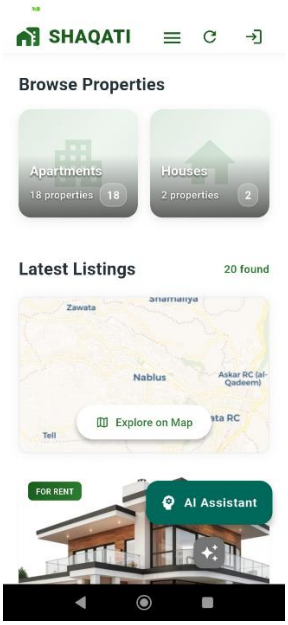
We move on to the role of the tenant

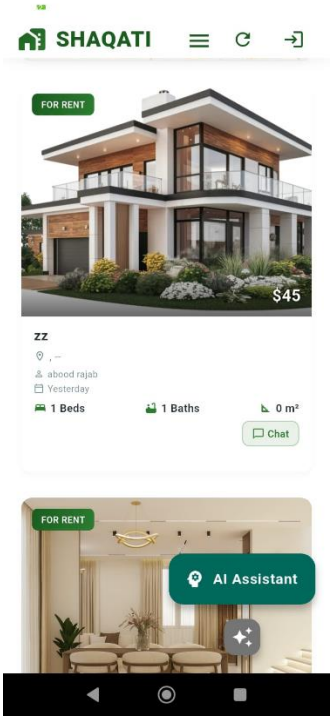
The **Tenant Home** experience provides a clear entry point to browse and discover properties, with quick filters for sale/rent, property types, and search by city/address/ID. Tenants can explore latest listings, jump to an interactive map view, and open property details pages showing key info such as price, location, bedrooms, bathrooms, and area. From listings and details, tenants can start a chat with the property landlord for inquiries. The home page also supports a guided experience through Smart Suggestions, where tenants can set preferences and request AI-based recommendations. Additionally, it offers helpful informational sections like FAQs, budget/cost guidance, neighborhood discovery, and service exploration, improving decision-making and user support.

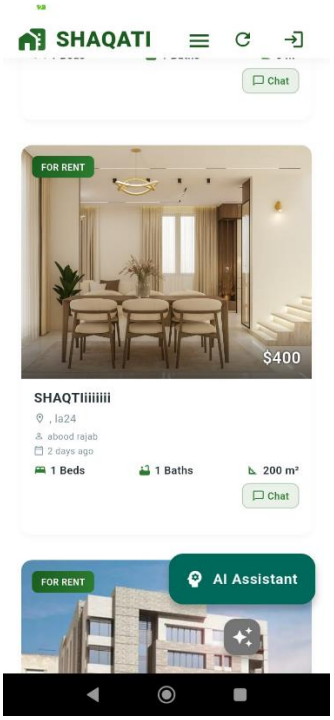


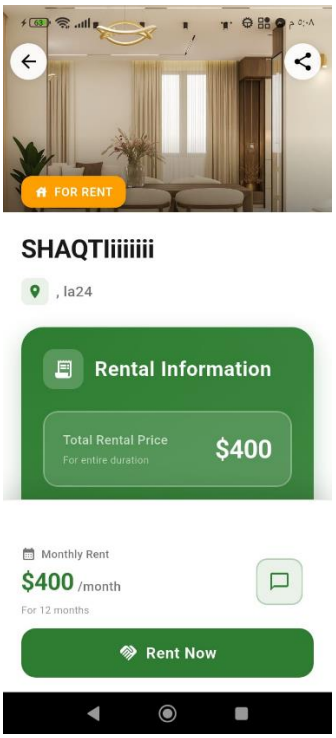


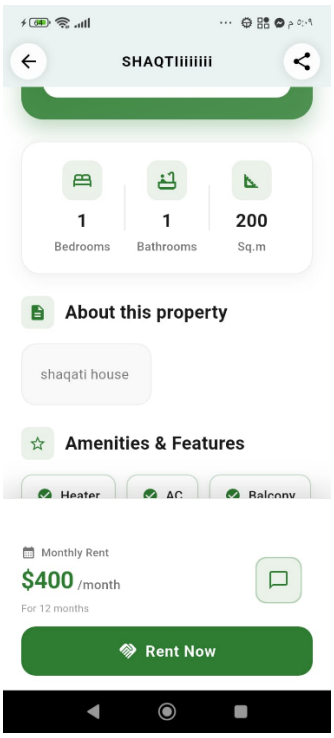


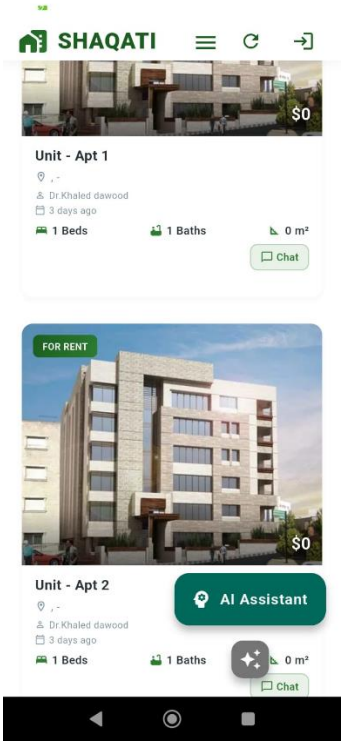


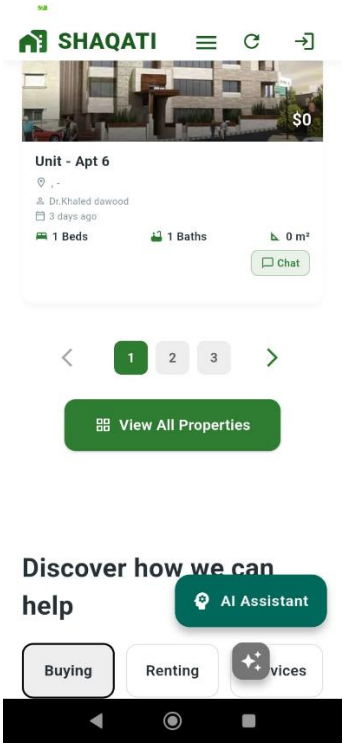

SHAQATI
FOR RENT
$45
zz
, --
abood rajab
Yesterday
1 Beds
1 Baths
0 m²
Chat
FOR RENT
AI Assistant

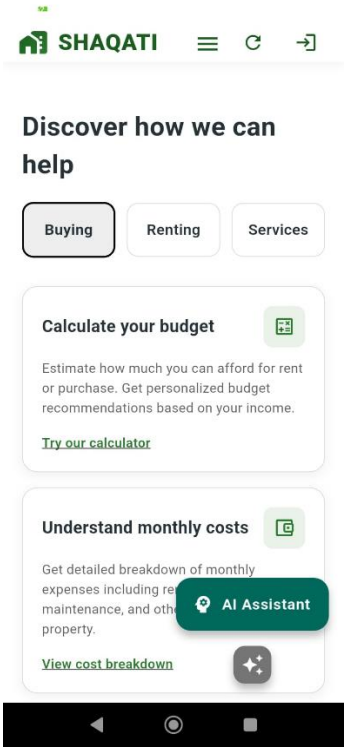

SHAQATI
Chat
FOR RENT
$400
SHAQTIiiiiiii
, la24
abood rajab
2 days ago
1 Beds
1 Baths
200 m²
Chat
FOR RENT
AI Assistant

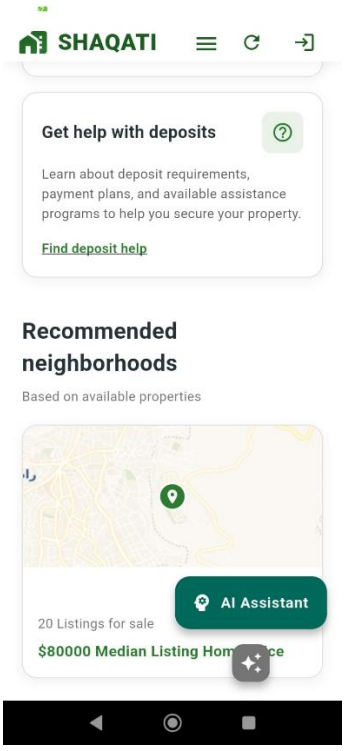

FOR RENT
SHAQTIiiiiiii
, la24
Rental Information
Total Rental Price
For entire duration
$400
Monthly Rent
$400 /month
For 12 months
Rent Now

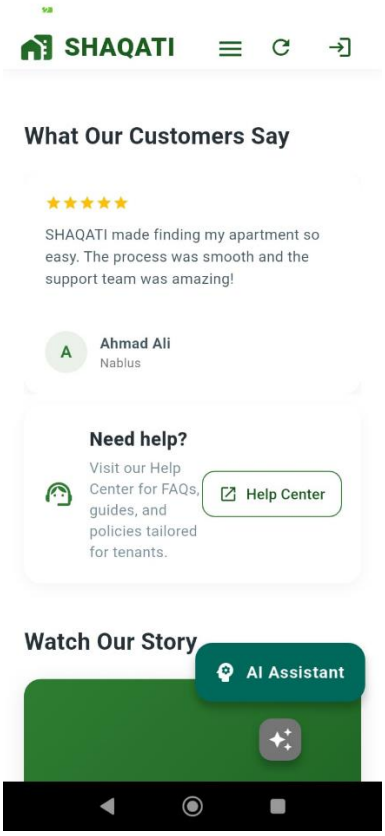

SHAQTIiiiiiii
1
Bedrooms
1
Bathrooms
200
Sq.m
About this property
shaqati house
Amenities & Features
Heater
AC
Balcony
Monthly Rent
$400 /month
For 12 months
Rent Now

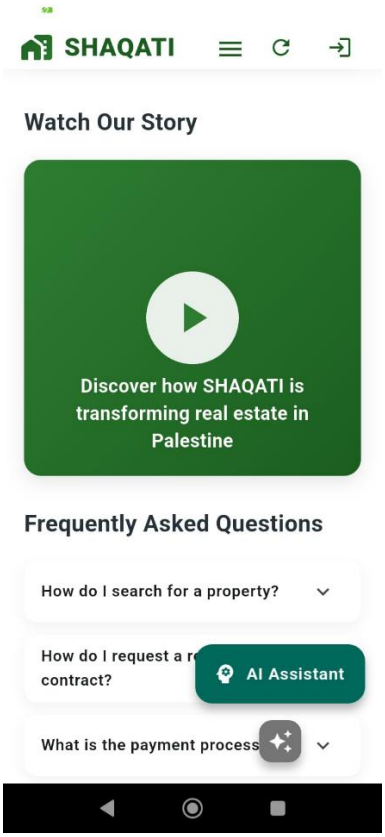

SHAQATI
$0
Unit - Apt 1
Dr.Khaled dawood
3 days ago
1 Beds
1 Baths
0 m²
Chat
FOR RENT
$0
Unit - Apt 2
Dr.Khaled dawood
3 days ago
1 Beds
1 Baths
0 m²
Chat
AI Assistant

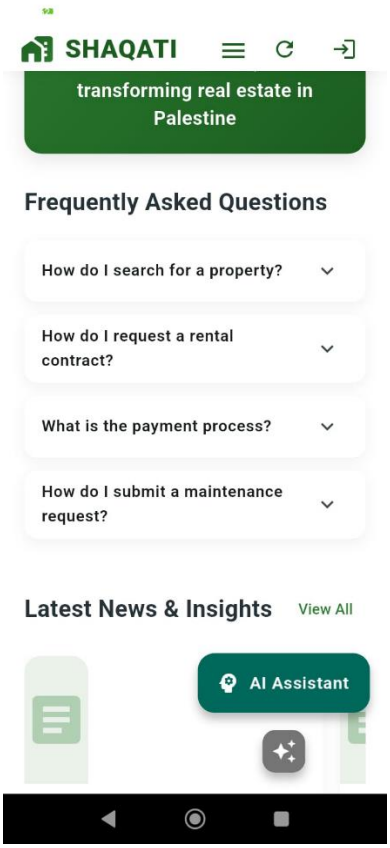

SHAQATI
$0
Unit - Apt 6
Dr.Khaled dawood
3 days ago
1 Beds
1 Baths
0 m²
Chat
1
2
3
View All Properties
Discover how we can help
AI Assistant
Buying
Renting

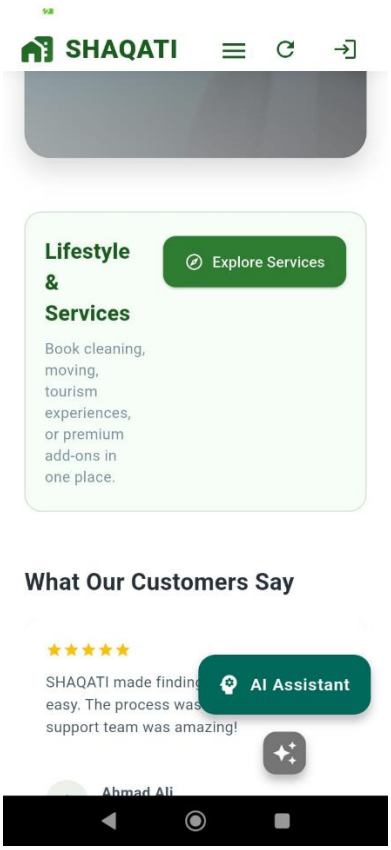

SHAQATI
Discover how we can help
Buying
Renting
Services
Calculate your budget
Estimate how much you can afford for rent or purchase. Get personalized budget recommendations based on your income.
Try our calculator
Understand monthly costs
Get detailed breakdown of monthly expenses including re
maintenance, and oth
property.
View cost breakdown
AI Assistant

