

INTELLIGENCE CORP — ENTERPRISE AI RESEARCH, PLANNING AND REPORT GENERATION SYSTEM

¹Madiha Muneerahmed Khazi, Student, Department of Computer Science, Bagalkot University, Jamkhandi-587301,

²Prof. Dayanand G Savakar, Professor, Department of Computer Science, Bagalkot University, Jamkhandi-587301,

³Mr. Vijaykumar Bellundagi, Faculty, Department of Computer Science, Bagalkot University, Jamkhandi-587301,

¹madiahkhazi21@gmail.com, ²dgsavakar@gmail.com, ³vijaykumarsb26@gmail.com

Abstract : Preparing accurate research and business reports is traditionally a slow, manual process that depends heavily on the writer's skills. This paper presents Intelligence Corp, a secure, web-based, multi-agent Artificial Intelligence system that automates research, planning, and report writing for organisations. The system employs three cooperating AI agents — a Researcher Agent that gathers information from external sources, a Planner Agent that structures the gathered material, and a Writer Agent that produces the final formatted content — built using the CrewAI and LangChain frameworks. A React.js frontend and a Python FastAPI backend provide secure, role-based access for normal users, managers, and administrators, along with task tracking, a manager approval workflow, notifications, and project collaboration. The completed report is automatically exported as a downloadable PDF or PowerPoint file. The system was implemented and validated through unit, integration, system, security, and validation testing, all of which were successfully passed, confirming that the proposed architecture reduces manual reporting effort while retaining human oversight through the approval workflow.

IndexTerms - Artificial Intelligence, Multi-Agent Systems, Automated Report Generation, CrewAI, LangChain, FastAPI, React.js, Role-Based Access Control.

I. INTRODUCTION

In today's organisations, research and business reports are usually prepared by hand: an employee searches for information, reads and organises it, and then writes a clean report. This process is slow, and the quality of the finished report depends heavily on the skill and effort of the person writing it. Intelligence Corp automates this entire process using Artificial Intelligence. Instead of one person doing everything, the system uses three cooperating AI agents that work together like a small team — a Researcher Agent that gathers information from the internet and external data sources, a Planner Agent that studies this information and builds a clear structure for the report, and a Writer Agent that writes the final content in proper language and formatting. Once the agents finish their work, the system automatically produces a downloadable PDF or PowerPoint report.

The name "Intelligence Corp" reflects the two ideas behind the project: "Intelligence" refers to the AI agents that perform the research and writing, while "Corp" reflects that the system is built for corporate and institutional use, with proper accounts, roles, and approval steps, rather than being a casual public tool. Unlike a plain chat-based AI assistant, Intelligence Corp remembers what has been researched before, keeps a complete history of every task and report, and lets more than one person collaborate on the same project, making it closer to a proper internal software tool than to a simple chatbot.

The specific objectives of the project were to: (i) design a secure, role-based web platform for organisational report generation; (ii) build a multi-agent AI pipeline capable of researching, planning, and writing a report with minimal human effort; (iii) provide a manager approval workflow so that AI-generated content is reviewed by a human before it is treated as final; and (iv) automatically export the finished report as a formatted PDF or PPT file. The remainder of this paper is organised as follows. Section II discusses the existing manual process and its limitations. Section III describes the proposed system architecture and methodology. Section IV describes the implementation. Section V presents the testing carried out and the results obtained. Section VI concludes the paper, and Section VII discusses its limitations and scope for future enhancement.

II. EXISTING SYSTEM AND MOTIVATION

Before building this system, report preparation in most organisations followed a fully manual process. An employee would search the internet or internal files for relevant information, copy it into a document, read through the material to decide what was important, organise it into sections, and finally write and format the report, often in Word or PowerPoint. This manual approach has several drawbacks: it takes a long time, often several days for one detailed report; the quality of the output depends on the skill and mood of the individual writer; there is no fixed structure, so reports from different employees look inconsistent; and there is no easy way to track who approved a report or which version is final.

Existing AI frameworks provide capabilities for agent development, language-model interaction, prompt processing, and multi-agent coordination. However, these capabilities are generally provided as separate frameworks rather than as a complete organisational web application. **Intelligence Corp was motivated by the need to integrate CrewAI, LangChain, and AutoGen with a full-stack web platform, where Researcher, Planner, and Writer agents work together for automated report**

generation while the system also provides role-based access, task management, human approval, notifications, PDF/PPT export, and persistent history. This integration combines AI-based automation with organisational workflow and human oversight in a single platform.

