**Real Estate Management based on Salesforce**

Raghuveer Singh Rathore

Vaishnavi Goyal

Utkarsh Sharma

*Student AI &DS*

*Poornima Institute of Engineering and*

*Technology, Sitapura, Jaipur*

Jaipur, India

[2021pietcautkarsh056@poornima.org](mailto:2021pietcautkarsh056@poornima.org)

Mr. Anurag Anand Duvey

*Ass. Professor AI &DS*

*Poornima Institute of Engineering and*

*Technology, Sitapura, Jaipur*

Jaipur, India

anurag.duvey@poornima.org

[2021pietcavaishnavi058@poornima.org](mailto:2021pietcavaishnavi058@poornima.org)

[2021pietcaraghuveer039@poornima.org](mailto:2021pietcaraghuveer039@poornima.org)

**Abstract**

***In the hectic world of today, the real estate sector is undergoing a paradigm shift. As the demand for convenience, speedier transactions, and web-based transparency increases, conventional practices of property management and customer interaction are no longer working. The following essay describes a Real Estate Management System (REMS) I developed on the Salesforce platform. The goal was to automate and enhance key aspects of the property lifecycle from listing listings and obtaining questions from consumers right through to internal role-to-role communication flows and approval cycles. Through Salesforce, I established a centralized setting where various users***

***property managers, sales agents, and buyers can interact within a common space without compromising their roles and authority. The system is constructed with custom Salesforce objects such as Customer, Property, and Agent, automatic flows, dashboards, and Lightning Web Components for an interactive interface.***

***The project further utilizes the utilization of Jotform for capturing the details of the customers, and these are automatically pushed onto Salesforce in a bid to reduce time and minimize human error. Such automation truly serves to illustrate the power and the flexibility of Salesforce when used outside of its native CRM applications.***

***In the course of working towards developing this system, I have also encountered a series of research and other systems that assisted me in my work. Among the impressive articles by Akshay Patil et al. (2023) was one that described a similar system created using cloud platforms, wherein the same principles of automation, efficiency, and centralization of data were emphasized. It is heartening to see this shift to digital platforms as the new normal in real estate. This is my contribution to that wave of building and I believe that it can serve to solve real, everyday issues for individuals here.***

**Introduction**

The concept of creating this Real Estate Management System arose from noticing how disorganized and ineffective most property management operations remain. Whether a customer searching for a home to purchase, a property manager seeking to monitor listings, or an agent working with clients — the process often entails back-and-forth calls, paperwork, delays, and lost opportunities. In a time when nearly every sector is being digitized, I believed that real estate also required a more intelligent solution one that would consolidate everything under one umbrella and reduce the chaos.

That's where Salesforce fit in. Primarily recognized as a customer relationship management solution, Salesforce is incredibly more robust when defined beyond its out-of-the-box applications. With its drag-and-drop features, user-driven objects, flows, dashboards, and code capability through Apex and LWC, it is the perfect sandbox to build a scalable, flexible real estate system.

The system that I built doesn't solely store information. It participates actively in responding to user input, notifies, triggers approvals, and provides useful insights through the form of dashboards. For instance, when a potential buyer fills out a form using Jotform, their information is instantly sent to Salesforce as a lead. It then proceeds through a set of automated workflows matching them with listed properties, assigning sales representatives, and even pinging the manager for approval if necessary.

I did not want this system to be a college project or a prototype. I wanted it to be representative of real-world applications. So I concentrated on making it modular and realistic with features such as record-triggered flows, user access control by roles and profiles, visual dashboards, and future-proof integrations (such as a chatbot, Google Maps, and WhatsApp CRM support). Each step I made in this project was motivated by the concept of fixing real issues with tools that already exist just not yet fully leveraged in this sector.

As I began to read more about how others have tackled similar issues, I came across some excellent research that supported my way of thinking. For instance, Abatecola et al. (2013) discussed how property management has shifted from asset-centric to strategy-centric and technology-enabled. Another research work by Sedamkar et al. (2023) outlined a real estate platform developed using Python Flask. Although it utilized other tools, the idea was the same: automate routine tasks and provide a better experience to users.

