WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

AI LAND VERIFICATION AND RECOMMENDATION PLATFORM

Janani J M[1], Kanishka V[2], Netra S D[3], Mrs. Dr. Deepa M [4]

Associate Professor, Department of Information Technology,
Sri Shakthi Institute of Engineering and Technology, Coimbatore-641062, Tamil Nadu, India.
Email ID: kanishkavenugopal123@gmail.com

ABSTRACT :

Buying land is a high-value decision, yet buyers often find it difficult to confirm that a listing is genuine, that the seller holds the title, and that the plot suits their budget, size, and purpose. Manual document checking is slow and error-prone, and listing portals usually leave the verification burden on the buyer. This paper presents the AI Land Verification and Recommendation Platform (LandVision AI), a web-based application in which owners list land and submit supporting legal documents, an AI model extracts the visible details from those documents, administrators review them against a structured checklist, and buyers receive ranked land recommendations. The proposed system aims to combine role-based authentication, land listing management, secure document upload, AI-based field extraction, mismatch warnings, administrator approval with an audit trail, natural-language search, and transparent match scoring. The application uses a React.js frontend, a Node.js and Express.js backend, Firebase Firestore for data storage, Cloudinary for images and documents, and the Gemini API for document analysis and request parsing. AI extraction is treated as preliminary assistance and never as proof of ownership; only an administrator decision can mark a listing as verified. The system is designed with consideration for usability, privacy, security, and reliable handling of submitted documents. Functional and integration testing can be used to evaluate the individual modules and their interaction. The proposed system provides a foundation for combining document-assisted verification and preference-based recommendation in a single application, with further development required to evaluate extraction accuracy on real-world documents.

Keywords: Land Verification, Land Recommendation, Document Analysis, Gemini AI, Role-Based Access Control, Match Scoring, React.js, Firebase Firestore.

1. INTRODUCTION

Land is one of the most valuable assets that individuals and families purchase, and the decision involves legal, financial, and practical considerations. Before buying, a purchaser is expected to confirm that the seller is the rightful owner, that the survey number and extent in the sale deed match the plot being offered, and that no encumbrance or dispute is attached to it. In practice, these checks depend on reading registered documents, comparing details across several records, and consulting officials or advocates, which is time-consuming and difficult for first-time buyers.

The growth of web platforms, cloud databases, and Artificial Intelligence (AI) has created new possibilities for property services. Document-understanding models can read scanned records and return structured fields, cloud databases can store listings and review decisions, and recommendation logic can rank listings against a buyer's requirements. Combining these technologies may help create a centralized platform that supports buyers while keeping the final decision about authenticity with a human reviewer.

1.1 Existing challenges in land search and verification

Existing approaches include classified advertisements, property listing portals, brokers, and direct enquiries. Although these channels make listings easy to find, they usually do not show whether the supporting documents have been examined, and a listing described as verified may not state what was actually checked. Buyers therefore often need to search on one platform, arrange document checks separately, and compare plots manually.

Another challenge is the dependence on expert knowledge. Owner names may be spelt differently across records, survey numbers may be written in several formats, and land extent may be stated in cents, acres, or square feet. Fraudulent or duplicate listings, outdated records, and unreadable scans can further reduce trust. Automatic text extraction is also imperfect, so any AI-assisted tool can only provide preliminary evidence and cannot, by itself, establish ownership or legal validity.

1.2 Role of AI and rule-based recommendation

Document-understanding AI can read the visible text of a land document and return structured fields such as document type, owner names, survey numbers, extent, location, document numbers, and registration date. These extracted fields can be compared with the details declared by the owner, and differences can be highlighted for a reviewer. This reduces the time spent on reading but does not replace the reviewer's judgement.

Recommendation can be handled by transparent, rule-based scoring. Hard requirements such as location, purpose, budget, and area range are applied as filters, and the remaining listings are scored for location fit, affordability, and area suitability. Large language models can additionally convert a buyer's free-text request into structured search filters. However, AI outputs may contain errors, and results should be presented as assistance together with the reasons behind each recommendation.

1.3 Problem statement and objectives

Many existing platforms keep listing, document checking, and recommendation separate. This makes it difficult for buyers to find land that is both suitable and trustworthy. There is therefore a need for an integrated application that provides convenient access to verified listings while considering accuracy, transparency, privacy, and ease of use.

The main objective of this work is to propose a web-based AI Land Verification and Recommendation Platform that combines owner listing, AI-assisted document extraction, administrator review, and preference-based recommendation in one platform. The system aims to increase trust in listings, explain why a plot is recommended, and reduce the effort needed to shortlist suitable land.