III. SYSTEM ARCHITECTURE AND METHODOLOGY

3.1 Development Methodology

The system was built following an iterative Software Development Life Cycle (SDLC) model. Requirements were first gathered and documented as a formal Software Requirements Specification (SRS), covering functional requirements, design constraints, and quality attributes such as security, reliability, usability, maintainability, scalability, and portability. Each iteration added one functional module — authentication, task management, the AI agent pipeline, report generation, and the approval workflow — which was designed, implemented, and tested before the next module was started, allowing problems to be caught and fixed early.

3.2 System Architecture

Fig. 1 shows the overall architecture of Intelligence Corp. The client layer (the React.js web application) communicates with the application layer using secure REST API calls over HTTPS. The application layer, built with FastAPI, authenticates every request using a JSON Web Token (JWT) and checks the user's role before performing any action. For tasks that require AI processing, the application layer invokes the AI agent layer, which in turn calls external services such as a large-language-model API, a web search API, a news API, and a financial-data API to gather real information. All structured data is stored in the database layer, using SQLAlchemy as the Object-Relational Mapping (ORM) layer over SQLite/PostgreSQL, while a separate vector database (ChromaDB) stores embeddings used by the Researcher Agent, and generated report files are kept in a dedicated file-storage layer.

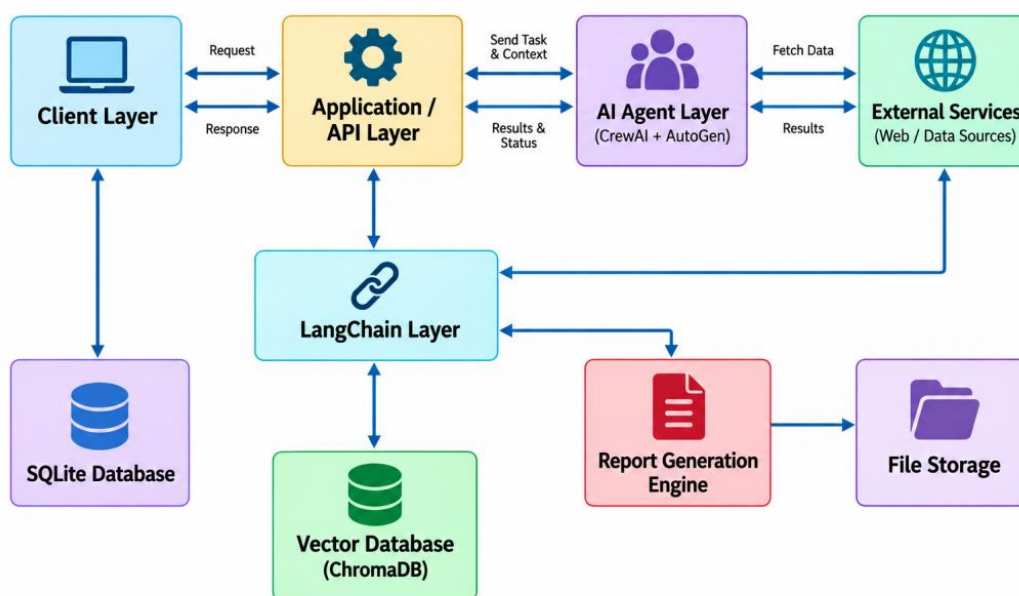


fig. 1: system architecture of intelligence corp

3.3 Functional Requirements

The functional requirements, derived from studying how report preparation happens in an organisation, were grouped module-wise as summarised in Table I. Each requirement was later mapped to one or more backend API routes and frontend screens.

table i: summary of functional requirement modules

Module	Key Functional Behaviour
Authentication	Secure registration, login, logout and password reset using JWT tokens
Task Management	Create, view, update and track research/report tasks
AI Agent Pipeline	Automatically runs Researcher → Planner → Writer agents in sequence
Report Generation	Exports completed tasks as PDF and/or PPT files
Approval Workflow	Routes a report to a manager for approval via a secure email link
Project Management	Create projects, add members and exchange project-level chat

Module	Key Functional Behaviour
Admin Panel	Manage user accounts, roles, API keys and activity logs
Notifications & History	In-app alerts and a searchable history of past tasks

3.4 Database Design

The database was designed around nine core tables — Users, Tasks, Reports, Projects, Project_Members, Email_Approvals, Search_History, File_Uploads, and Notifications — connected through primary and foreign keys, as shown in the Entity-Relationship diagram in Fig. 2. A user can own many tasks; each task produces one or more report versions; a report can generate an email approval record; and a project links many users through the Project_Members table, allowing collaborative work on the same set of tasks and reports.