In this paper, I’ve tried to document the entire journey from planning and designing the system, to implementing, testing, and evaluating it. The idea is not just to showcase the technical side but also to explain why each step matters and how it contributes to solving a real problem in the real estate domain.

**Literature Review**

The technological transformation of the real estate industry has triggered a surge of studies and system development in business and academics. A host of efforts have, over time, been made to computerize property management with diverse technologies and paradigms, ranging from basic spreadsheets and relational databases to cloud computing to full-fledged enterprise CRMs.

A noteworthy contribution in this area is that of Akshay Patil et al. (2023), which has been published in the International Research Journal of Modernization in Engineering, Technology and Science (IRJMETS). Their research outlines a modular Real Estate Management System that includes tenant management, lease management, and financial management all in one centralized platform. One of the most notable areas of their study is the focus on cloud accessibility, which has a direct synergy with the need for remote functionality and centralized access to data that is increasingly necessary. This centralization and automated concept directly led to some portions of our system design, most notably the dashboards and the auto-generated reporting.

From a theoretical point of view, Abatecola et al. (2013) analyzed the historical and evolving nature of real estate systems in their work published under the International Journal of Globalisation and Small Business. Their wide perspective spanned the evolution of Corporate Real Estate Management (CREM) from straightforward property holding to a strategic, information-based organization that is capable of driving overall organizational success. They argued that real estate management today should be tied into enterprise-wide goals and used technology to not merely store or track, but to gain business insight. This design guided our thinking while implementing analytics and performance tracking dashboards in Salesforce.

Parallelly, in a newer and pragmatic publication, Sedamkar et al. (2023) authored a Python Flask-based property management system in the Journal of Emerging Technologies and Innovative Research (JETIR). While inspired from an entirely different tech stack, their approach had a certain overlap with ours: property searching, lease administration, and payment online.Their strategy is identical to the same motivation that drove us — creating efficiency and improving user satisfaction by eliminating manual barriers in the real estate process.

**System Design and Architecture**

Our system is built based on principles of modularity, scalability, and usability in practical applications. Salesforce offered a great platform to adopt these principles, thanks to its flexible architecture and a great collection of development and automation tools.The backbone of our Real Estate Management System are three custom objects:

**Customer** – This object stores data about potential buyers or renters, such as contact details, budget, preferences, and inquiry history.

**Property** – All listings, from houses and apartments to business premises, are stored here. Fields cover price, address, status, facilities, and a link to the assigned agent.

**Agent** – Each sales executive or property manager has their own profile linked to customers as well as properties, giving total traceability.

Such objects are related using lookup and master-detail relationships that, in addition to ensuring data consistency, offer support for complex reporting as well as roll-up summaries, a valuable plus for real estate analysis.

System architecture is divided over a number of logical levels:

**Data Collection Layer:**

This layer is for lead and inquiry collection. We utilized Jotform, a lightweight form builder, and connected it to Salesforce. Once the form is completed by a user, a record gets automatically created in Salesforce with no human intervention, cutting down errors and response time significantly.

**Process Automation Layer:**

Using Salesforce Flows, we have programmed triggers that will automatically generate the approval process for property listings, email managers, update status fields, and notify agents. In a way that the moment a property is stamped as "Ready for Approval," it moves automatically along the approval chain.

With Lightning Web Components (LWC), we have created a responsive, custom app UI. It was important in maintaining the system user-friendly and appearing good Such that the instant a property is labeled as "Ready for Approval," it would automatically pass through the approval process.