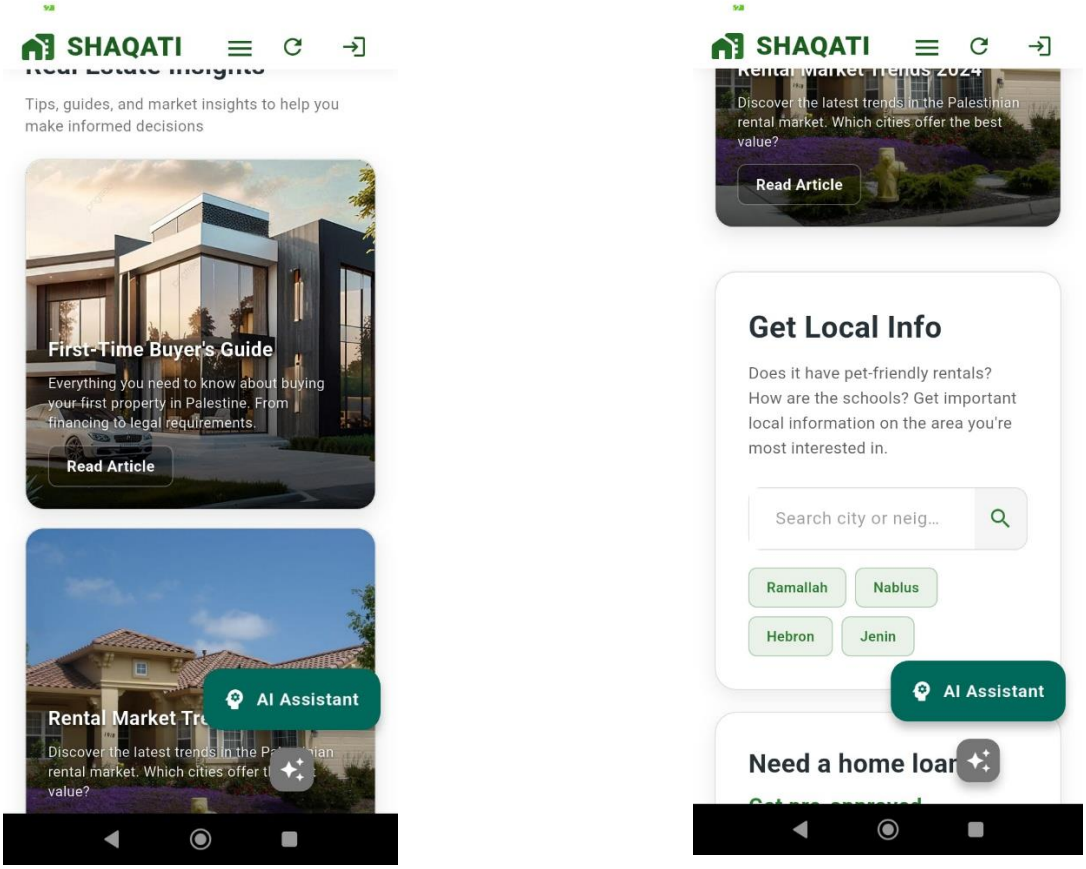

SHAQATI
Get help with deposits
Learn about deposit requirements, payment plans, and available assistance programs to help you secure your property.
Find deposit help
Recommended neighborhoods
Based on available properties
20 Listings for sale
$80000 Median Listing Hom
AI Assistant


SHAQATI
What Our Customers Say
★★★★★
SHAQATI made finding my apartment so easy. The process was smooth and the support team was amazing!
A
Ahmad Ali
Nablus
Need help?
Visit our Help Center for FAQs, guides, and policies tailored for tenants.
Help Center
Watch Our Story
AI Assistant


SHAQATI
Watch Our Story
Discover how SHAQATI is transforming real estate in Palestine
Frequently Asked Questions
How do I search for a property?
How do I request a r
contract?
AI Assistant
What is the payment process


SHAQATI
transforming real estate in Palestine
Frequently Asked Questions
How do I search for a property?
How do I request a rental contract?
What is the payment process?
How do I submit a maintenance request?
Latest News & Insights
View All
AI Assistant


SHAQATI
Lifestyle & Services
Explore Services
Book cleaning, moving, tourism experiences, or premium add-ons in one place.
What Our Customers Say
★★★★★
SHAQATI made findin
easy. The process was
support team was amazing!
AI Assistant
Ahmad Ali



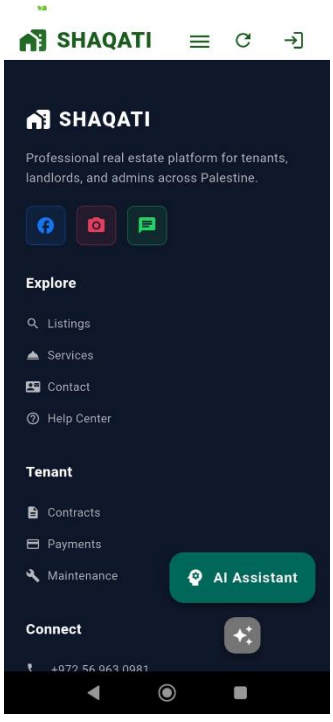


Figure36: Tenant home page: search, browse, chat

AI System

SHAQATI integrates an AI assistant to support tenants, landlords, and administrators. The system uses Retrieval-Augmented Generation (RAG), intent detection, and personalized recommendations.

Core Capabilities

1. **Intelligent Property Recommendations**

   The system analyzes user behavior, preferences, and search history to suggest properties. Recommendations consider budget, location, property type, and user activity.

2. **Natural Language Processing**

   Users can interact in natural language (Arabic/English). The system detects intent and routes queries to appropriate functions, such as property search, contract management, payment tracking, and maintenance requests.

3. **Context-Aware Assistance**

   The assistant adapts responses by user role (Tenant, Landlord, Admin) and maintains session context for follow-up questions.

4. **4. Database Integration**

   The AI queries live data from MongoDB, including properties, contracts, payments, and maintenance requests, to provide accurate, up-to-date information.

5. **5. Knowledge Base Integration**

   The system uses project documentation (API routes, database schema, features) to answer questions and guide users.

Technical Architecture

The AI system consists of:

Frontend: Flutter-based chat interface with session memory

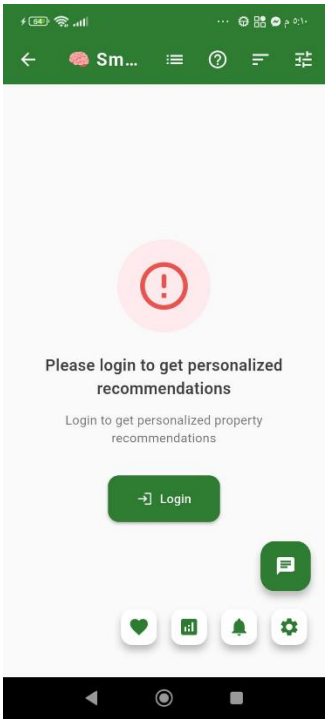
Backend: Node.js/Express API with intent detection and RAG processing

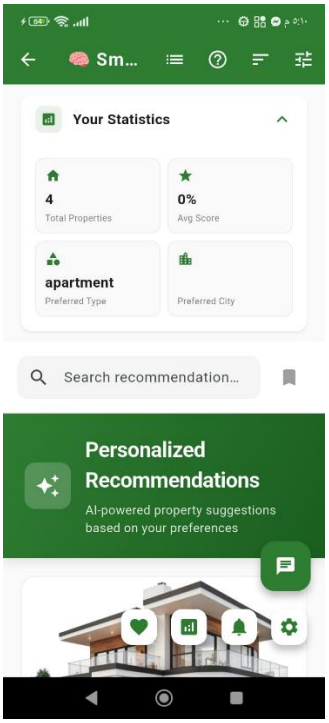
AI Providers: OpenAI GPT models with fallback support

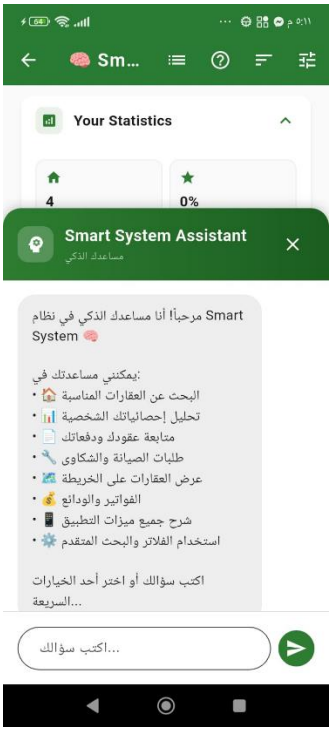
Knowledge Base: Structured documentation files for accurate responses

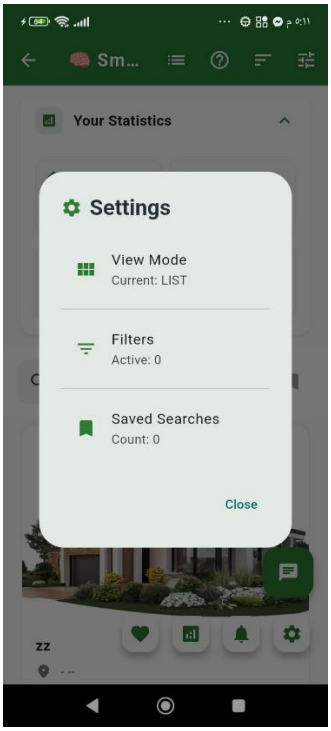
Benefits For Tenants:

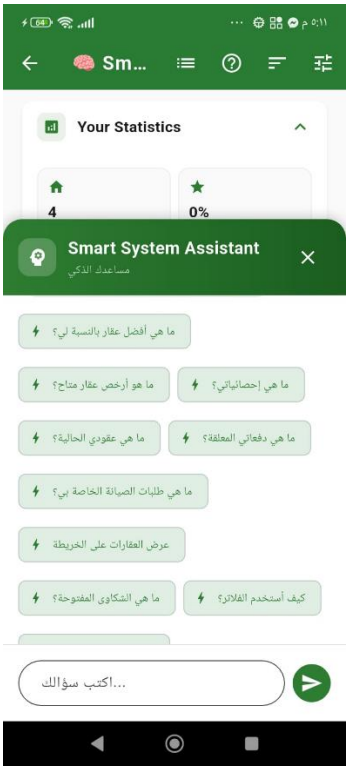
- Personalized property recommendations
- Natural language property search
- Contract and payment tracking assistance
- Maintenance request guidance
- Investment and market insights

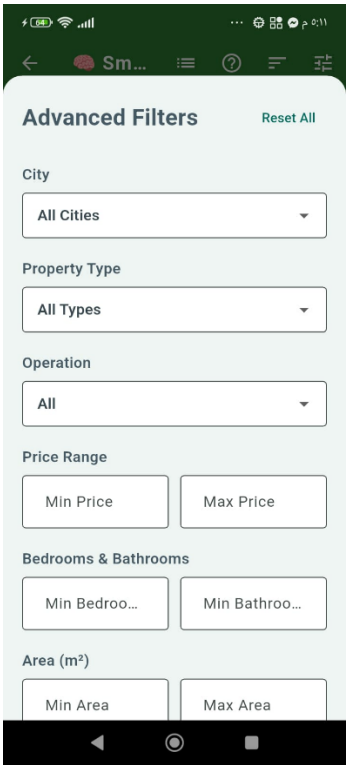

Sm...
Please login to get personalized recommendations
Login to get personalized property recommendations
Login

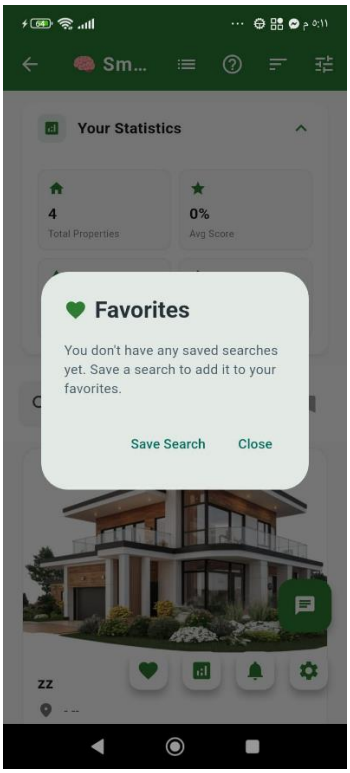

Sm...
Your Statistics
4
Total Properties
0%
Avg Score
apartment
Preferred Type
Preferred City
Search recommendation...
Personalized Recommendations
AI-powered property suggestions based on your preferences

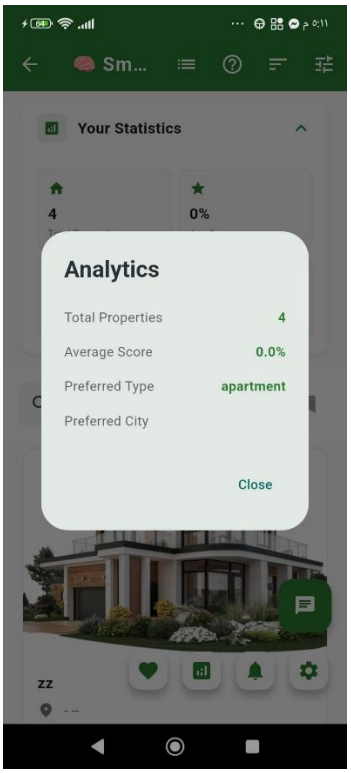

Sm...
Your Statistics
4
0%
Smart System Assistant
مساعدك الذكي
مرحباً! أنا مساعدك الذكي في نظام Smart System 🧠
يمكنني مساعدتك في:
• 🏠 البحث عن العقارات المناسبة
• 📊 تحليل إحصائياتك الشخصية
• 📄 متابعة عقودك ودفعاتك
• 🔧 طلبات الصيانة والشكاوى
• 🗺️ عرض العقارات على الخريطة
• 💰 الفواتير والودائع
• 📱 شرح جميع ميزات التطبيق
• ⚙️ استخدام الفلاتر والبحث المتقدم
اكتب سؤالك أو اختر أحد الخيارات السريعة...
اكتب سؤالك...

