Project Atlas
An Autonomous LinkedIn Job Application System

Rahul

February 2, 2026

Abstract

Job Application workflows on networking platforms (like LinkedIn) are highly influenced by Applicant Tracking Systems (ATS) making manual job searching and applications both time consuming and inefficient. This document presents **Project Atlas** an autonomous ATS-aware job application particularly designed to operate on LinkedIn for user configurable roles or positions. This system uses controlled web automation, AI-assisted job evaluation using Google Gemini (Free Tier) and containerized execution to form a reproducible and modular pipeline.

Over a 10-day deployment period, Project Atlas has shown consistent automation while respecting platform safety and API limitations. This document serves as both technical as well as applied research-style reporting of the system's design, constraints and real-world performance.

1 Introduction

Many recruitment platforms provide candidates a large and continuously changing set of job postings. However, most of the applications are passed through Applicant Tracking Systems (ATS) prior to any human review. These systems apply structured filtering mechanisms based on already predefined criteria like keyword presence, role alignment and required years of experience, which significantly reduces candidate visibility at early stages of hiring process.

The job-hunting process is mostly repetitive in nature. Candidates repeatedly search for similar roles, examine comparable job descriptions and apply for multiple roles with minor variations. This repetition is largely manual and time-consuming, especially when performed across multiple platforms and organizations.

ATS-based filtering further amplifies this inefficiency by limiting transparency. Candidates hardly receive little or no feedback regarding screening outcomes, making it difficult to determine whether rejections result from skills mismatch, CV structure, missing keywords or just automated filtering rules. As a result, this substantial effort is often spent on applying jobs that are unlikely to progress beyond any initial ATS screening.

Most existing tools focus on isolated components in job-hunting process, such as resume formatting, keyword optimisation or job aggregation. These tools generally lack job evaluation logic and interpretation of job descriptions. As a result, candidates get limited support in reducing ineffective applications.

2 Problem Statement

Most Manual job application workflows are inefficient and difficult to scale. Each application needs repeated evaluation of similar job descriptions and manual resume adjustments which leads to higher time consumption and higher cognitive costs per applications.

These workflows mostly lack ATS-aware relevance assessments. Candidates often apply to jobs without considering automated screening criteria which makes the application unable to pass initial ATS Filtering despite being suitable.

The repetitive nature of the manual applications also contributes to fatigue and inconsistency, further limiting number of daily applications. Although automation may reduce this burden, but naïve automation approach due to unreliable execution and policy violation by non-human interaction patterns can risk the platform detection.

Project Atlas addresses the challenge of reducing manual inefficiency through an ATS aware automation which operates within the platform safeguards and real-world API constraints, prioritizing controlled human-like behaviour over application volume.

3 System Overview

Project Atlas is implemented as a sequential, modular pipeline which had a single execution entry point. The system is designed by ensuring clear stage separation, controlled execution flow and reproducibility.

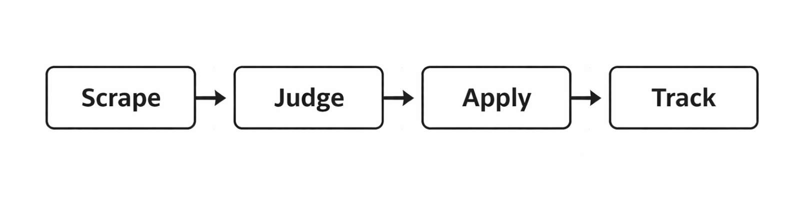


Figure 1: Pipeline sequence

Every pipeline execution is initiated from a central controller (`main.py`). This single point of entry simplifies the testing, logging and system level control while allowing each individual component to evolve independently.

Each of the stage is implemented as an isolated modules with well defined inputs and outputs, allowing independent testing, replacement or extension without affecting the overall system.

The project Atlas is deployed as a docker based application to ensure environment consistency, dependency isolations and reproducible execution across runs. Containerization also ensures controlled resource consumption and predictable behaviour during automated execution.