The specific objectives are:

i. To provide role-based registration and login for buyers, owners, and administrators.
ii. To allow owners to create and manage land listings with details and images.
iii. To accept land documents in secure storage and extract their visible fields using AI.
iv. To compare declared details with extracted details and highlight potential mismatches.
v. To enable administrators to approve or reject documents through a mandatory review checklist and record an audit trail.
vi. To recommend and rank listings using location, purpose, budget, and area preferences, with natural-language search support.
vii. To support buyer favourites and controlled access to owner contact details.
viii. To evaluate the application through functional, integration, and usability testing.

2. MATERIALS AND METHODS

2.1 Development approach

The project follows a structured software development approach consisting of problem identification, requirement analysis, system design, implementation, integration, testing, and evaluation. During problem identification, the main challenges associated with unverified land listings and manual document checking are considered.

Requirement analysis identifies the major functions required by the application, including authentication, land listing, document submission, AI extraction, administrator review, search, and recommendation. System design defines the relationship between the user interface, REST API, verification and recommendation services, and data storage components.

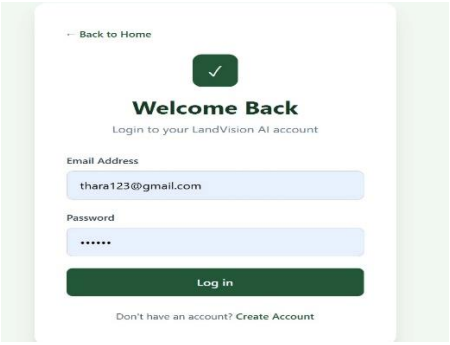
The implementation stage develops the frontend pages and backend modules using React.js, Node.js, and Express.js. Integration connects the individual modules with Firebase Firestore, Cloudinary, and the Gemini API so that they can exchange information correctly. Testing is then used to identify errors, examine the application's behaviour, and determine areas requiring improvement.

2.2 System architecture

The proposed system is organized into multiple layers, including the user interface layer, application logic layer, authentication layer, intelligent service layer, and data storage layer.

The user interface layer provides the screens through which users register, log in, search for land, list plots, upload documents, and view verification status. The application logic layer is an Express.js REST API that processes requests and coordinates communication between the interface and backend services. The authentication layer issues JSON Web Tokens and applies role-based middleware, while the intelligent service layer manages document extraction, natural-language request parsing, and recommendation scoring.

The data storage layer uses Firebase Firestore for users, lands, favourites, verification records, and audit logs, and Cloudinary for land images and authenticated document files. The architecture is intended to keep the modules logically separate so that individual components, such as the verification route or the recommendation service, can be maintained and improved without redesigning the complete application. The login screen of the implemented application is shown below.



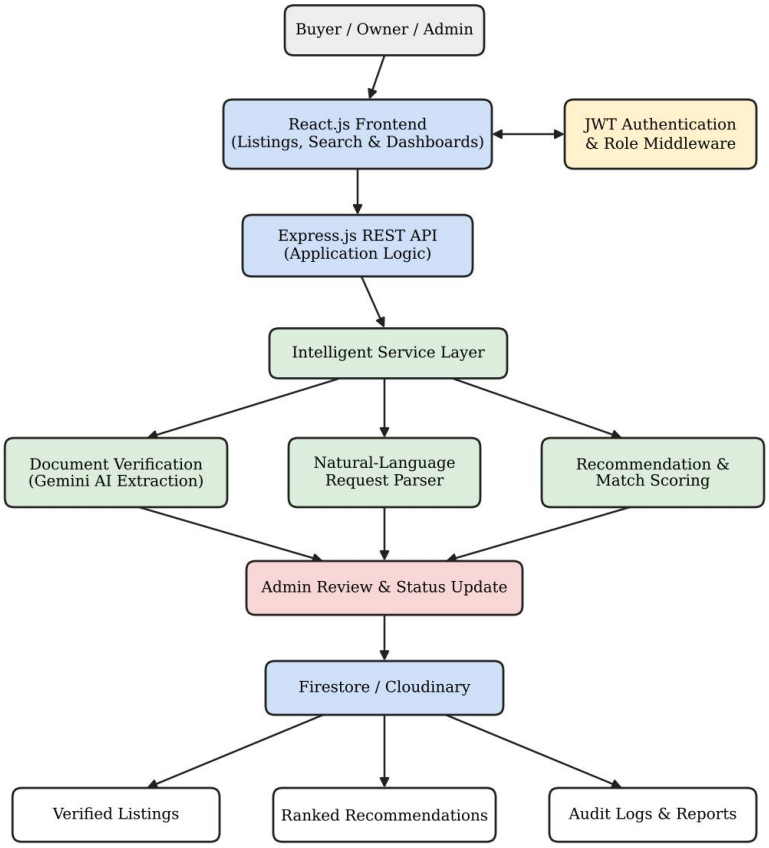


Fig. 1. System Architecture of the Proposed AI Land Verification and Recommendation Platform.