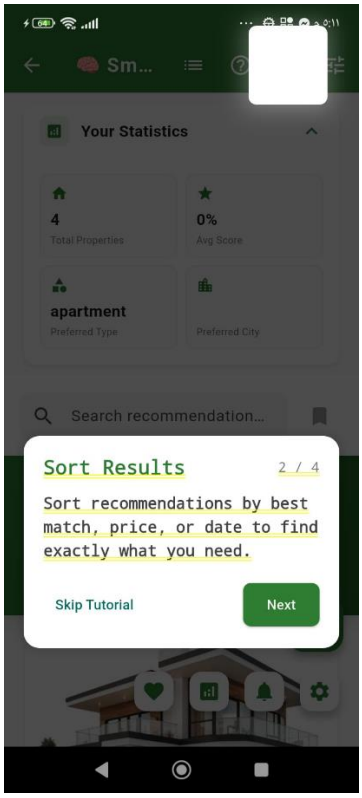

Sm...
Your Statistics
Settings
View Mode
Current: LIST
Filters
Active: 0
Saved Searches
Count: 0
Close
ZZ


Your Statistics
4
0%
Smart System Assistant
مساعدك الذكي
ما هي أفضل عقار بالنسبة لي؟
ما هو أرخص عقار متاح؟
ما هي إحصائياتي؟
ما هي عقودي الحالية؟
ما هي دفعاتي المعلقة؟
ما هي طلبات الصيانة الخاصة بي؟
عرض العقارات على الخريطة
ما هي الشكاوى المفتوحة؟
كيف أستخدم الفلاتر؟
...اكتب سؤالك

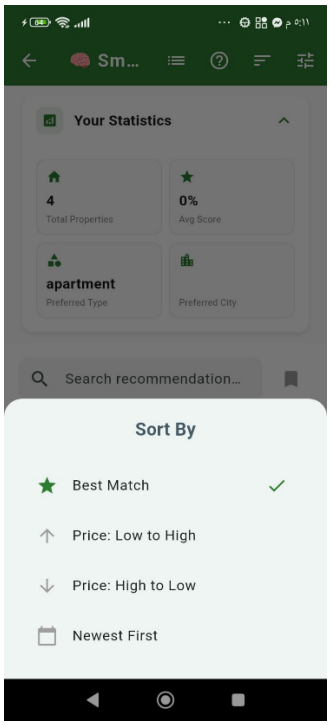

Advanced Filters
Reset All
City
All Cities
Property Type
All Types
Operation
All
Price Range
Min Price
Max Price
Bedrooms & Bathrooms
Min Bedroo...
Min Bathroo...
Area (m²)
Min Area
Max Area

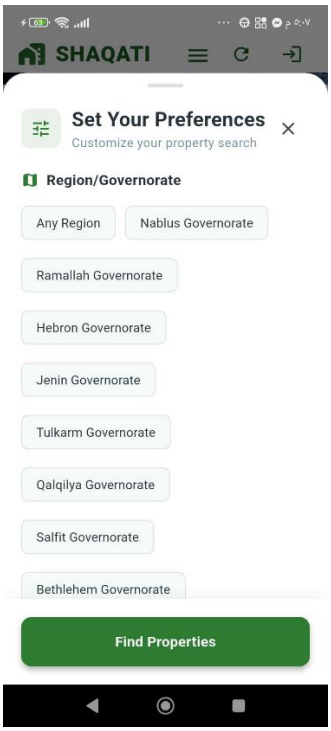

Your Statistics
4
Total Properties
0%
Avg Score
Favorites
You don't have any saved searches yet. Save a search to add it to your favorites.
Save Search
Close
zz

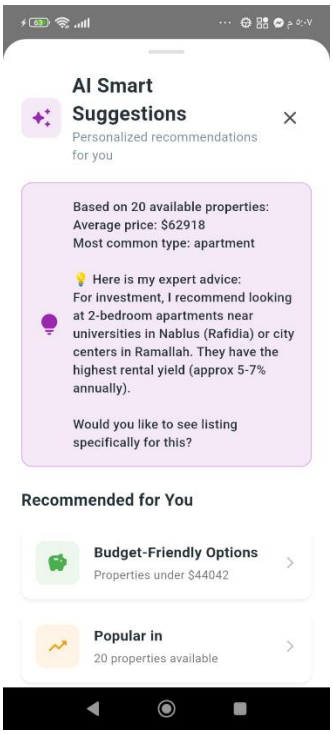

Your Statistics
4
0%
Analytics
Total Properties
4
Average Score
0.0%
Preferred Type
apartment
Preferred City
Close
zz


Your Statistics
4
Total Properties
0%
Avg Score
apartment
Preferred Type
Preferred City
Search recommendation...
Sort Results
2 / 4
Sort recommendations by best match, price, or date to find exactly what you need.
Skip Tutorial
Next


AI Assistant
مرحباً! أنا مساعدك الذكي
يمكنني مساعدتك في:
• تحليل بنية مشروع SHAQATI
• الإجابة على أسئلة حول الكود
• اقتراح تحسينات
• شرح الميزات والوظائف
• حل المشاكل التقنية
اكتب سؤالك أو اختر أحد الخيارات السريعة...
حلل بنية مشروع SHAQATI
اشرح نظام العقود
كيف يعمل نظام الدفعات؟
ما هي الأدوار في النظام؟
اكتب سؤالك...


Your Statistics
4
Total Properties
0%
Avg Score
apartment
Preferred Type
Preferred City
Search recommendation...
Sort By
Best Match
Price: Low to High
Price: High to Low
Newest First


SHAQATI
Set Your Preferences
Customize your property search
Region/Governorate
Any Region
Nablus Governorate
Ramallah Governorate
Hebron Governorate
Jenin Governorate
Tulkarm Governorate
Qalqilya Governorate
Salfit Governorate
Bethlehem Governorate
Find Properties


AI Smart Suggestions
Personalized recommendations for you
Based on 20 available properties:
Average price: $62918
Most common type: apartment
Here is my expert advice:
For investment, I recommend looking at 2-bedroom apartments near universities in Nablus (Rafidia) or city centers in Ramallah. They have the highest rental yield (approx 5-7% annually).
Would you like to see listing specifically for this?
Recommended for You
Budget-Friendly Options
Properties under $44042
Popular in
20 properties available

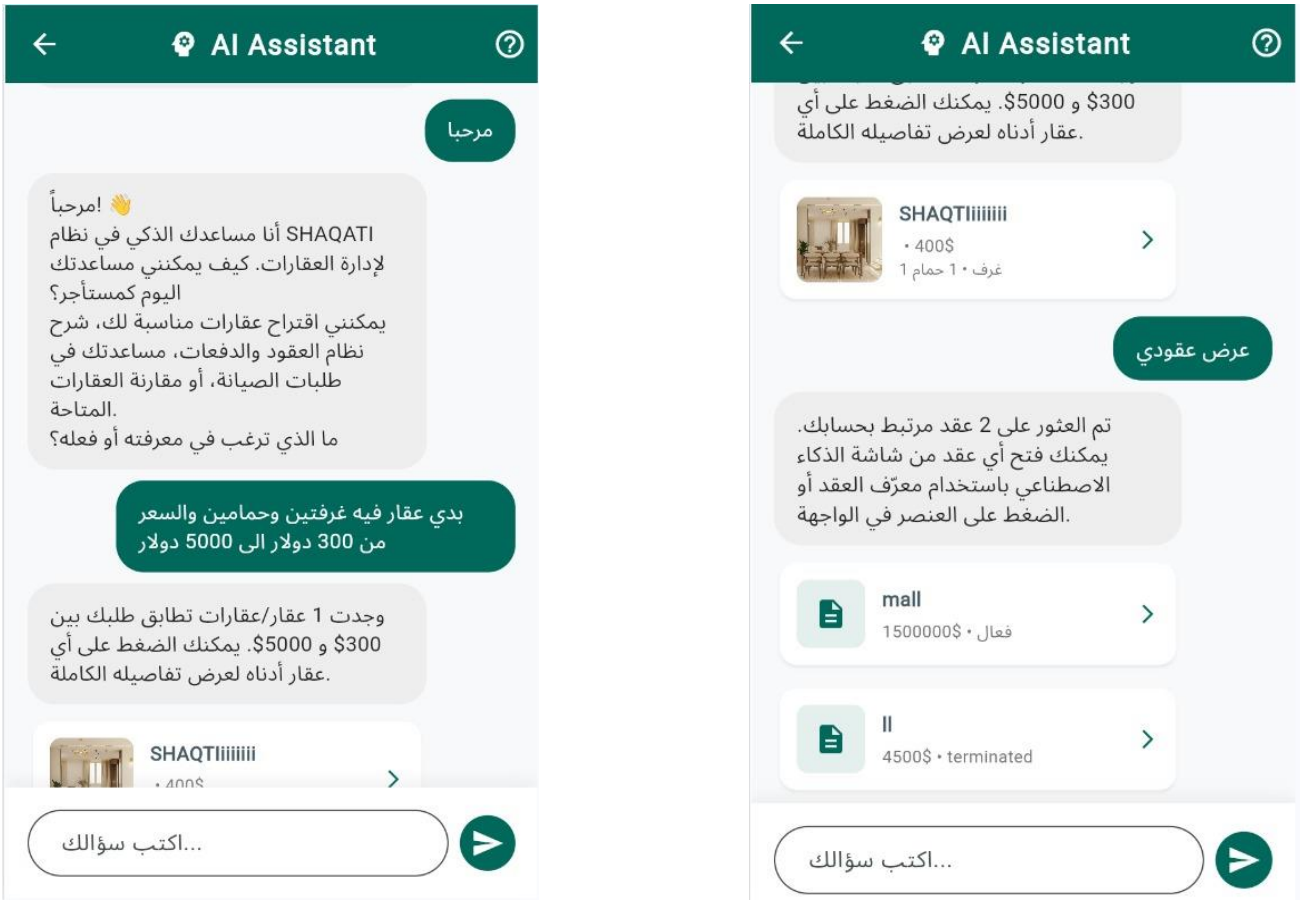


Figure37: Artificial Intelligence - Investment Tips in Real Estate

The **tenant dashboard** provides a unified view of account activity. It shows an overview with key metrics (active contracts, pending payments, expenses, deposits, maintenance requests), visual insights via charts for payments and maintenance, quick actions for common tasks, and a recent activity feed with transactions and requests. This helps tenants monitor finances, track maintenance, and access services quickly.