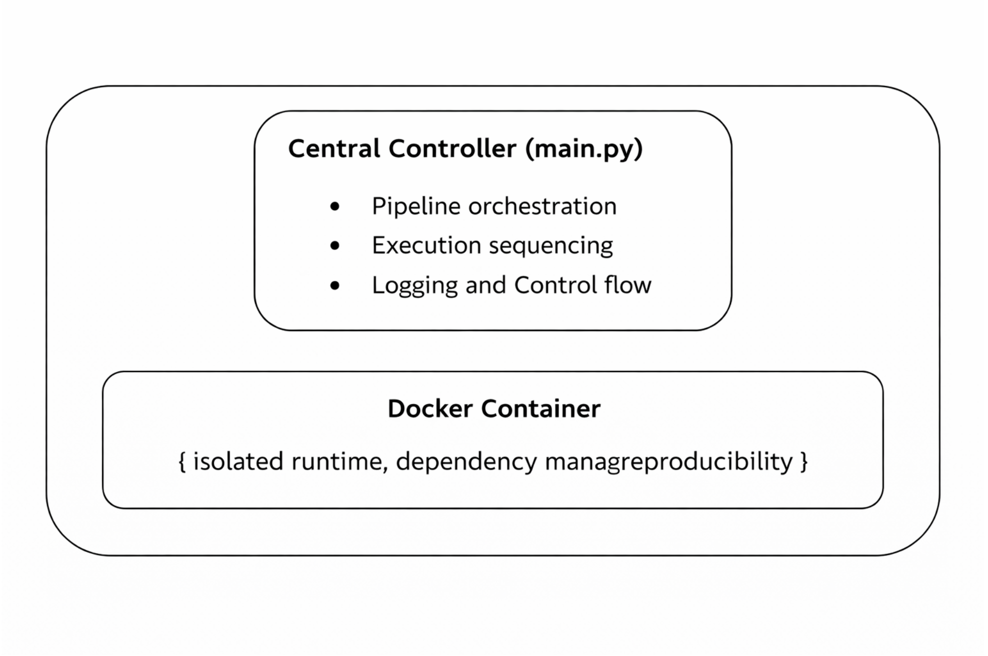


Figure 2: Controller and Environment

4 Target Scope and Constraints

4.1 Target Role and Platform

- Role focus: Software Engineering Associate (user-Configurable).
- Platform: LinkedIn Only.
- Application method: Easy Apply.

- Scope limitation: No multiple-role or multiple-platform optimization.
- Observed constraint: It was noticed that many roles labelled as entry-level or associate level still specifies 3-5 years of experience, reflecting common HR tagging practices rather than entry level screening criteria.

4.2 AI and API Constraints

- AI model: Google Gemini (Free Tier).
- Daily scan limit: around 20 jobs (configurable with paid tiers).
- Constraints: Strict rate and quota limit due to free tier usage.
- Design implication: Job evaluation prioritizes and relevance over application volume.

5 Implementation Details

5.1 Job Scraping and Automation

Playwright-based browser automation collects:

- Job posts are collected using **Playwright-based browser** automation.
- Automation is limited to publicly available metadata only:
  - Job title
  - Company name
  - Location
  - Application link
- Design decisions:
  - Human like interaction patterns were achieved by controlled delays and sequential actions.
  - No forced scraping or parallel execution is performed.
  - Automation avoids direct API calls to platform endpoints to reduce detection risks.

5.2 ATS-Aware Job Evaluation

- Job descriptions are evaluated using Google Gemini to simulate ATS-style screening logic.
- This Evaluation compares each job description against structured candidate profile.
- The system returns a binary decision:
  - YES: Job is considered suitable for the user.
  - NO: Job is filtered out not suitable for the user.
- Design Decisions:
  - Strict filtering rules are applied due to daily quota limits.
  - Few applications are submitted but with higher confidence.
  - Resume reuse is enforced within in a single run:
    - One CV is used across all evaluated jobs in that run.
    - Different CVs can be supplied for different runs but cannot be swapped dynamically in mid run.

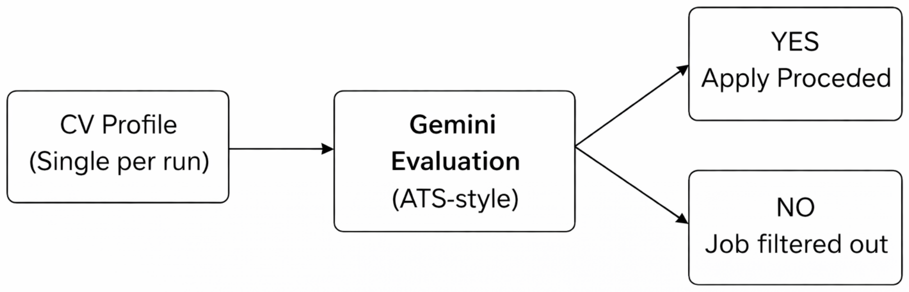


Figure 3: ATS evaluation flow

6 Execution and Orchestration

- All system components have a single entry point (`main.py`).
- Design decisions:
  - Preplanned execution order across all modules.
  - Centralized control to prevent uncontrolled automation behaviour.
- Application results and outcomes are logged in an Excel file containing:
  - Applied roles
  - Evaluation outcomes
  - Timestamps and metadata
  - Job description link
- This enables post-run analysis and auditing without external databases.
- The system avoids hardcoding or exposing API secrets and platform credentials.
- The entire application is containerized using Docker to make sure:
  - Environmental consistency
  - Dependency isolation
  - Reproducible execution across runs