2.3 Technology stack and development environment

The web-based implementation uses React.js with React Router for page navigation, Vite as the build tool, and Tailwind CSS for styling. The interface includes pages for login, registration, land listing, land details, favourites, document submission, and the administrator dashboard. The backend uses Node.js with Express.js to expose REST endpoints and middleware for authentication, role restriction, and file upload.

Firebase Firestore, accessed through the Firebase Admin SDK, stores the application records. Cloudinary stores land images and land documents, and documents are kept as authenticated assets that are accessed through short-lived links. The Gemini API is used for document field extraction and for converting natural-language requests into search filters. Passwords are hashed with bcrypt, and logged-in sessions use JSON Web Tokens [6], [7].

Development and testing can be performed using an appropriate code editor, Node.js, and a supported web browser. The required environment depends on the frontend framework, backend runtime, cloud services, and API credentials, which are supplied to the backend through environment variables.

2.4 Data collection and processing

The system processes information required to provide its features, such as user account details, land listing details, uploaded land images, uploaded land documents, extracted document fields, review decisions, and favourite lists. Documents submitted for verification may include sale deeds, title deeds, patta, encumbrance certificates, parent deeds, property tax receipts, and layout approvals.

Personal information should be collected only when necessary for the application's intended functions. Land documents contain sensitive personal and property information, so they are stored as authenticated assets and are shown only to the owner who uploaded them and to administrators. Owner contact details are returned only to logged-in buyers through a protected endpoint.

The application validates inputs before processing them and handles errors such as missing fields, unsupported file types, files larger than 5 MB, mismatched file content, and unavailable AI services. Uploaded files are accepted only as PDF, JPG, PNG, or WebP, one file at a time, and the file signature is checked in addition to the declared type. The extraction prompt treats document contents as untrusted data so that text inside a document cannot change the extraction task.

2.5 Database design

The database is organized as Firestore collections. A suitable data model contains records for users, lands, favourites, land verifications, and verification audit logs.

User records maintain the name, email, phone number, hashed password, and role (buyer, owner, or administrator). Land records store the title, location, price, area, purpose, description, image, owner identifier, verification status, and active flag. Verification records associate an uploaded document with a land and its owner, and store the declared details, extracted fields, extraction status, review status, review notes, checklist, and reviewer.

Earlier verification records for a land are retained but marked as superseded when a new document is uploaded. Audit-log records store each administrator decision with its time, checklist, and notes. The database should be configured to prevent unauthorized access to documents and review data, and only the information required for the application should be retained.

2.6 Document verification and recommendation logic

The verification process begins when an owner selects one of their own listings and uploads a land document together with declared details such as document type, document number, registration year, registered owner's name, survey numbers, land extent, village, taluk, district, and sub-registrar office. The backend sends the file to the Gemini model, which returns the visible document type, owner names, survey numbers, location, extent, document numbers, registration date, other dates, and a list of uncertain fields [4]. The model is instructed not to guess missing values and not to judge authenticity, legal validity, or ownership.

The administrator dashboard compares the declared details with the extracted fields for document type, owner name, survey numbers, extent, document number, and registration date, and shows potential mismatches and uncertain fields as warnings. Approval is allowed only after the administrator confirms seven checks: the document is readable, the document type and registration details are checked, the owner name, survey numbers, and extent are compared, the location is reviewed, and the official records are checked. A rejection requires a written reason. An approved document sets the listing status to verified, and a rejected document sets it to rejected.

The recommendation module first converts the stored price into rupees and the stored area into cents, where one acre equals 100 cents. It then applies the buyer's location, purpose, budget, and area range as strict filters, so that no listing outside the requested criteria is returned. Listings without an owner, with an inactive or rejected status, or with a missing price or area are excluded. Each remaining listing receives a score out of 100, as summarised in Table 1, together with readable match reasons [8]. The listings are sorted by score, with the lower price preferred when scores are equal, and the top ten are returned.