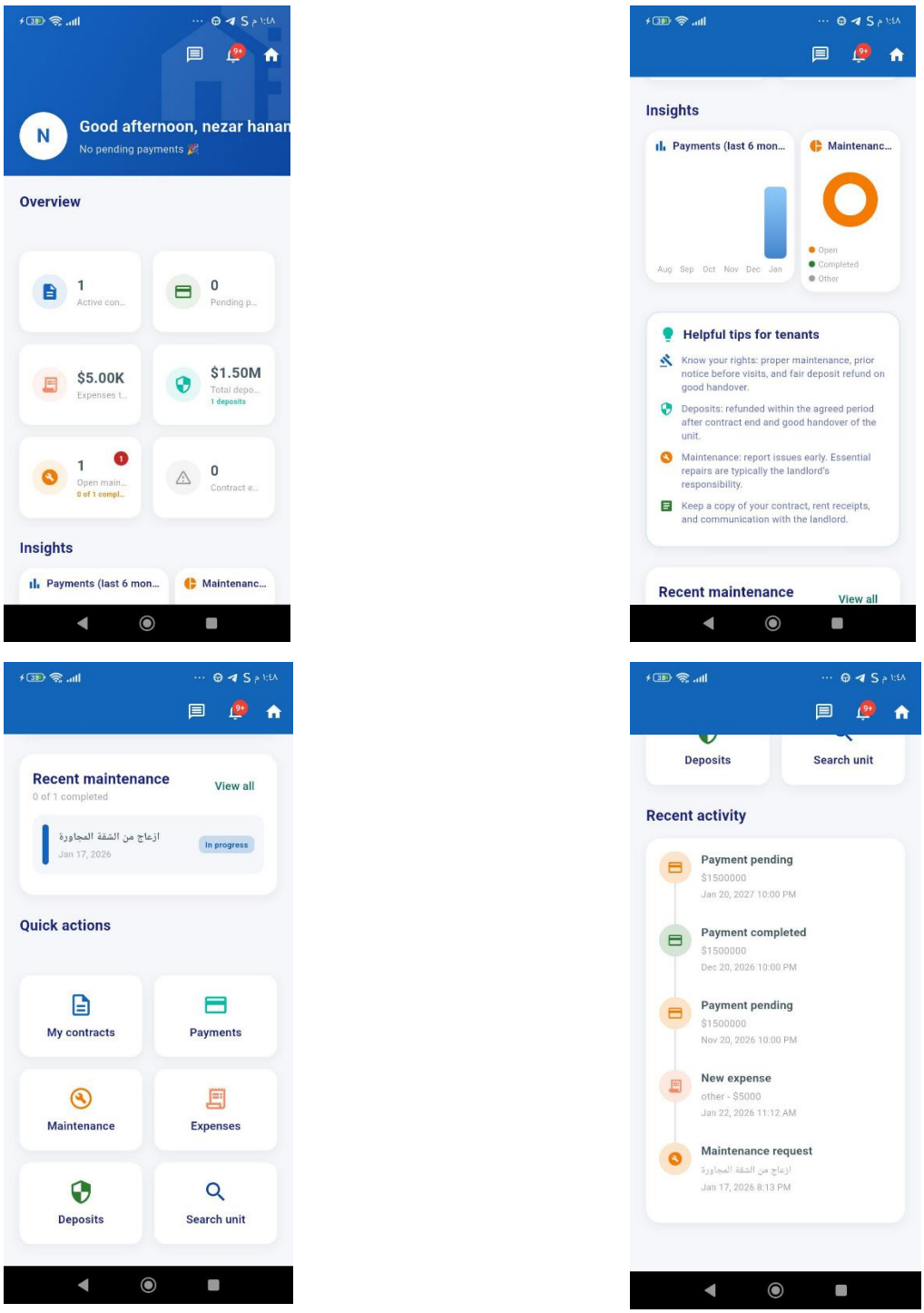
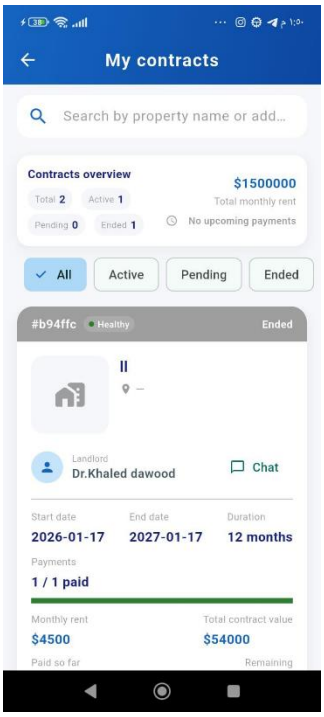
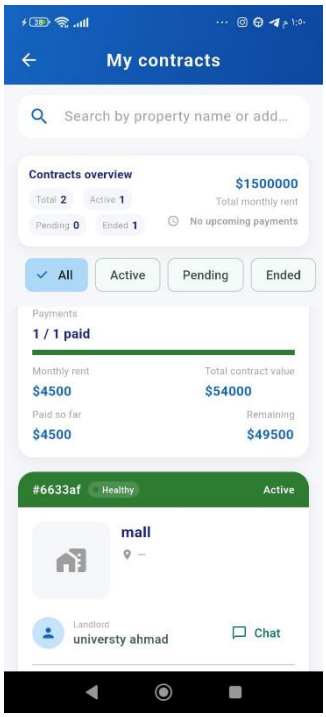


Figure38: tenants - dashboard

This **contract management** system provides for tenants and landlords. Users can view all contracts with status filters (active, pending, ended), track payment progress, see financial summaries (total monthly rent, contract value, paid/remaining amounts), and manage actions like auto-pay, deposits, and termination requests. The system sends notifications for payment updates, contract approvals, and status changes, keeping tenants informed of contract-related activities.





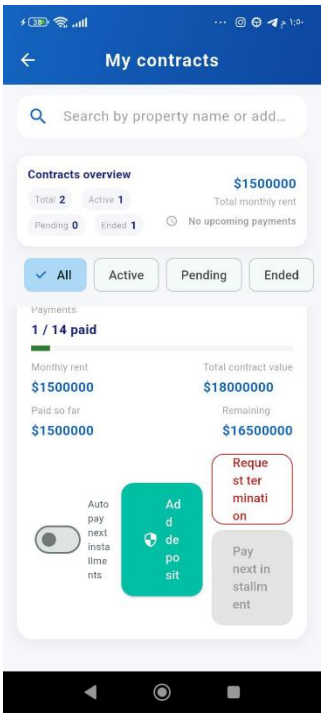


Figure39: Contract management and payment tracking

This **payment management** system provides the ability to track all transactions. Users can view payment status (paid, pending, overdue), total amounts paid, check outstanding balances, and access payment history. The user interface includes filters to organize payments by status, ensuring complete clarity regarding financial obligations and completed transactions.

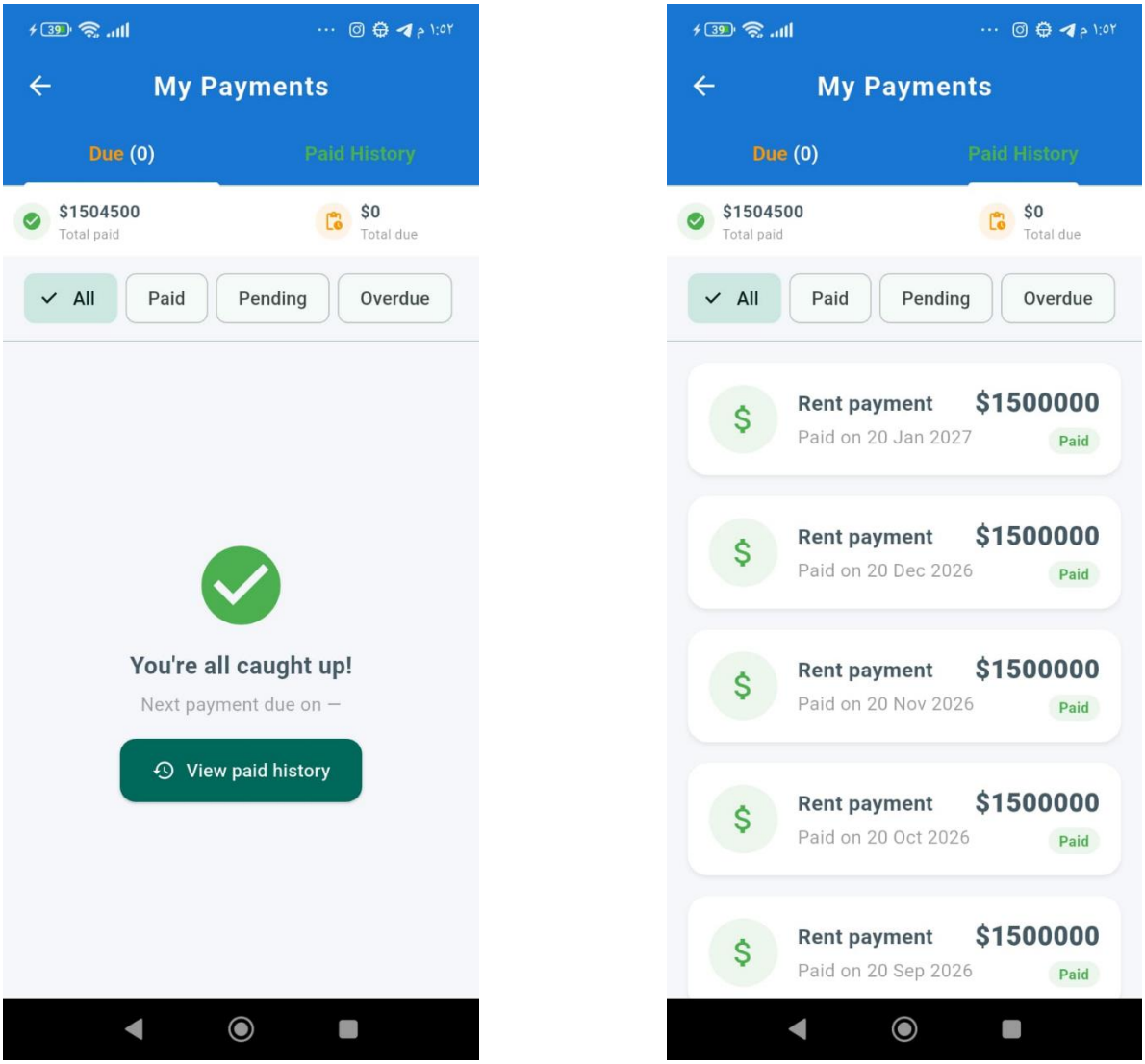


Figure40: Payment tracking and history management

Maintenance and complaints management screens showing request submission forms, status tracking, filtering options, and detailed request information with priority and severity indicators.

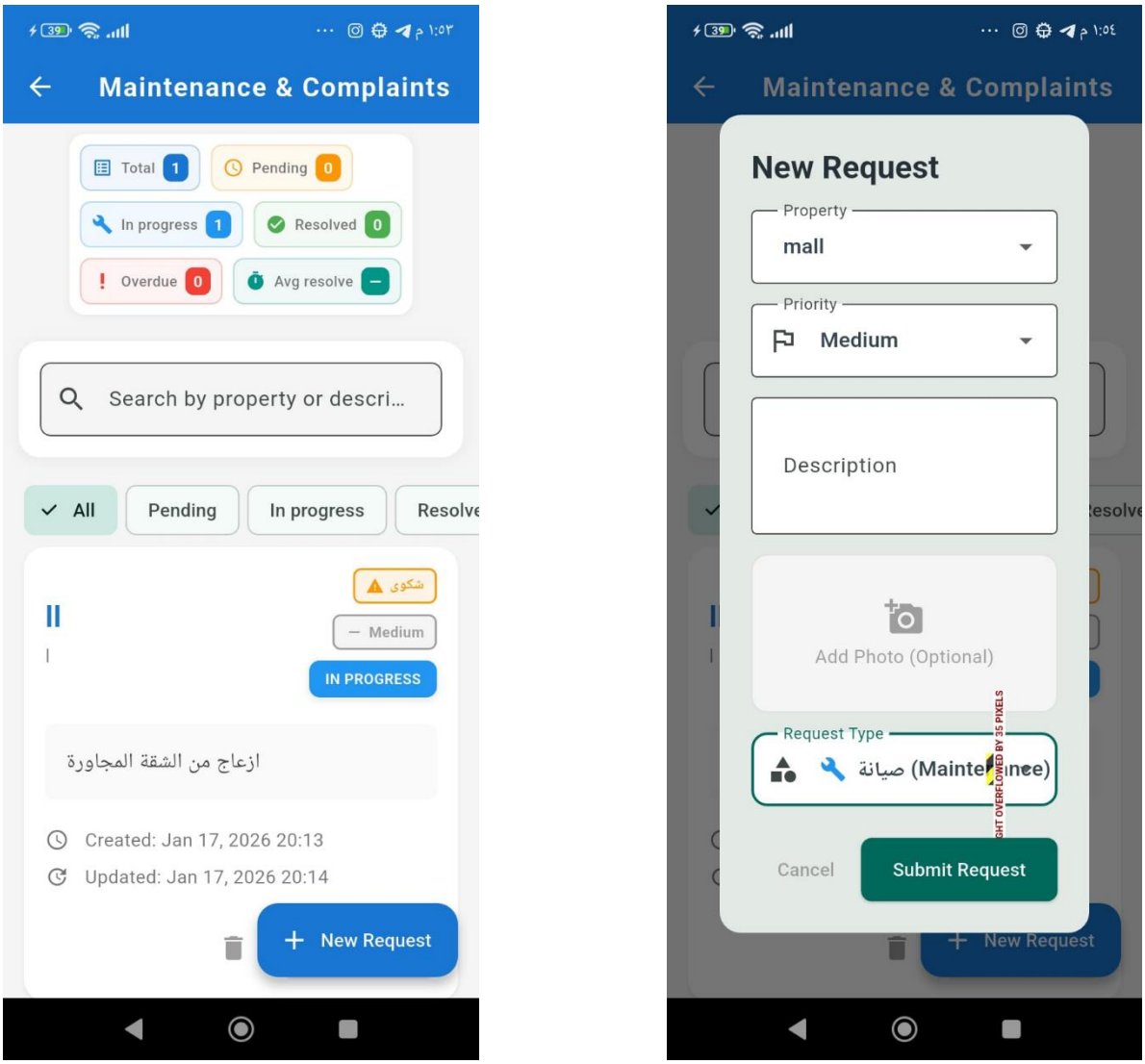


Figure41: Maintenance request and tracking system

The system offers comprehensive **expense management**, allowing tenant to track and categorize expenses. Tenants can record expenses, specifying categories, amounts, descriptions, and dates, as well as set monthly budgets and view comprehensive analytics including total expenses, highest expenditure, average monthly spending, and growth trends. The system also includes charts illustrating expense distribution by property type and trend over time. Tenants can filter and download expense data and upload receipts for documentation.