Intelligence Corp Enterprise ER Diagram

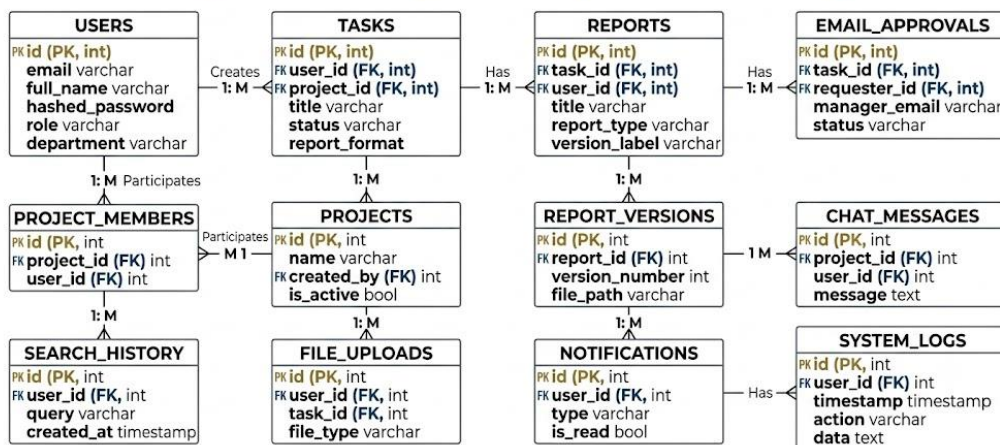


fig. 2: Entity-relationship diagram of the Intelligence Corp database

3.5 Module-Level Design

At the detailed design level, the backend is organised into an Authentication module (registration, login, JWT issuance and role checks), an AI Agent module (crew_manager.py, researcher_agent.py, planner_agent.py and writer_agent.py, each responsible for one stage of the pipeline), a Report module (formatting Writer-agent output into PDF using ReportLab and into PPT using python-pptx), and an Approval module (secure token generation, email dispatch and status update on approval or rejection). The frontend is similarly organised into reusable pages and components for the dashboard, task creation, report viewing, and the admin panel, so that a change in one module does not require understanding the entire codebase.

IV. IMPLEMENTATION

4.1 Technology Stack

The backend is implemented in Python 3.10+ using the FastAPI framework, with SQLAlchemy as the ORM over a SQLite database (PostgreSQL for larger deployments) and ChromaDB as the vector store. The AI agent pipeline is built using the CrewAI and LangChain frameworks. The frontend is implemented in JavaScript using React.js. Passwords are hashed before storage, sessions are managed with JWT tokens with a limited lifetime, and role checks are enforced on the backend, preventing permissions cannot be bypassed by directly calling the API.

4.2 AI Agent Pipeline

When a user creates a task, the Crew Manager passes it through the three agents in sequence. The Researcher Agent gathers information from external sources and any supporting files uploaded by the user. The Planner Agent studies the gathered material and produces a structured outline. The Writer Agent expands this outline into the final report text. The task status is updated at each stage — Pending, Researching, Planning, Writing, completed — and is shown to the user through a progress indicator; if an external data source is temporarily unavailable, the Researcher Agent continues using the remaining available sources rather than failing the whole task.

4.3 User Interface

The interface follows a consistent layout — a top bar, a side menu, and a main content area — across all screens, so that a user who learns one screen can easily use the rest. Fig. 3 shows the main dashboard, which gives a quick summary of a user's tasks and reports and provides access to task creation, report viewing, search history, notifications and, for administrators, user management and system settings.