For natural-language search, the buyer's request is sent to the Gemini model, which returns the location, maximum budget, area range, and purpose as structured values. The backend validates these values before they are used, and the interface keeps a simple rule-based parser as a fallback when the AI service is unavailable. Automated extraction and recommendation should complement, rather than replace, the buyer's own due diligence and professional legal advice, and the application should communicate these limitations to the user.

Scoring component	Maximum points	Basis of calculation
Location fit	35	35 for an exact location match; 30 when the stored location contains the searched place
Land purpose	25	Awarded when the listing purpose equals the requested purpose
Affordability	25	25 × (1 − price ÷ maximum budget); lower-priced listings score higher
Area suitability	15	15 × (1 − 0.5 × distance from the middle of the range ÷ half range)
Total	100	Sum of the components, rounded and limited to the range 0–100

Table 1. Match-score components used by the recommendation module.

2.7 Verification and recommendation workflow

The workflow begins when an owner registers, logs in, and creates a land listing with its details and an image. The owner then uploads a land document for that listing, and the application validates the file type, size, and content before storing it securely. The listing is set to pending review while the document is processed.

The Gemini model extracts the visible fields, which are saved together with the owner's declared details. The administrator opens the review queue, examines the document and the highlighted warnings, completes the checklist, and approves or rejects the document. The decision is stored with the reviewer, time, and notes, and the listing receives a verified or rejected label.

After processing, the application should provide clear feedback about the status of the listing. A pending submission should be distinguished from an approved or rejected one, and an owner whose document was rejected should be able to read the reason and submit a corrected document. Buyers can then search with filters or a natural-language request, view ranked recommendations with verification labels, save favourites, and request the owner's contact details.

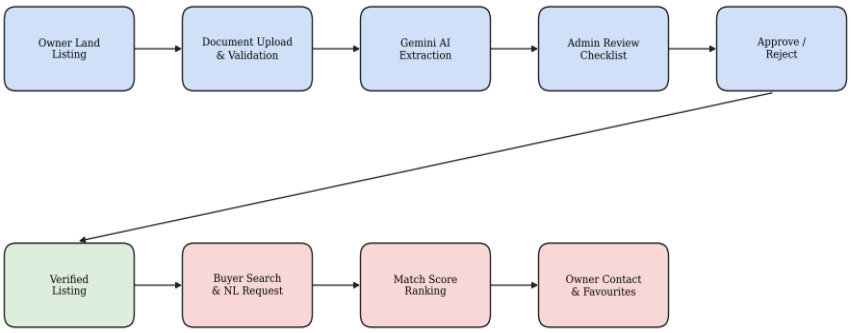


Fig. 2. Workflow of the Proposed AI Land Verification and Recommendation Platform.

2.8 Testing and evaluation

Testing is required to evaluate the correctness and reliability of the proposed system. Unit testing can be used to examine individual functions such as unit conversion and match scoring, while integration testing checks communication between the user interface, REST API, authentication middleware, Firestore, Cloudinary, and the Gemini service.

System testing evaluates the overall application workflow, including registration, login, listing creation, document upload, administrator review, search, recommendation, and favourites. User interface testing examines readability, navigation, responsiveness, and ease of understanding of the displayed results.

The evaluation should also include error conditions such as invalid credentials, unauthorized role access, unsupported or oversized files, expired tokens, unreadable documents, and unavailable AI services. Any quantitative claims about extraction accuracy, recommendation quality, or response time should be reported only when supported by actual measurements.

3. RESULTS AND DISCUSSION

3.1 Proposed system functionality

The proposed application brings several land-related functions into a centralized interface. These include role-based authentication, land listing with image upload, document submission, AI-assisted field extraction, administrator review, natural-language search, and ranked recommendation. Additional functions include buyer favourites, owner contact access, and an audit trail of review decisions.

The integration of these functions aims to reduce the effort required to move between listing sites, document checks, and manual comparison. The actual performance of the application must be established through testing of the implemented features and their interactions.

3.2 Functional evaluation

Functional evaluation examines whether the implemented modules behave according to their requirements. Representative test scenarios for each module are summarised in Table 2. The results should be documented based on the observed behaviour of the application, and features that have not been tested should be clearly identified as pending evaluation.

Test ID	Module	Test scenario	Expected behaviour
T1	Authentication	Login with valid and invalid credentials	Valid user receives a token; invalid user rejected with message
T2	Access control	Buyer attempts to create a listing or open the review queue	Request denied with a permission error
T3	Document upload	Upload an unsupported or oversized file	File rejected with a clear validation error
T4	AI extraction	Upload a sale deed with declared details	Fields extracted; mismatches and uncertain fields flagged
T5	Admin review	Approve with an incomplete checklist; reject without a reason	Both actions blocked with an explanatory message
T6	Recommendation	Search by location, purpose, budget, and area range	Only matching listings returned, ranked by score
T7	Favourites	Add and remove a listing from favourites	Only the buyer's own favourites shown

Table 2. Representative functional test scenarios for the proposed system.