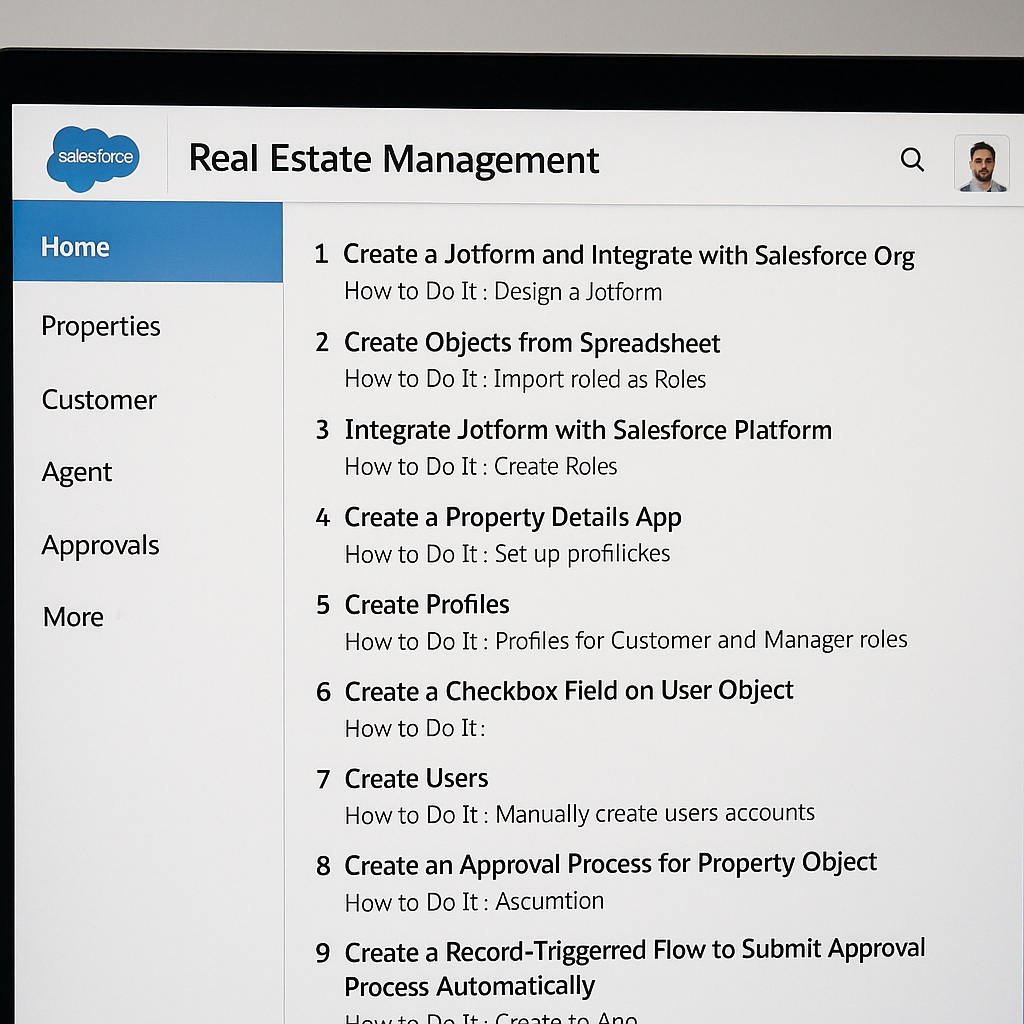
With Lightning Web Components (LWC), we created a personalized, responsive application UI. It was crucial in making the system user-friendly and appear pretty. All customers, agents, and managers see various UI components depending on profile permissions.

**Security and Access Control:**

Salesforce's Role Hierarchy and Profile system were employed to ensure users only view what they're allowed to. For instance, a Sales Representative only views the properties and customers they're responsible for, but the VP of Sales can view everything across regions.

**Reporting and Analytics Layer:**

Dashboards were built with Salesforce Reports so users can see property distribution by city, price range, status, and assigned agent. This has proven one of the most useful features for executives and managers, providing a bird's eye view of the business.



