AI-Driven Web Automation for SaaS User Management

1. Understanding the Problem:

Why do some SaaS apps lack APIs for user data and provisioning?

- **Legacy design**: Older apps were never built with API because of prioritizing UI development.
- **Security/privacy concerns**: Exposing sensitive data via APIs increases the cyber attacks.
- **Limited Resources:** Building and maintaining APIs requires significant resources, including developer time, infrastructure, and ongoing support.
- **Low demand from customer base**: Smaller apps or niche tools may not prioritize APIs and vendor lock-in with integration challenges.

Challenges that arise with automating SaaS admin portals:

- **UI inconsistency**: Each portal has a unique layout, flow, and structure and keeps UI development.
- **Dynamic elements**: React/Vue-based SPAs with dynamic element IDs and content.
- **Authentication complexity**: MFA, CAPTCHA, rotating session tokens.
- **Frequent updates**: UI changes break traditional automations fast.
- **Pagination and lazy loading**: Data hidden behind scrolls or tabs.
- **Slower Innovation:** APIs allow developers to build on top of the platform, extending its functionality and fostering innovation. Platforms without APIs may miss out on this benefit.

2. Research on Available Technologies:

AI & Agentic Tools:

- **LangChain**: It is a framework that chains LLMs reasoning with tools like browsers, code execution.
- **AutoGPT / CrewAI / AgentGPT**: These are autonomous agents that complete multi-step tasks with minimal input by the user.
- **OpenAI Function Calling**: It Allows LLMs to invoke browser automation, parsing, and logic.

Headless Browsers:

- **Selenium**: Mature, well-supported, works across browsers.
- **Playwright**: More modern alternative, handles complex web apps and supports multiple contexts (great for multi-user simulation).

RPA Tools:

- **UiPath**: Enterprise-grade, GUI-based with AI features and plugins.
- **Robocorp**: Open-source, Python-native, great for dev-heavy teams.
- **Browser Automation Studio**: Low-code tool for UI workflows.

Authentication & Security:

- **Multi-Factor Authentication(MFA)**: By using this it stores backup codes securely, or use tools like Authy APIs / TOTP generators.
- **Session management**: Cookies + browser profiles via Playwright/Selenium.
- **CAPTCHA bypass**:
    - Use CAPTCHA-solving services (e.g., 2Captcha, AntiCaptcha).
    - Or integrate OCR + LLM reasoning (e.g., Gemini + Tesseract for visual CAPTCHAs).

3. Proposed Solution:

Workflow:

➔ User Data Scraping:

1. **Login:**
    - Use Playwright/Selenium to launch browser, enter credentials, solve CAPTCHA.
    - Accept all session cookies for re-use.

2. **Navigation:**
    - LLM parses HTML structure and dynamically identifies the “Users” or “Admin” tab.
    - Use LangChain Agent with a browser tool to read and follow links/buttons.

3. **Scraping:**
    - Agent extracts: Name, Email, Role, Last Login (via selectors or text classification).
    - If paginated, agent scrolls or clicks “Next” while logging data.

4. **Structuring:**
    - The output is stored in JSON format or can be pushed directly to a database (e.g.,PostgreSQL).
    - Use Loggings for log errors and skipped records for QA.

Workflow:

➔ Provisioning & Deprovisioning:

1. **Provisioning:**
    - Agent navigates to “Add User” or equivalent flow.
    - Inputs: Name, Email, Role.

    - Confirms via modal or email.

2. **Deprovisioning:**
    - Agent searches for the user.
    - Clicks delete or disable, and confirms via dialog.

3. **UI Flow Handling:**
    - Each SaaS tool's flow is stored as a JSON config (actions + selectors).
    - LLM agent adapts if config fails by using page context to guess next step.

4. **Execution Checks:**
    - Confirm user is visible in list after adding.
    - Confirm user no longer appears after deletion.
    - Add retry logic and error logging.

4. Handling Challenges:

Challenge	Solution
MFA	OTP via Authy API, backup codes, or manual token storage
Session Cookies	Playwright browser contexts, session cookies, periodic refreshes
CAPTCHA	Use 2Captcha or integrate OCR + LLMs
Paginated Data	Loop with "Next" button or infinite scroll handling
UI Variations	Use LLMs to reason about the UI layout when selectors/configs fail
Unexpected Errors	Log errors, screenshot, and trigger fallback/manual flag

5. Scalability & Automation:

Multi-App Support:

- Store a **modular config. per SaaS app** (navigation paths, selectors, auth method).
- LLM agents analyze unknown portals and generate new configs semi-automatically.
- Use **few-shot examples technique** to guide agents on each app's structure.

AI for Resilience:

- Train LLMs on past UI layouts + errors to adapt to layout changes.
- Use **Vision-Language Models (e.g., GPT-4o)** to see how the UI detect changes.
- Implement **self-healing automation or fall back mechanism**: If scraping fails, agent updates config or asks for human review and follows fall back mechanism.