Any failed tests should be investigated and repeated after corrections are made. The extraction module in particular should be evaluated on a separate set of labelled documents, and its accuracy reported per field rather than only as a single overall figure. The recommendation module should be evaluated using listings of known suitability.

3.3 Comparison with conventional approaches

Conventional approaches may require buyers to search on classified portals, ask brokers for documents, and consult advocates for each plot. The proposed system aims to bring listing, document-assisted verification, and recommendation together in a common interface, with a visible verification label and an explanation for each recommendation, potentially improving trust and accessibility.

The inclusion of AI-based extraction provides an opportunity to reduce manual reading effort. However, the usefulness of this approach depends on document quality and the verified performance of the model, and the administrator's decision remains the only basis for a verified label. The proposed system should not be considered more reliable than existing verification services without comparative testing.

Discussion

The proposed architecture provides a framework for integrating listing management, document-assisted verification, and personalised recommendation. A centralized interface can make verified land easier to find, while modular development can simplify maintenance and the addition of new document types or scoring rules.

The system also faces practical challenges. AI extraction may misread low-quality scans, handwritten entries, or regional-language documents, and it cannot confirm that a document is genuine. Matching depends on consistent location names, units, and spelling, and a text-based location filter may miss alternative spellings. Records with unknown price or area units need careful handling, so the application converts values to common units and applies a documented rule to older records.

Privacy and security are important because the application handles personal and property documents. Suitable authentication, authorization, secure document storage with temporary access links, audit logging, and safe handling of uploaded files are therefore necessary [11]. Further evaluation should measure extraction accuracy, recommendation relevance, response time, and the usability of the application. These measurements would provide stronger evidence of the system's effectiveness in practical situations.

4. CONCLUSION

The AI Land Verification and Recommendation Platform presents a technology-based approach to helping buyers find suitable and trustworthy land through a centralized application. The proposed system combines land listing, AI-assisted document extraction, administrator review, and preference-based recommendation, with scope for additional features such as notifications and richer search.

The modular architecture provides a foundation for integrating these functions while considering usability, privacy, and data security. However, the practical effectiveness of the application depends on the correct implementation and testing of its individual components, the quality of the submitted documents, and the accuracy of the extraction model.

Future development can focus on supporting regional-language documents, cross-checking details against official land-record services, adding map-based search, detecting forged or altered documents, introducing notifications and messaging between buyers and owners, and improving ranking with feedback from buyer behaviour. Further testing and user evaluation are required to determine the reliability of the system on real-world documents.

In conclusion, the proposed application offers a foundation for developing an integrated land verification and recommendation system. With appropriate implementation, validation, and privacy safeguards, it can contribute to making land purchase more transparent and accessible for buyers and owners.

5. ACKNOWLEDGMENTS

The authors would like to thank [College Name] for providing the academic environment and facilities required for this project. The authors also express their sincere gratitude to the Head of the Department of [Department Name], the project guide, and the faculty members for their guidance, encouragement, and support throughout the development of the project.

6. REFERENCES

1. Google Firebase. Cloud Firestore Documentation. Available at: https://firebase.google.com/docs/firestore
2. React. React Documentation. Available at: https://react.dev/
3. Express. Express – Node.js Web Application Framework. Available at: https://expressjs.com/
4. Google. Gemini API Documentation. Available at: https://ai.google.dev/gemini-api/docs
5. Cloudinary. Cloudinary Documentation. Available at: https://cloudinary.com/documentation
6. M. Jones, J. Bradley, and N. Sakimura, "JSON Web Token (JWT)," RFC 7519, Internet Engineering Task Force, 2015.
7. N. Provos and D. Mazières, "A Future-Adaptable Password Scheme," in Proc. USENIX Annual Technical Conference, 1999.
8. F. Ricci, L. Rokach, and B. Shapira, Eds., Recommender Systems Handbook, 3rd ed. New York, NY, USA: Springer, 2022.
9. Tailwind Labs. Tailwind CSS Documentation. Available at: https://tailwindcss.com/docs
10. Government of India. The Registration Act, 1908 (Act No. 16 of 1908).
11. OWASP Foundation. OWASP Application Security Guidance. Available at: https://owasp.org/