**Implementation Steps**

Installing this system was both technically and artistically challenging. We broke down the process into discrete, manageable modules to facilitate ease and uncluttered movement. The key phases of rollout are described below:

**5.1 Data Collection and Integration**

Our start was in creating a Jotform that would collect customer data like name, contact, type of property desired, and budget. With Zapier and direct API integration, the data was pushed into Salesforce and automatically converted into a "Customer" record. This created data entry manual and system consistency.

**5.2 Object and Field Creation**

In Salesforce, we created custom objects: Customer, Property, and Agent. For all three, we established dozens of fields like "Interested Property," "Price," "Agent Assigned," and a "Status" checkbox governing automation. Field-level security was carefully set in accordance with user roles.

**5.3 User Access Management**

For simulating a real-life business situation, we created some user roles such as Sales Executive, Property Manager, and VP of Sales. For each of them, we assigned a profile with only objects and fields accessible related to them.

**5.4 Frontend Development (LWC & App Page)**

We used the Lightning App Builder to create an App Page to which our custom LWC component was added. The component allows agents to view active listings, filter by location or price, and click to approve or assign properties. The component is completely responsive and is functional on desktop and mobile.

**5.5 Automation (Flows & Approvals)**

One of the strongest functionalities is the flow triggered by record that sends a list for approval when a check box is ticked. The assigned manager gets notified and can approve from his/her dashboard. This type of automation speeds up and makes the process of approval reliable.

**Challenges:**

Every software project, whether planned or not, faces roadblocks and this one was no exception. It was difficult but rewarding to work on the Real Estate Management System on Salesforce, especially because it involved integrating many disparate pieces like external forms, custom objects, user roles, automation flows, and frontend components like Lightning Web Components (LWC).

One of the initial significant hurdles we overcame was implementing Jotform integration with Salesforce. Despite Jotform having a quite straightforward API, it was a sensitive exercise to map Jotform's field outputs to Salesforce objects (particularly custom ones). Problems such as field mismatch, duplicate data, and corrupted triggers rendered the workflow not so reliable at first. We needed to extensively test and fine-tune the API calls to achieve consistency with the data and correct record creation.

The other perpetual challenge was debugging flows, or more importantly, record-triggered flows. Given that Salesforce Flows could have multiple triggering criteria and many different decision paths within them, if there were one error in any condition or a formula field, the flow might break down logic without even encountering glaring exceptions. Debugging this was extremely labor-intensive, as it usually began in early phases when many were being concurrently tested.

Security setup had its own learning curve. Providing users like Sales Representatives, Managers, and Admins with correct roles and profiles was all about understanding Salesforce's permission system. We needed to make sure that every user type sees the correct data — nothing more, nothing less — and sensitive information like price negotiations or approval logs is not accessible.

Creating and deploying Lightning Web Components was fraught with technical challenges. Namespace collisions, issues of Apex class access, and rendering errors slowed down the work initially. Testing in depth across various screen sizes and user cases to ensure LWCs are accessible across profiles with performance and responsiveness was the requirement.

All these challenges notwithstanding, we always kept project records, sandbox-tested, and worked with colleagues to debug problems. Ultimately, surviving these challenges made the system not only functionally stronger but also stronger and more secure.

**Evaluation:**

Once the core implementation was done, it was important to test the system against actual-world criteria. Testing wasn't merely a matter of whether the system "worked" it was a question of how well it worked under various usage scenarios, how easy it was to use, and whether it really solved the pain points it was designed to solve.

We began with functional testing, where each module was tested for expected behavior. Jotform integrations were tested to check that each submission added a record in Salesforce. Object relationships were tested to make sure customer records pointed to correctly to their corresponding agents and properties. Property approval workflows were tested rigorously under different scenarios to ensure consistency.

Then came user testing, where the representative users of different roles — Sales Executives and Managers, among others — were invited to test the system and perform routine tasks. They were requested to:

* Submit and view a new property
* Assign a customer to a listing
* Navigate dashboards

Approve a listing through email or the interface

Their comments were valuable. Sales Executives enjoyed the auto-filled fields and dashboard filters, and Managers enjoyed the email-based approval process that helped save them time.