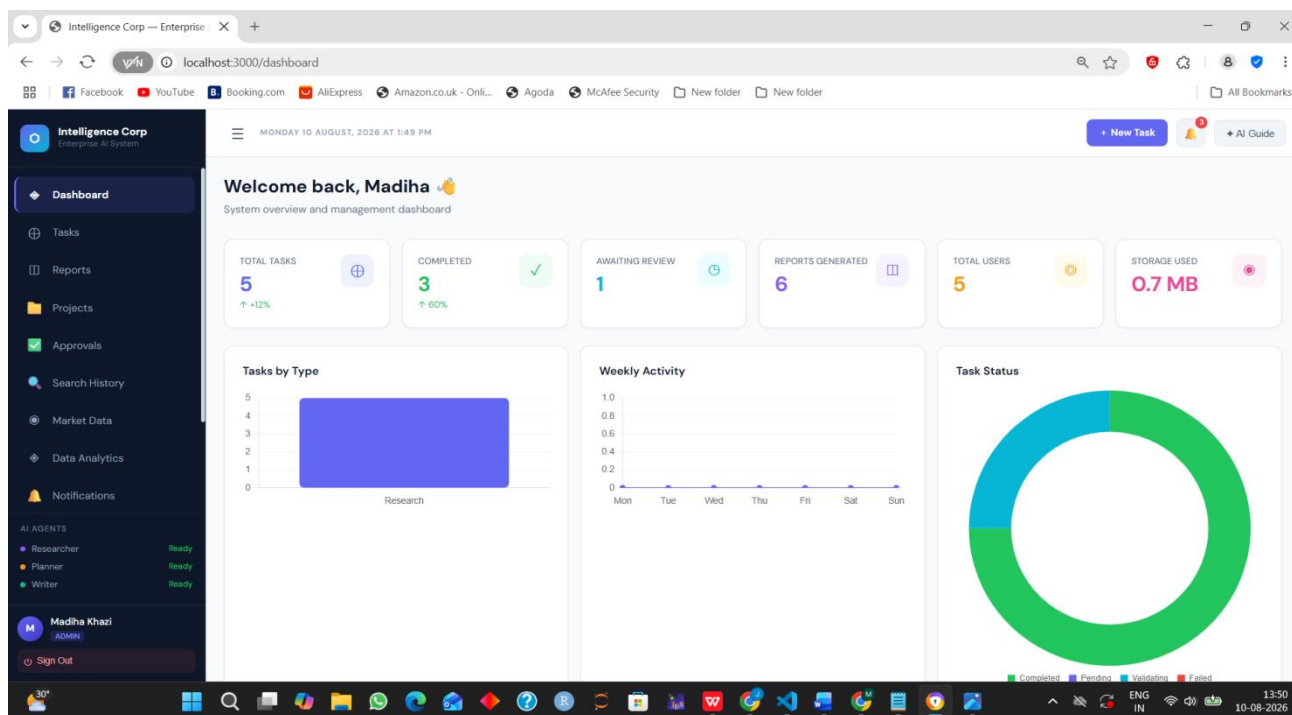


fig. 3: main dashboard screen of Intelligence Corp

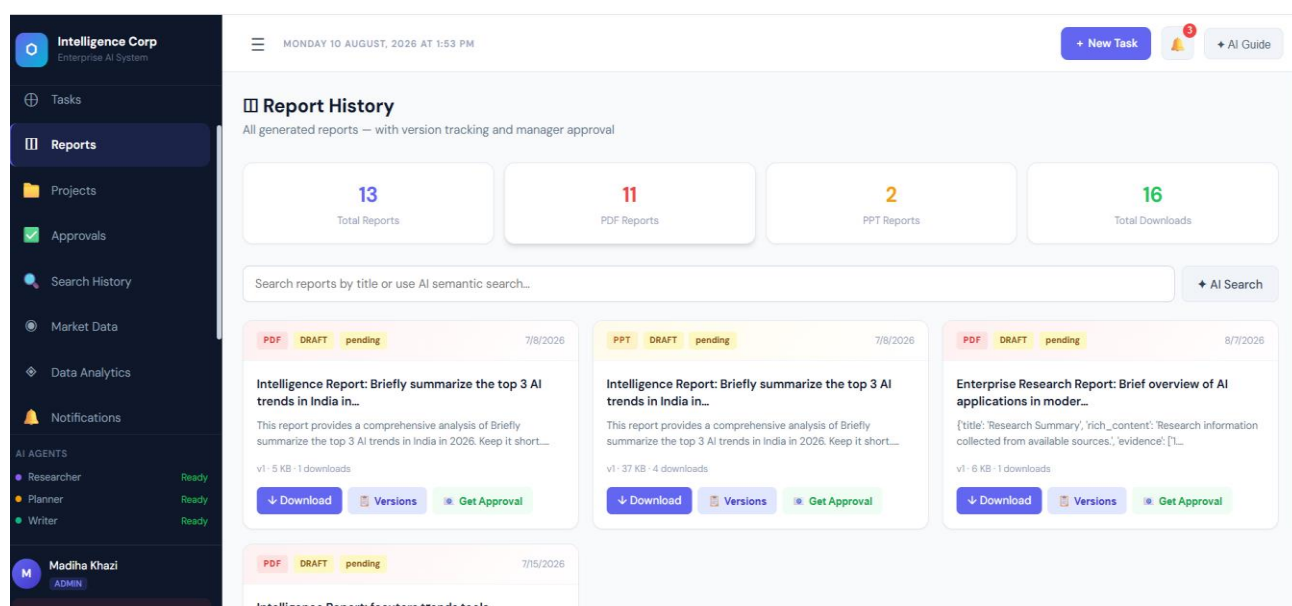


Fig. 4: Report screen of Intelligence Corp

V. TESTING AND RESULTS

Testing was carried out continuously alongside development rather than only at the end, following the iterative approach described in Section III-A. Five levels of testing were performed: unit testing of individual functions such as password hashing and token creation; integration testing of module interactions such as the AI-agent pipeline and the manager-approval email flow; system testing of complete end-to-end flows from registration to report download; security testing of authentication and role-based access control; and validation testing of form-level input checks. A representative summary of the twenty system test cases executed is given in Table II; all listed test cases, and all test cases across the five testing levels, produced a Pass result.

table ii: representative system test cases and results

Test Case	Expected Result	Result
User registration with valid details	Account created successfully	Pass
Login with correct credentials	JWT issued, dashboard shown	Pass
Create task and run AI pipeline	Status moves Researching → Planning → Writing → Completed	Pass
Report generation (PDF / PPT)	Downloadable file created and linked to task	Pass
Send report for manager approval	Secure approval email sent	Pass
Manager approves / rejects report	Report status updated, user notified	Pass
Login with wrong password	Access denied with error message	Pass
Normal user opens admin panel	Access denied, redirected	Pass
Duplicate email registration	Error shown, account not created	Pass
Oversized file upload	Upload rejected with error message	Pass

The consistent Pass results across unit, integration, system, security and validation testing confirm that the requirements identified in Section III-C were successfully met by the implemented system, and that the role-based access control and manager-approval workflow correctly enforce the intended organisational controls.

VI. CONCLUSION

This paper presented Intelligence Corp, a secure, web-based multi-agent AI system that reduces the time and manual effort involved in researching, planning, and writing organisational reports. By combining a role-based web application with three cooperating AI agents — the Researcher, the Planner, and the Writer — the system converts a simple task request into a complete, well-structured, formatted report without requiring the user to perform the research and writing personally. The project demonstrates that modern web technologies (React.js and FastAPI) can be combined with AI agent frameworks (CrewAI and LangChain) to build a practical tool that still keeps a human reviewer in control of the final decision through its manager-approval workflow. Testing confirmed that the system meets its functional and security requirements, showing that AI can meaningfully reduce repetitive manual work without removing human judgement from the process.