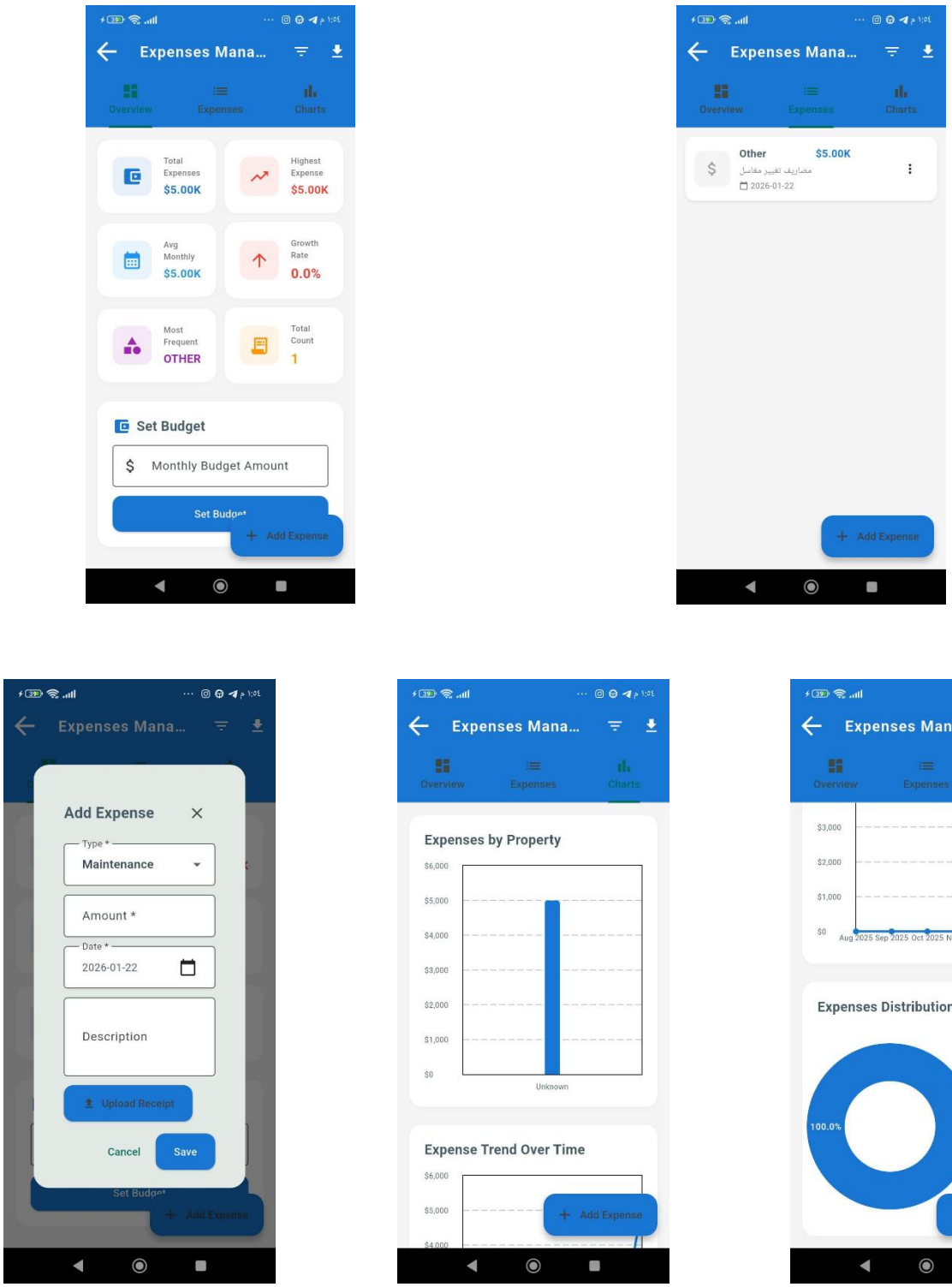


Figure42: Expenses tracking and budget management

The system provides comprehensive **deposit management**, enabling tracking and monitoring. Tenants can view key performance indicators (KPIs) such as total deposits, average deposit, refunds, and pending refunds, track deposits by property and status, and access data analytics, including refund rates and trends. The system also allows tenants to add new deposits linked to contracts, view details of each individual deposit with financial information (deducted amounts, refunds, and available funds), and perform actions such as withdrawing, modifying, or deleting deposits. Charts display the distribution of deposits by status and property, as well as trends over time.

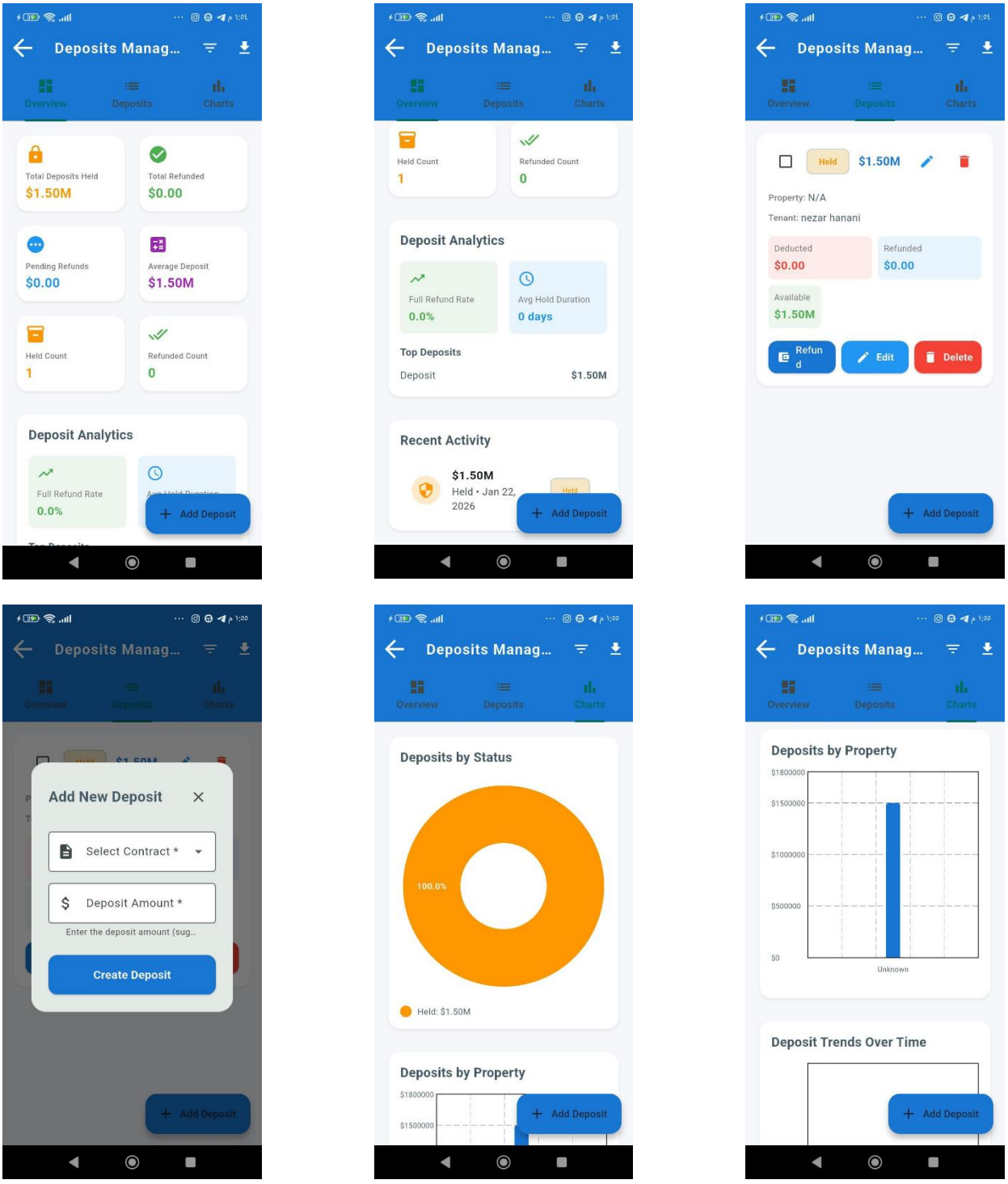
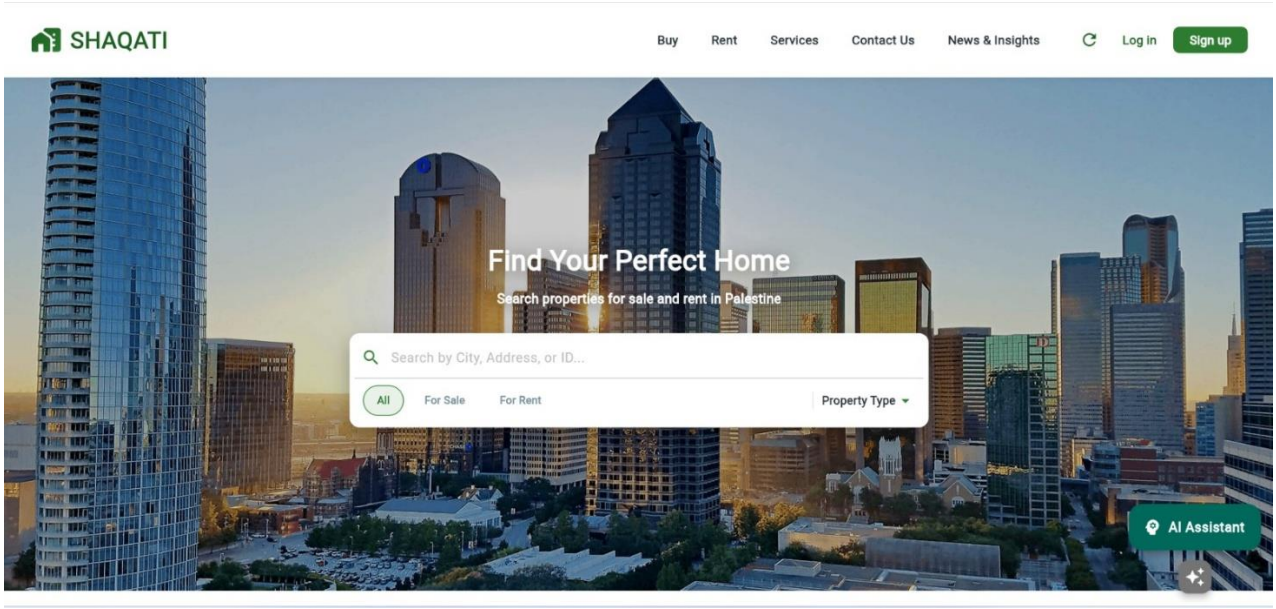
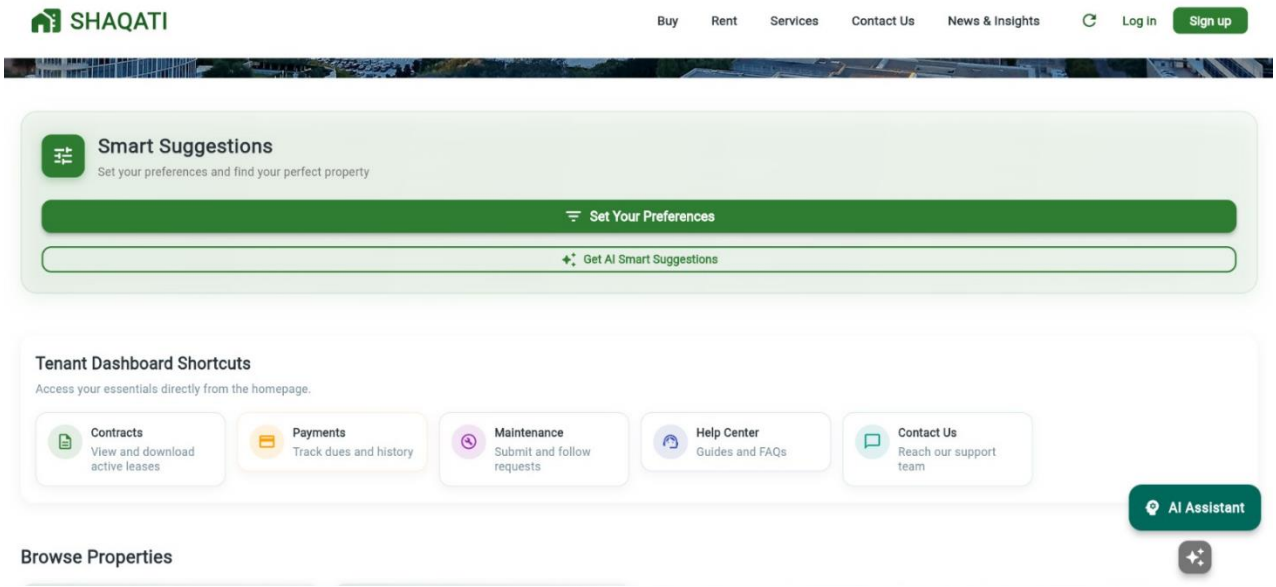