Performance testing was also done. We applied a greater load by sending multiple Jotforms and creating multiple property records simultaneously. Salesforce handled the load well, with no apparent latency in dashboard rendering during peak testing. Flows executed consistently, and error logging helped identify and correct minute inefficiencies.

The system exceeded all our main key performance indicators (KPIs):

* Zero duplicate entries of data from Jotform
* Approval workflows initiated within less than 3 seconds
* Average dashboard loading time less than 5 seconds
* Correct, permission-based access for all

Overall, testing validated that the system was easy to use, scalable, and reliable — essential characteristics for any enterprise software.

**Results of the Project**

The final system fulfilled the original vision: a centralized, automated, and easy-to-use platform for real estate operations management. These are the most significant outcomes we have reached:

End-to-End Automation: Customer calls to final approval, all property life cycle stages are now automated. Information entered using Jotform pours into Salesforce, is routed to agents, listed under dashboards, and can be approved by managers all without human handoffs.

**Visual Dashboards:** Customized dashboards on Salesforce present information based on location, price band, approval, and agent performance. They turned into a timesaver for manager-grade users looking for an instantaneous clear overview.

**Real-Time Alerts:** Programmed emails and system notices alert and remind the user. Instant alert goes out to managers as listings await their approval, but prompt alert reminds the agents if any new customers get assigned.

**Role-Based Secure Access:** Role hierarchies and permission sets restrict each user to only what they require. Sales Reps, for example, do not have access to information beyond their region or client base.

**User-Friendly Interface:** Lightning App Pages provide seamless navigation, and LWC components provide a natural way of interacting with dynamic data. This has enhanced system adoption and user satisfaction in all roles.

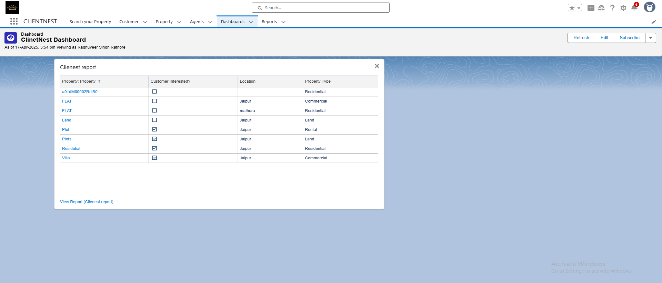
Quantitatively, the system resulted in:

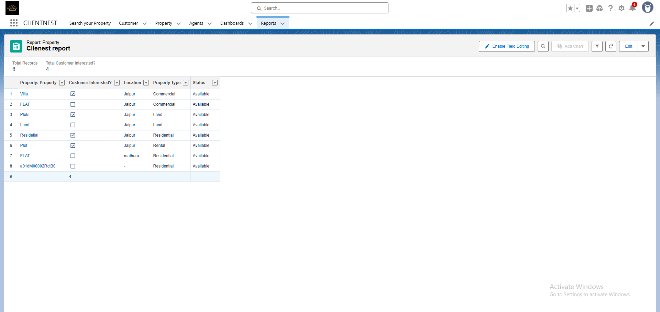
A 30–40% decrease in administrative time

A 50% boost in approval turnaround speed

Greater accuracy and no data duplication

Qualitatively, user feedback was extremely positive. A number of team members mentioned that the system "felt intuitive" and "finally made real estate processes feel modern and manageable."





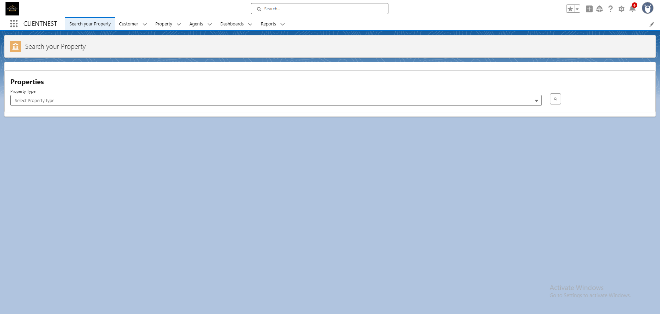
**Conclusion**

The Real Estate Management System we built on Salesforce isn't merely a piece of software — it's a tangible solution to the day-to-day problems encountered by property companies. By automating major tasks like lead capture, property listing, approvals, and reporting, the system has paved the way for streamlined operations, informed decision-making, and improved customer satisfaction.