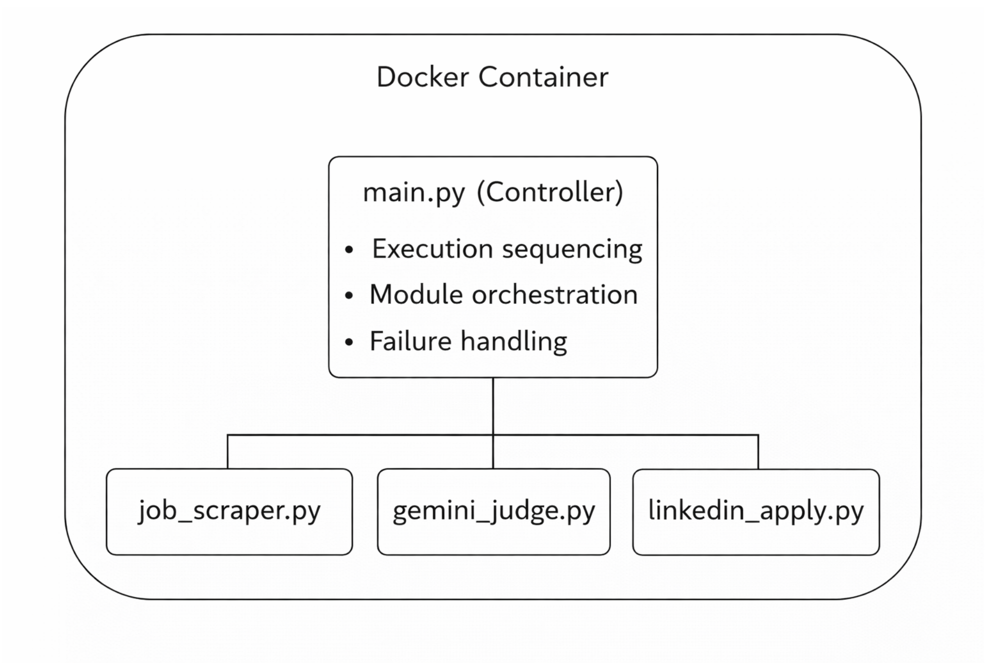


Figure 4: Docker based execution model

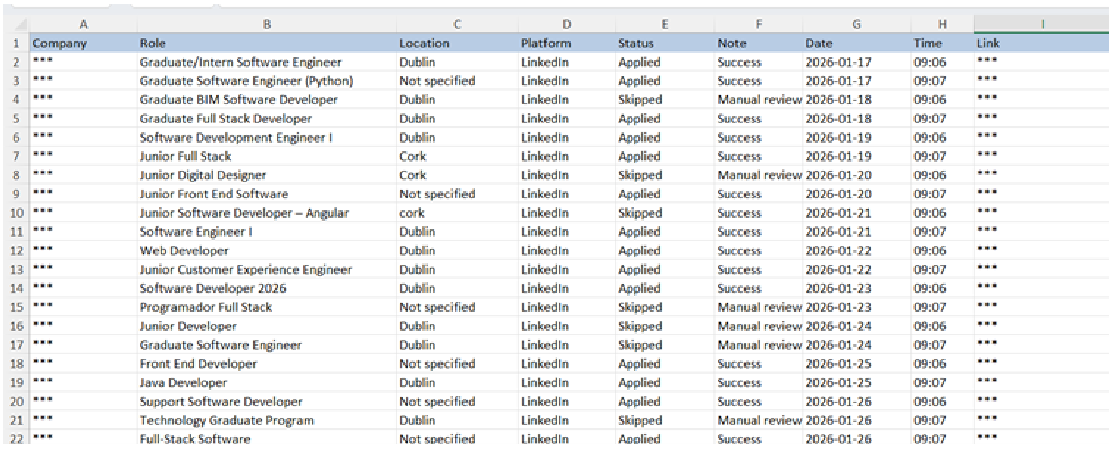
7 Evaluation and Observations

7.1 Deployment Period

- Duration: 10 days
- Daily scans: ∼20 jobs
- Applications/day: 3–4

7.2 Observations

- The application had consistently completed the full pipeline across all the runs without crashes.
- ATS filtering had reduced the application volume, but higher relevance submissions.
- Manual effort was reduced to post application follow ups only, eliminating repetitive steps such as job screening and eligibility checking.
- End-to-end automation of the easy apply jobs was reliably handled for standard submissions. Applications that introduced additional, non-standard questions were not force submitted. These applications were saved as drafts and flagged for manual review.
- A huge reduction in total time spent per application was observed due to removal of manual filtering and navigation.