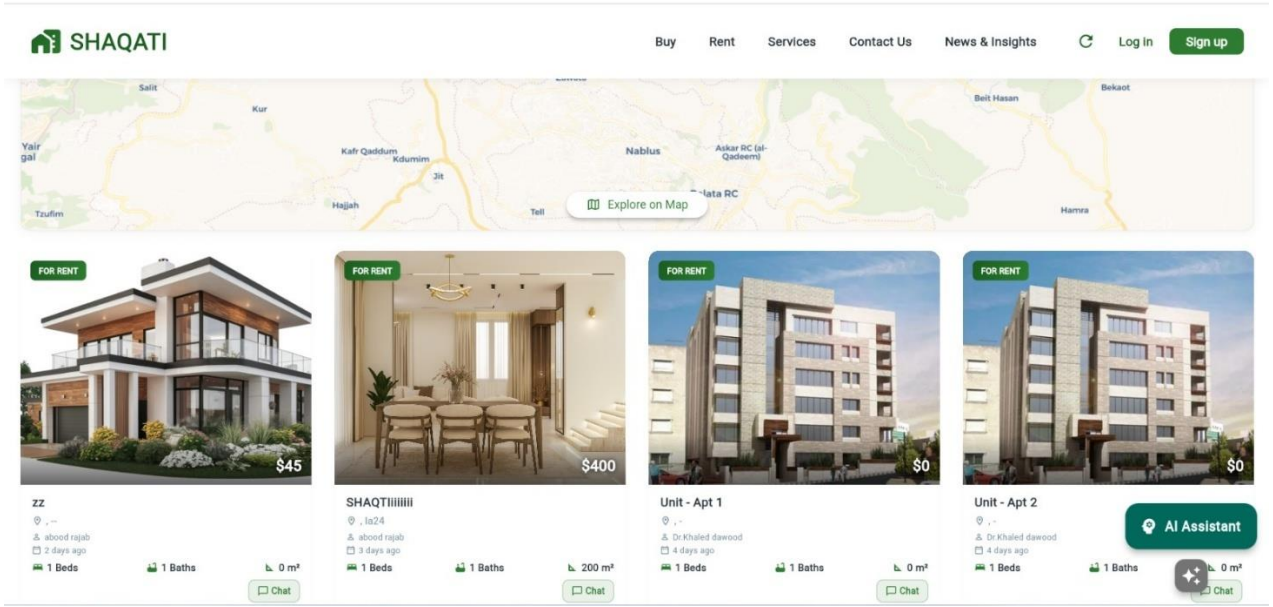
Figure43: Deposit tracking and refund management

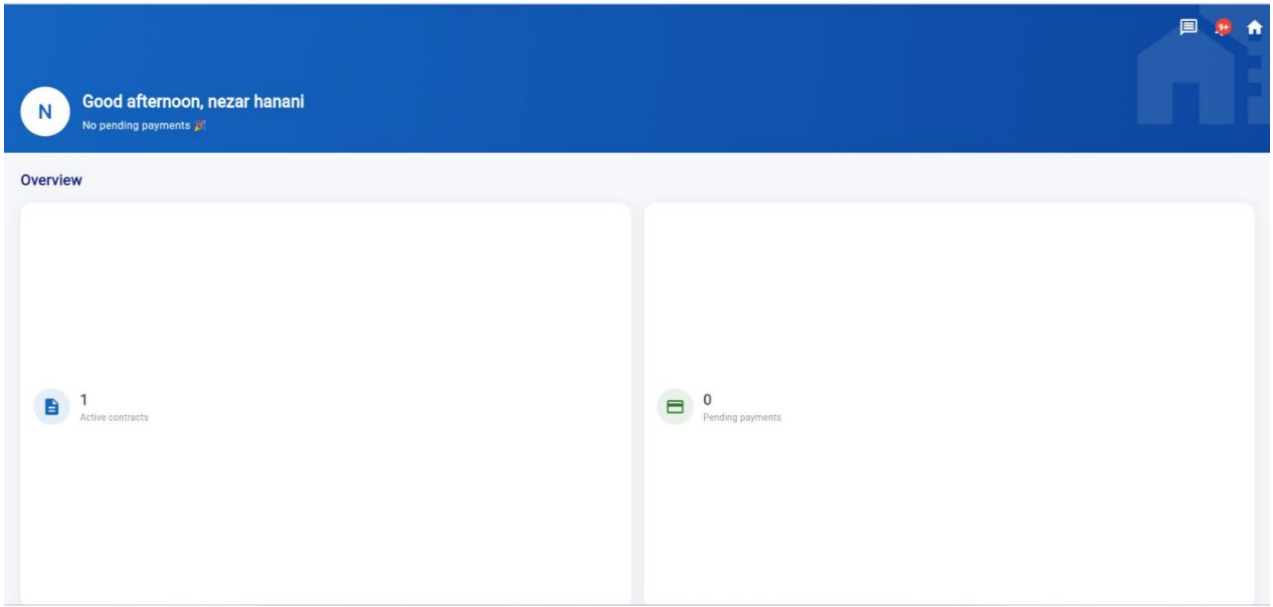
The main pages for the three roles in the project are on the website

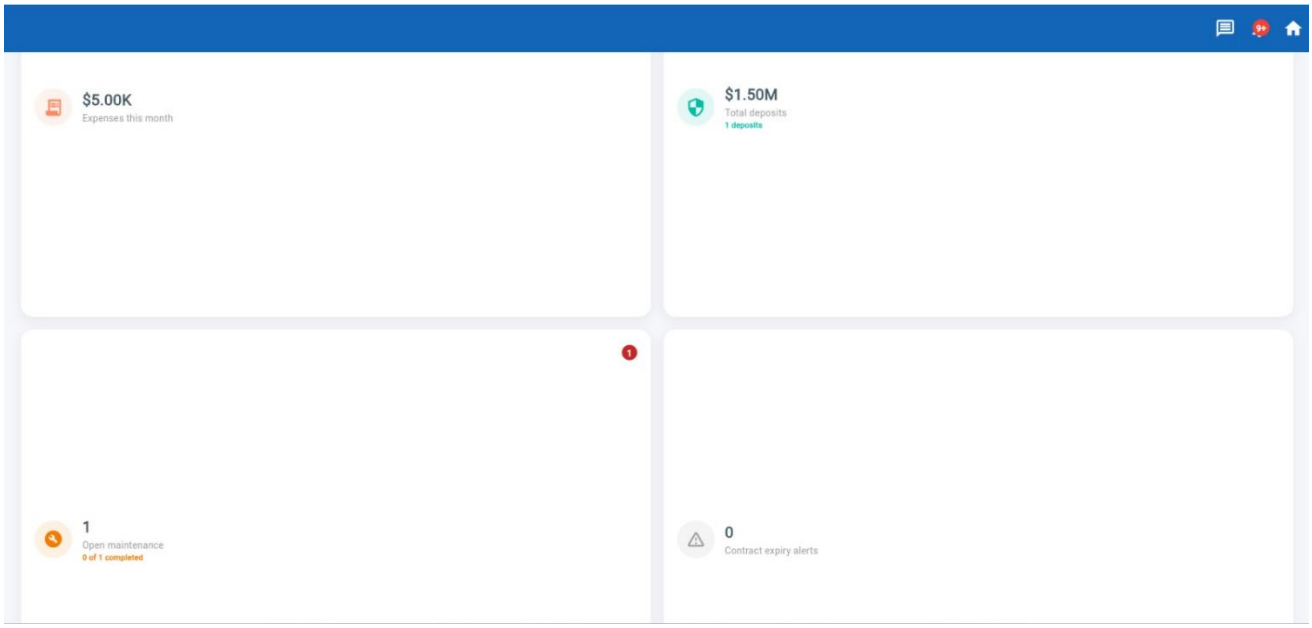
- **For tenant**

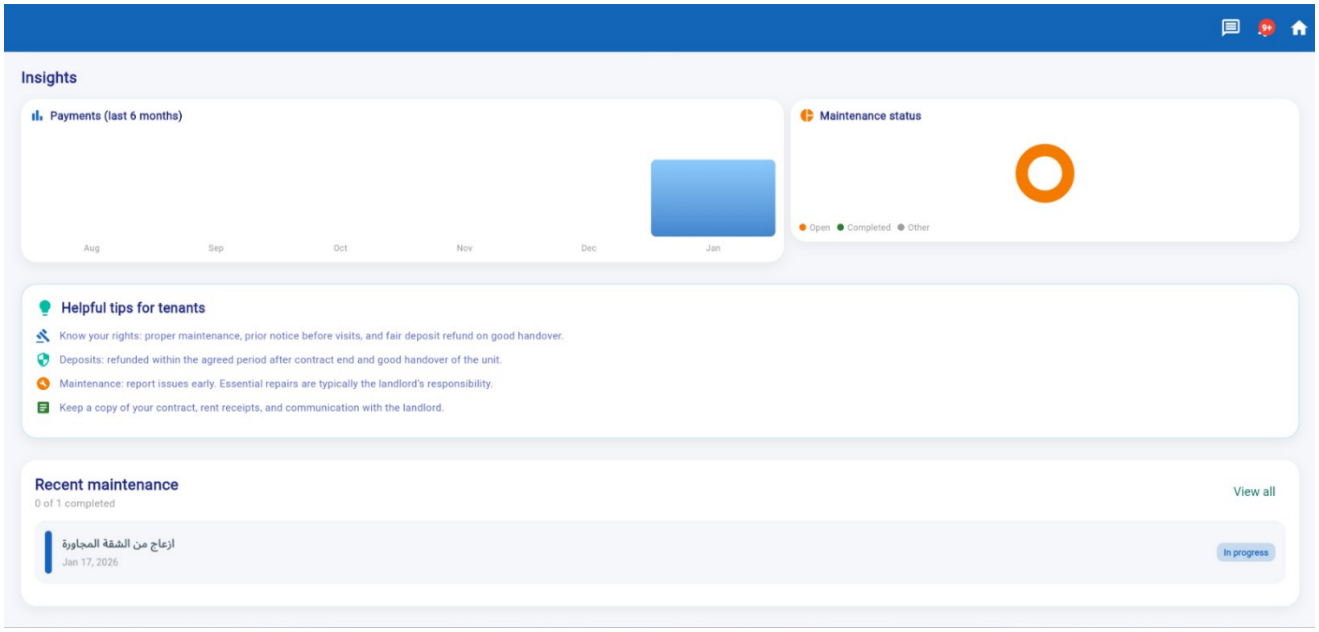





SHAQATI
Buy
Rent
Services
Contact Us
News & Insights
Log in
Sign up
Explore on Map
FOR RENT
$45
zz
1 Beds
1 Baths
0 m²
Chat
FOR RENT
$400
SHAQTIiiiiii
1 Beds
1 Baths
200 m²
Chat
FOR RENT
$0
Unit - Apt 1
1 Beds
1 Baths
0 m²
Chat
FOR RENT
$0
Unit - Apt 2
1 Beds
1 Baths
0 m²
Chat
AI Assistant


Good afternoon, nezar hanani
No pending payments
Overview
1
Active contracts
0
Pending payments


$5.00K
Expenses this month
$1.50M
Total deposits
1 deposits
1
Open maintenance
0 of 1 completed
0
Contract expiry alerts


Insights
Payments (last 6 months)
Aug
Sep
Oct
Nov
Dec
Jan
Maintenance status
Open
Completed
Other
Helpful tips for tenants
Know your rights: proper maintenance, prior notice before visits, and fair deposit refund on good handover.
Deposits: refunded within the agreed period after contract end and good handover of the unit.
Maintenance: report issues early. Essential repairs are typically the landlord's responsibility.
Keep a copy of your contract, rent receipts, and communication with the landlord.
Recent maintenance
0 of 1 completed
View all
ازعاج من الشقة المجاورة
Jan 17, 2026
In progress