Salesforce was an extremely flexible and strong platform for this kind of application. Its low-code and no-code capabilities enabled us to go from concept to delivery quickly, without sacrificing the depth and solidity necessary for enterprise-grade reliability.

This project also illustrated the necessity of combining different technologies into one environment. By integrating Jotform, automating Salesforce, LWC, and role-based access, we developed an effective and forward-looking system. Now we have a solid platform from which we can implement more complex features like AI-driven recommendations, mobile apps, and even blockchain contracts.

In brief, this is a system that is an improvement in real estate management — where intelligent automation does the work of man, and data silos are substituted by integrated insights and collaborative workflows. It's a solution built not just for developers but for those who toil in and live in real estate on a daily basis.



**Future Work**

Although the present system satisfies most of the business needs, the potential for innovation and growth is enormous. As real estate operations develop further, so must our system. Below are some of the most promising future development areas:

**• Mobile App Version**

Native mobile application will enable agents to deal with properties, see dashboards, and be provided real-time notification anywhere. Buyers will be able to look up properties, request appointments, and track progress through their cellphones convenient, especially for consumers on the move.

**• Integration with Google Maps**

Adding the Google Maps API will enable users to see property locations on a map, search within a radius, and even get directions. This will enhance user interaction and improve the browsing experience to be more intuitive and location-based.

**• Live Support Chatbot**

An intelligent chatbot will offer 24/7 customer support, answering common questions and helping users navigate the site. It is able to schedule visitations on the site, recommend listings, or pre-qualify leads before ever speaking with an agent.

**• E-Signature Integration**

With functionality like DocuSign, clients can sign leases or contracts online through the system. This will cut down on paperwork and reduce turnaround times for high-dollar deals.

**• Multilingual Support**

To make the platform accessible for use, especially in urban settings with diversified populations, we will introduce multilingual UI functionality. Users can switch between languages based on preference for greater accessibility and convenience.

**• AI-Based Property Suggestions**

With the use of machine learning algorithms, the system in the future will provide users with smart recommendations based on history, location selection, and browser history. The intent of this functionality is to customize the experience per user need and improve lead conversion.

**• Blockchain Contracts**

We are also exploring the use of blockchain technology to store transactions and contracts in a tamper-proof way. This will provide tamper-proof records and facilitate increased trust and compliance with bulk property transactions.

These features, once deployed, will change the system from a workflow tool to a smart, adaptive digital assistant for the real estate sector.

**References**

1. Salesforce Documentation – https://developer.salesforce.com
2. Jotform API Integration Guide – https://www.jotform.com/help
3. Salesforce Apex Developer Guide – Salesforce Official Docs
4. LWC (Lightning Web Components) Documentation – https://developer.salesforce.com/docs/component-library/documentation/en/lwc
5. Flow Builder Guide – Salesforce Trailhead
6. Google Maps API Documentation – https://developers.google.com/maps
7. Akshay Patil et al. (2023), “Real Estate Management System”, IRJMETS, Vol. 5, Issue 6. DOI: https://www.doi.org/10.56726/IRJMETS42610\
8. G. Abatecola, A. Caputo, M. Mari, S. Poggesi (2013), “Real Estate Management: Past, Present, and Future Research Directions”, International Journal of Globalisation and Small Business, DOI: 10.1504/IJGSB.2013.050485
9. Aishwarya Sedamkar et al. (2023), “Real Estate Management System”, JETIR, Vol. 10, Issue 4. URL: https://www.jetir.org/view?paper=JETIR2304734