	A	B	C	D	E	F	G	H	I
1	Company	Role	Location	Platform	Status	Note	Date	Time	Link
2	***	Graduate/Intern Software Engineer	Dublin	LinkedIn	Applied	Success	2026-01-17	09:06	***
3	***	Graduate Software Engineer (Python)	Not specified	LinkedIn	Applied	Success	2026-01-17	09:07	***
4	***	Graduate BIM Software Developer	Dublin	LinkedIn	Skipped	Manual review	2026-01-18	09:06	***
5	***	Graduate Full Stack Developer	Dublin	LinkedIn	Applied	Success	2026-01-18	09:07	***
6	***	Software Development Engineer I	Dublin	LinkedIn	Applied	Success	2026-01-19	09:06	***
7	***	Junior Full Stack	Cork	LinkedIn	Applied	Success	2026-01-19	09:07	***
8	***	Junior Digital Designer	Cork	LinkedIn	Skipped	Manual review	2026-01-20	09:06	***
9	***	Junior Front End Software	Not specified	LinkedIn	Applied	Success	2026-01-20	09:07	***
10	***	Junior Software Developer – Angular	cork	LinkedIn	Skipped	Success	2026-01-21	09:06	***
11	***	Software Engineer I	Dublin	LinkedIn	Applied	Success	2026-01-21	09:07	***
12	***	Web Developer	Dublin	LinkedIn	Applied	Success	2026-01-22	09:06	***
13	***	Junior Customer Experience Engineer	Dublin	LinkedIn	Applied	Success	2026-01-22	09:07	***
14	***	Software Developer 2026	Dublin	LinkedIn	Applied	Success	2026-01-23	09:06	***
15	***	Programador Full Stack	Not specified	LinkedIn	Skipped	Manual review	2026-01-23	09:07	***
16	***	Junior Developer	Dublin	LinkedIn	Skipped	Manual review	2026-01-24	09:06	***
17	***	Graduate Software Engineer	Dublin	LinkedIn	Skipped	Manual review	2026-01-24	09:07	***
18	***	Front End Developer	Not specified	LinkedIn	Applied	Success	2026-01-25	09:06	***
19	***	Java Developer	Dublin	LinkedIn	Applied	Success	2026-01-25	09:07	***
20	***	Support Software Developer	Not specified	LinkedIn	Applied	Success	2026-01-26	09:06	***
21	***	Technology Graduate Program	Dublin	LinkedIn	Skipped	Manual review	2026-01-26	09:07	***
22	***	Full-Stack Software	Not specified	LinkedIn	Applied	Success	2026-01-26	09:07	***

Figure 5: Application tracking snapshot (Masked)

8 Limitations

The Following limitations of The Project Atlas were observed:

- API Constraints: AI-based evaluation is limited by free-tier quota and rate restriction. However, it could be lifted with paid version.
- Filtering Trade-off: ATS screening reduces overall application volume.
- Platform dependency: Current version of Project Atlas depends on the stability and consistency of the LinkedIn user interface.
- Evaluation Subjectivity: This ATS-style screening may not reflect every recruiter preference.

These limitations are treated as intentional design trade-offs rather than implementation flaws.

9 Ethical and Responsible Use

Project Atlas is designed for responsible and compliant usage:

- Platform Compliance: This Application doesn't attempt to bypass any safeguards or restrictions. It is intended to operate within platform terms and conditions.
- No misuse behaviour:
  - No credential bypassing
  - No impersonation
  - No rate limit evasion
- Human oversight: Human review is required where application asked for additional questions beyond the system awareness.
- UI level support: The system depends on LinkedIn's existing Easy Apply interface which auto-fills common applicant information. When unfamiliar or additional information is required then application gets deferred to manual review rather than force-submit.

10 Future Enhancements

The potential future improvement includes:

- User interface support: Introduction of a lightweight UI to review decision, config Job searching and inspect application outcomes.
- Muti-resume support: Support for selecting different resume across a single run or selecting resume to specific role categories.
- AI-assisted resume tailoring: Use of AI to adapt resume content to job description while preserving transparency and user oversight.

11 Conclusion

Project Atlas was developed to examine whether ATS-aware automation could reduce the inefficiencies and repetitive effort involved in manual job applications while remaining compliant with platform constraints. The system was designed as a controlled, modular pipeline rather than a high-volume automation tool.

During a 10-day deployment, Project Atlas demonstrated stable execution, reduced manual intervention and practical use of AI-assisted evaluation under real-world API and platform limitations. This project illustrates how responsible automation and human oversight can be combined to improve efficiency without bypassing safeguards.

12 Appendix

12.1 Source Code and Reproducibility

The complete source code is publicly available:
`https://github.com/Rahul9315/Job_Hunting_AI_Bot.git`

This repository contains the modular pipeline implementation, configuration files and documentation required to reproduce the system behaviour. Sensitive information such as credentials and API keys is excluded.