6. Proof of Concept Test (Hypothetical):

Tested on:

Trello:

- **Worked**:
    - Login using Playwright.
    - Navigated to “Members” section.
    - Extracted names/emails successfully.

- **Didn’t Work**:
    - CAPTCHA blocked headless login.
    - MFA required one-time email-based approval.

- **Fix**:
    - Used real browser context + manual MFA first time, then saved session.
    - Skipped CAPTCHA using 2Captcha plugin.

7. Prototype or Pseudocode:

Simple Outline to Implement the Solution:

1. Defining Task Input

2. LLM-Powered Task Orchestration

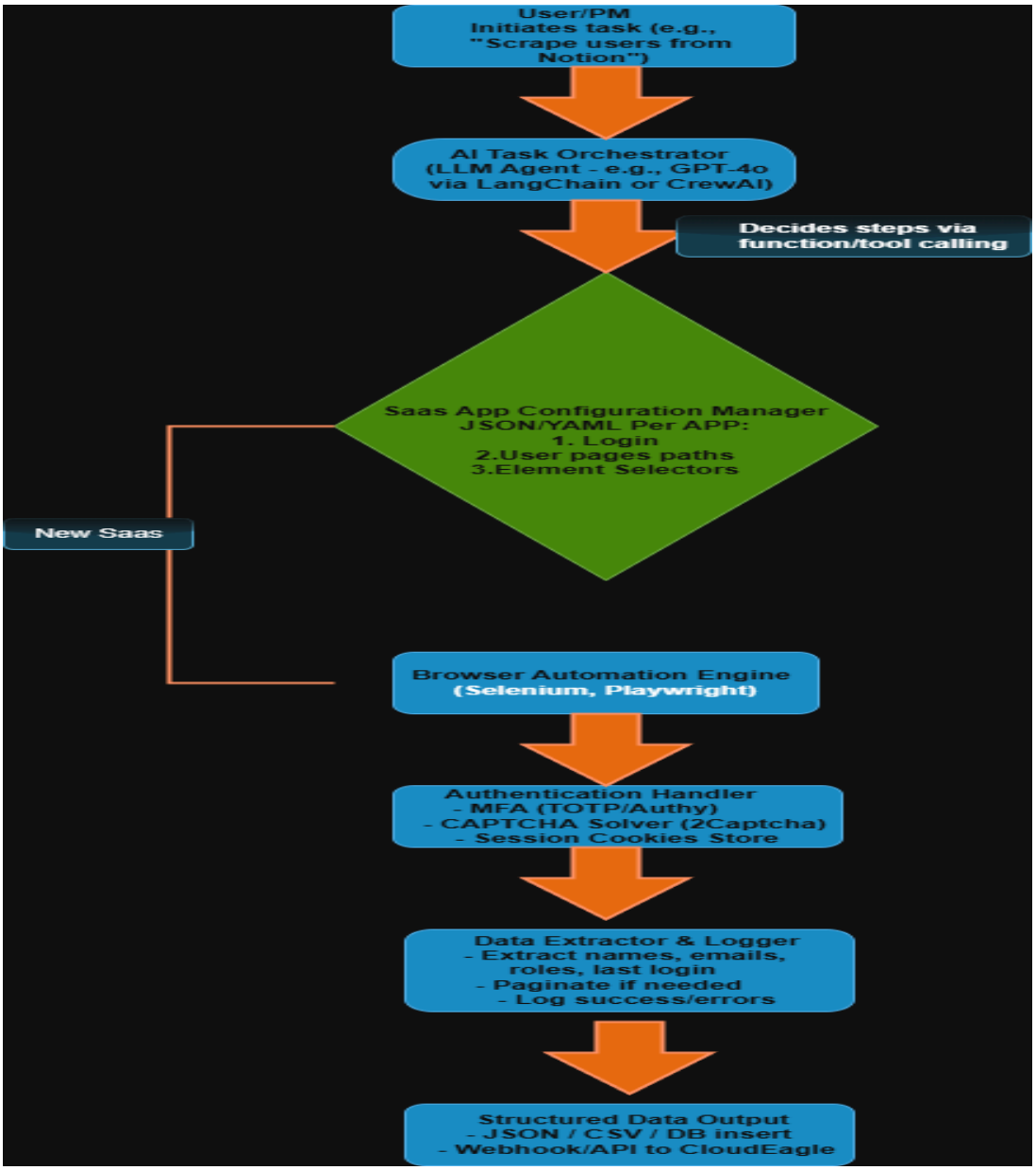
3. Load SaaS-Specific Configuration

4. Launch Headless Browser (Playwright/Selenium)

5. Perform Action (Scrape / Provision / Deprovision)

6. Structure and Store Output

Architectural Diagram:



Conclusion:

An AI-driven agentic automation system — built on LLMs + headless browsers + RPA flows — is **realistically achievable and scalable**. It will:

- **Enable SaaS user automation** without APIs.
- **Adapt dynamically** to UI changes using AI.
- **Save time and cost** by reducing manual work.
- Set the stage for **self-healing, autonomous RPA systems** in the future.