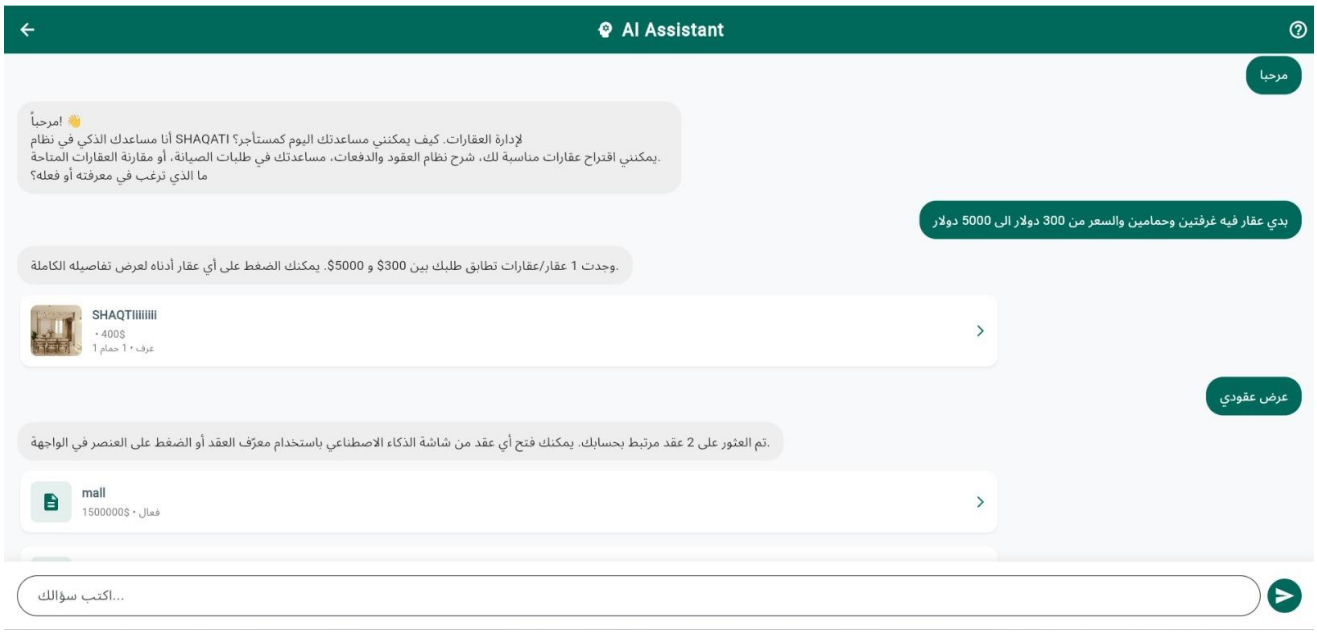
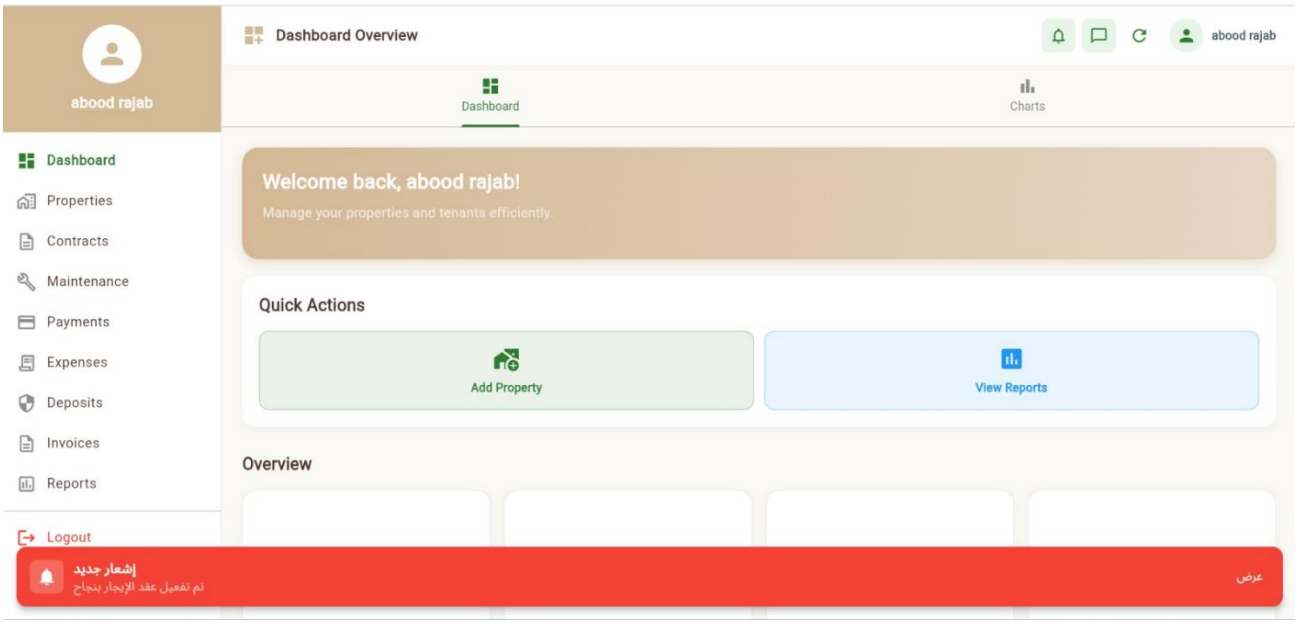
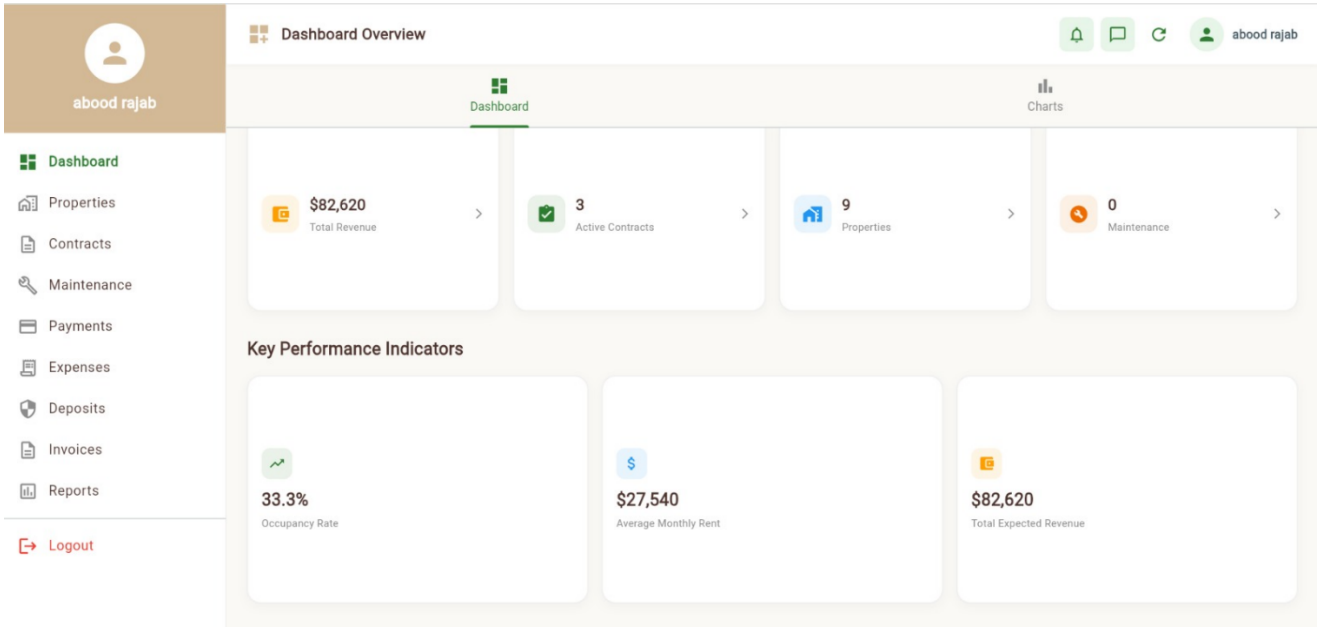


Figure44: Tenant web pages

- Landlord


abood rajab
Dashboard Overview
abood rajab
Dashboard
Charts
Dashboard
Properties
Contracts
Maintenance
Payments
Expenses
Deposits
Invoices
Reports
Logout
Welcome back, abood rajab!
Manage your properties and tenants efficiently.
Quick Actions
Add Property
View Reports
Overview
إشعار جديد
تم تفعيل عقد الإيجار بنجاح
عرض


abood rajab
Dashboard Overview
abood rajab
Dashboard
Charts
Dashboard
Properties
Contracts
Maintenance
Payments
Expenses
Deposits
Invoices
Reports
Logout
$82,620
Total Revenue
3
Active Contracts
9
Properties
0
Maintenance
Key Performance Indicators
33.3%
Occupancy Rate
$27,540
Average Monthly Rent
$82,620
Total Expected Revenue

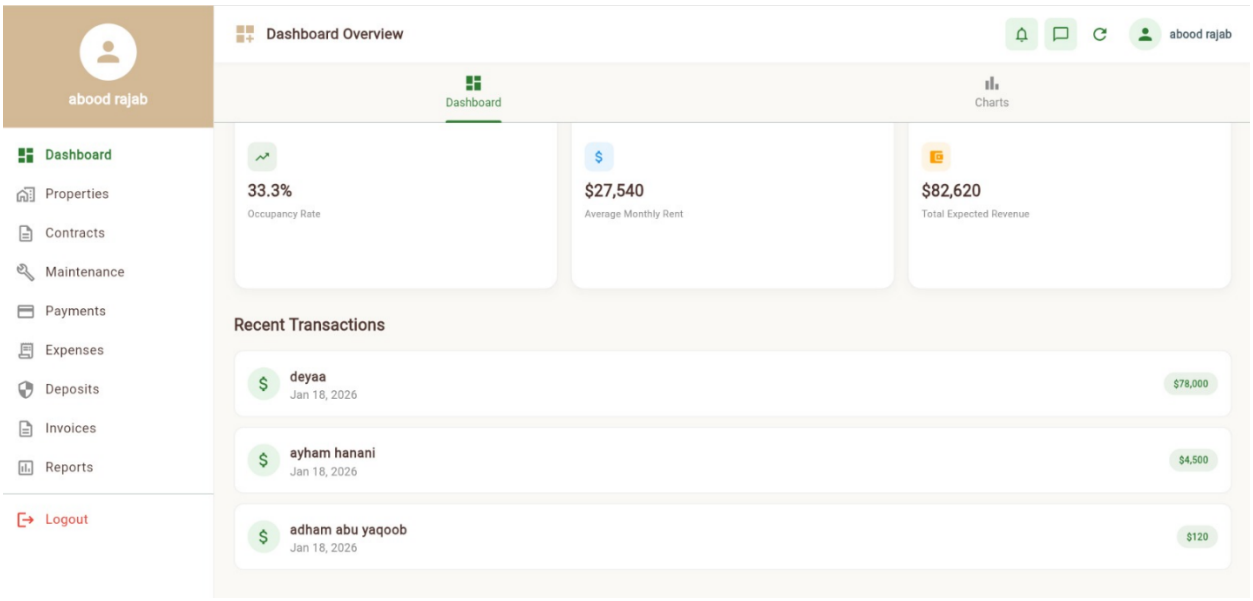
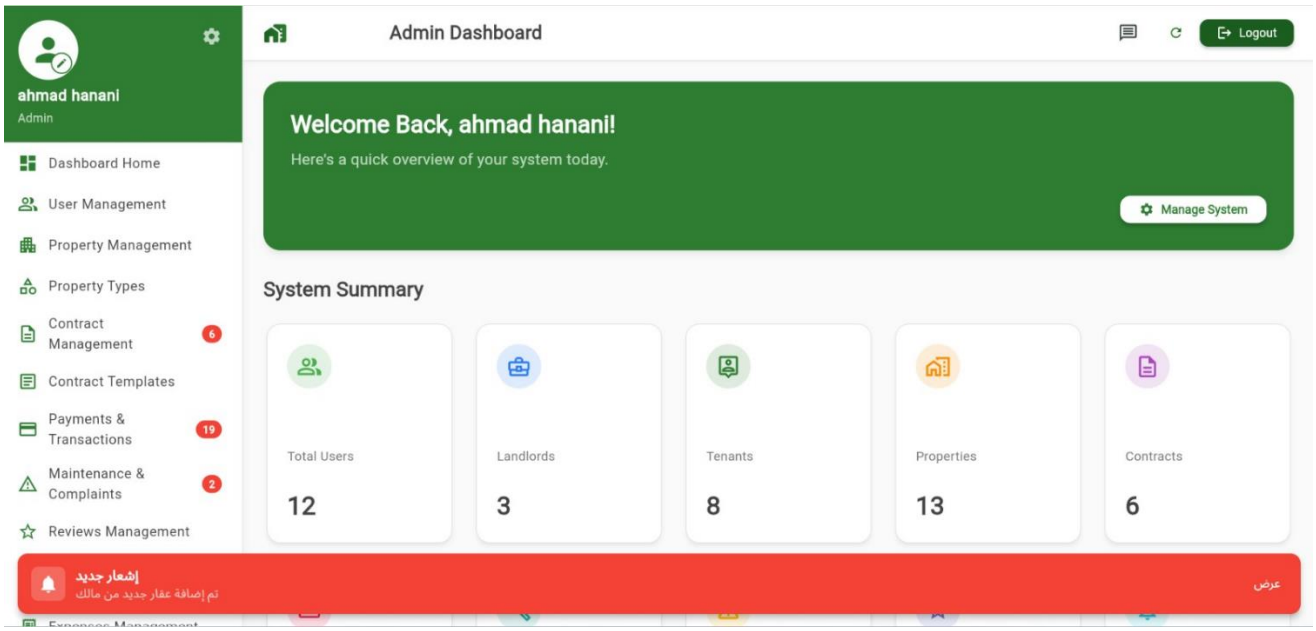
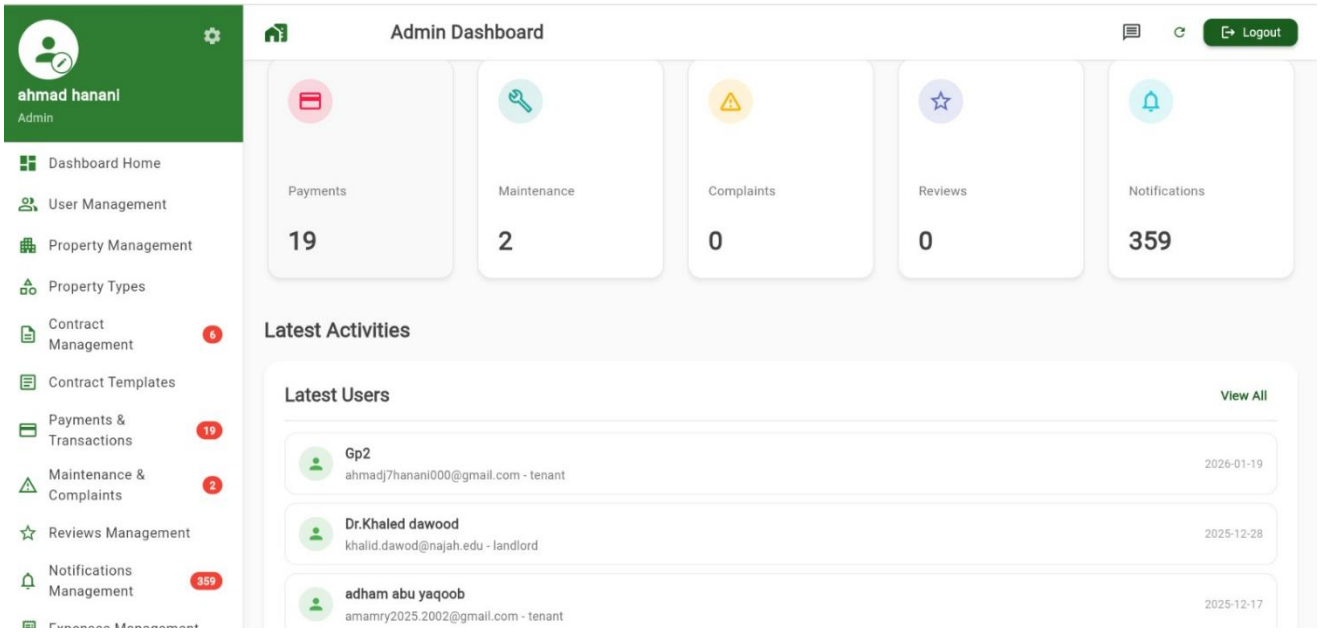