VII. LIMITATIONS AND FUTURE SCOPE

The quality of an AI-generated report depends on the availability of external data sources and a stable internet connection, and the current system supports only one organisation at a time, uses built-in PDF/PPT templates, and generates content in English only. Planned enhancements include multi-tenant support so that a single deployment can serve several organisations with isolated data, additional report languages and custom templates, predictive analytics for estimating task completion time, voice-based task creation, and a public API so that other internal systems can request reports programmatically. Among these, multi-tenant support is considered the highest priority, since it would increase the platform's value without requiring a separate installation for every client organisation.

ACKNOWLEDGMENT

The author sincerely thanks the Department of Computer Science, Bagalkot University, for the guidance and resources provided during the development of this project.

REFERENCES

- [1] [Autogen: Enabling next-gen llm applications via multi-agent conversation](#)
Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., ... & Wang, C. (2023). Autogen: Enabling next-gen llm applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*.
- [2] [Chatdev: Communicative agents for software development](#)
Qian, C., Liu, W., Liu, H., Chen, N., Dang, Y., Li, J., ... & Sun, M. (2024, August). Chatdev: Communicative agents for software development. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 15174-15186).
- [3] [MetaGPT: Meta programming for a multi-agent collaborative framework](#)
Hong, S., Zhuge, M., Chen, J., Zheng, X., Cheng, Y., Wang, J., ... & Schmidhuber, J. (2024, May). MetaGPT: Meta programming for a multi-agent collaborative framework. In *International Conference on Learning Representations* (Vol. 2024, pp. 23247-23275).

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). ReAct: Synergizing Reasoning and Acting in Language Models. ICLR 2023. *arXiv preprint arXiv:2210.03629*.

[5] A survey on large language model based autonomous agents

Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., ... & Wen, J. (2024). A survey on large language model based autonomous agents. *Frontiers of computer science*, 18(6), 186345.

Copyright & License:



© Authors retain the copyright of this article. This work is published under the Creative Commons Attribution 4.0 International License (CC BY 4.0), permitting unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.