Figure45: Landlord web pages

- Admin


ahmad hanani
Admin
Admin Dashboard
Logout
Dashboard Home
User Management
Property Management
Property Types
Contract Management
6
Contract Templates
Payments & Transactions
19
Maintenance & Complaints
2
Reviews Management
Welcome Back, ahmad hanani!
Here's a quick overview of your system today.
Manage System
System Summary
Total Users
12
Landlords
3
Tenants
8
Properties
13
Contracts
6
إشعار جديد
تم إضافة عقار جديد من مالك
عرض


ahmad hanani
Admin
Admin Dashboard
Logout
Dashboard Home
User Management
Property Management
Property Types
Contract Management
6
Contract Templates
Payments & Transactions
19
Maintenance & Complaints
2
Reviews Management
Notifications Management
359
Payments
19
Maintenance
2
Complaints
0
Reviews
0
Notifications
359
Latest Activities
Latest Users
View All
Gp2
ahmadj7hanani000@gmail.com - tenant
2026-01-19
Dr.Khaled dawood
khalid.dawod@najah.edu - landlord
2025-12-28
adham abu yaqoob
amamry2025.2002@gmail.com - tenant
2025-12-17

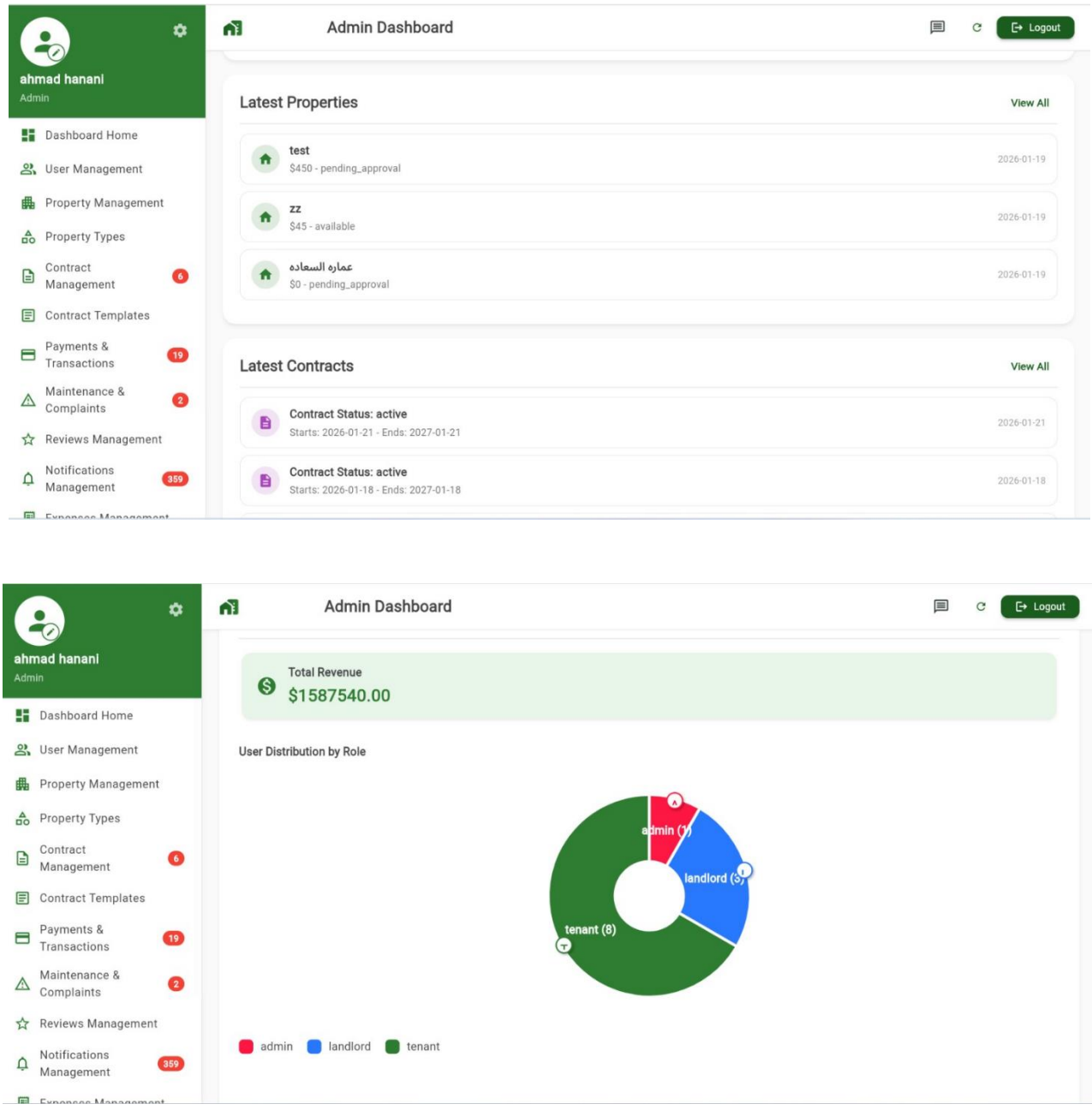


Figure46: Admin web pages

4.1.12 Database and API Server

In order to create a high-performance backend server and implement all of the RESTful APIs, we utilized the Node.js framework in our application. Numerous built- in Node.js functions and libraries helped us implement and build the necessary functionality and methodology.

5 Chapter 5

5.1 Result and Analysis

In developing the Property Management System, the implementation resulted in a cross-platform system that supports property management operations. Two databases were used: Firebase Cloud Messaging (FCM) for real-time push notifications about maintenance requests, contract updates, and payment reminders, and MongoDB as the primary data storage for users, properties, maintenance requests, contracts, invoices, and roles.

Using Flutter enabled building mobile and web interfaces from a single codebase, ensuring consistent UI and faster development. Node.js served as the backend, managing requests and interactions with MongoDB and Firebase.

For hosting, platforms like Render were used to deploy the server online, keeping it accessible. Media management was handled using Cloudinary, where property images and 3D model assets were stored externally, with URLs saved in the database for scalability and improved load times.

The result is a stable, scalable, and user-friendly system that provides:

- Real-time notifications for maintenance requests, contracts, and payments.
- Role-based access and dashboards (tenant, landlord, admin).
- Admin capabilities to manage properties, users, maintenance requests, contracts, and invoices.
- A clean interface for property browsing, maintenance tracking, contract management, and financial oversight.

The system met its functional goals and demonstrated solid integration of multiple components, including notification services, role-based access control, and multi-platform deployment.

6 Chapter 6

6.1 Discussion

Building the Property Management System presented numerous **technical and logistical challenges** but also served as a major learning opportunity.

6.1.1 Learning Outcomes

We were introduced to several new technologies:

- **Dart & Flutter** for UI and front-end development across platforms.
- **Node.js** for scalable backend APIs.
- **MongoDB** for NoSQL data storage, and **Firebase** for real-time functionality.
- Cloud platforms like **Render** and **Cloudinary**, used for backend deployment and media hosting respectively.

The experience strengthened our skills in **software architecture, RESTful API design, and real-time communication.**

6.1.2 Key Challenges

- **New technologies**: Dart, Flutter, and Firebase were completely new to us, which extended the learning curve.
- **Dual database management**: Maintaining data consistency between MongoDB and Firebase required careful design and synchronization.
- **Performance issues with Render**: Occasionally we faced slow response times or server downtimes, especially during deployment.
- **Cloudinary integration**: Secure media upload and linking presented initial difficulties until proper configuration was established.
- **Hardware limitations**: Running multiple emulators, development tools, and services caused heavy system load on our laptops, affecting productivity.

Despite these challenges, the team delivered a functioning system that aligns with our original vision and offers potential for real-world use in Palestine's property management sector.

7 Chapter 7

7.1 Conclusion and Recommendation

7.1.1 Summary

We developed a cross-platform Property Management System that provides a real-time experience for managing properties and rental operations. Tenants can browse properties, submit maintenance requests, view contracts, and track payments. Landlords can list properties, manage maintenance requests, oversee contracts, and monitor their portfolio. The admin dashboard lets administrators monitor system usage, manage users and properties, and view statistics through a clean interface. With a focus on usability, real-time features, and centralized management, the system connects tenants, landlords, and administrators, improving property management operations and service delivery.

7.1.2 Things We Learned

Throughout the project, we gained technical and practical skills, including:

- Mobile app development using the Flutter framework and Dart language.
- Building REST APIs with Node.js and Express.js using JavaScript.
- Testing and validating endpoints with Postman.
- Integrating real-time push notifications using Firebase Cloud Messaging (FCM).
- Deploying the back-end with Render, handling server-side operations and database connections.
- Uploading and managing media using Cloudinary, and storing their URLs in MongoDB.
- Working with two different databases (Firebase for notifications and MongoDB for core storage).
- Using Git and GitHub effectively to manage and collaborate on code.
- Debugging and optimizing performance under limited internet connections and hardware constraints.

7.1.3 Recommendations

Mobile app development has become an essential part of software engineering and should be given more attention in academic curriculums. We recommend offering a dedicated mobile development course that focuses on frameworks like Flutter and Firebase, enabling students to build real-world applications with modern tools and deploy them confidently.

7.1.4 Future Work

The Property Management System can be expanded and enhanced in the following:

- Enhancing notification systems with support for email notifications, SMS alerts.
- Integrating advanced property search with AI-powered recommendations, virtual tours, and augmented reality features.
- Introducing automated payment processing with payment gateways, recurring billing, and financial analytics.
- Creating comprehensive analytics dashboards with predictive insights, revenue forecasting, and maintenance trend analysis.
- Implementing advanced security features such as two-factor authentication and encrypted document storage.

8 Chapter 8

8.1 References

8.1.1 [1] GitHub, “Source Code Management and Version Control,” GitHub, 2024. [Online]. Available: https://github.com

8.1.2 [2] Flutter Dev Team, “Flutter Documentation,” Flutter, 2024. [Online]. Available: https://docs.flutter.dev

8.1.3 [6] Google Maps Platform, “Maps, Routes & Places APIs,” Google Developers, 2024. [Online]. Available: https://developers.google.com/maps

8.1.4 [7] Cloudinary, “Media Asset Management and Image Hosting,” Cloudinary, 2024. [Online]. Available: https://cloudinary.com

8.1.5 [8] Render, “Cloud Hosting and Deployment Platform,” Render, 2024. [Online]. Available: https://render.com

8.1.6 [9] Firebase, “Realtime Database and Notifications,” Firebase, 2024. [Online]. Available: https://firebase.google.com

8.1.7 [10] Node.js, “Node.js Runtime and NPM Libraries,” Node.js, 2024. [Online]. Available: https://nodejs.org

8.1.8 [11] Socket.IO, “Real-time Communication Library for Node.js,” Socket.IO, 2024. [Online]. Available: https://socket.io