

Project Plan: AI Operator – Local Browser Agent

1. Project Objectives

The AI Operator project seeks to develop a privacy-focused, intelligent browser agent capable of automating routine web-based tasks entirely on a user's local machine. The primary goal is to create a seamless, desktop-based assistant that can handle complex online actions—such as ordering groceries, booking tickets, filling out forms, and navigating websites—through natural language commands, without sending any data to external servers or cloud platforms. This local-first approach ensures user data, credentials, and browsing activity remain completely private and secure.

Unlike traditional automation tools that rely on cloud services or external processing, our agent will function entirely on the user's device. It will integrate a chatbot interface for intuitive communication, web scrapers to analyse open browser tabs, and a locally-hosted large language model (LLM) that interprets user instructions. At the heart of this process lies Microsoft's Semantic Kernel, which will act as the cognitive layer responsible for decomposing high-level instructions into logical and actionable web interactions. This semantic reasoning engine transforms vague or complex user goals into step-by-step sequences that can be executed with precision through browser automation tools like Selenium.

The project aims to combine multiple sophisticated components into a single modular system: a desktop UI for interaction, an LLM for interpreting intent, a task planner for generating execution steps, and browser automation scripts for performing actions. The modular architecture allows for continuous development, testing, and iteration within an agile framework. Each feature will be developed in sprints, enabling rapid feedback loops and measurable progress toward a fully functional minimum viable product (MVP).

Beyond functionality, our objective is to demonstrate that intelligent automation can be both practical and ethical. With rising concerns about data misuse, surveillance, and AI transparency, this project promotes a responsible AI vision where users retain control of their digital interactions. By running entirely on local infrastructure, the AI Operator protects against leaks of sensitive personal data and mitigates risks associated with third-party data handling.

The broader ambition of this project is not just to build a smart assistant, but to redefine what intelligent web automation looks like in the era of AI. It bridges natural language processing, web scraping, real-time planning, and local execution in a coherent system that reflects strong software engineering principles and privacy-by-design values. By leveraging open-source technologies and a clear ethical stance, the AI Operator will offer a powerful alternative to cloud-dependent automation services, tailored for privacy-conscious users and developers alike.

In essence, the AI Operator is more than a tool—it's a statement. A statement that useful, personalised AI can exist without compromising on security or privacy. Through this project, we aim to show that local-first, AI-powered agents are not only feasible but also preferable in a growing landscape of decentralised digital intelligence.

2. Project Plan (8-Week Sprint Plan)

Week Sprint Focus		Deliverables
1	UI Setup + Environment Configuration	Desktop chatbot prototype, GitHub repo setup
2	Web Scraping Module	Scraper to read content from active browser tabs
3	LLM Integration	Basic prompt-response loop with parsed HTML
4	Semantic Kernel Integration	Convert intent to structured actions
5	Browser Automation	Selenium: Clicks, inputs, navigation
6	Agent Pipeline Assembly	Full pipeline: Chatbot > LLM > Action > Feedback
7	Testing & Edge Case Handling	Local execution tests + security validation
8	Final Refinements + Demo Prep	MVP demo, documentation, report, code freeze

3. Team Roles

- **Sudhanshu Ghuge – LLM Prompt Engineer and Backend Developer**
Designs effective prompts and integrates the LLM with the semantic kernel.
- **Deepak Shelke – Full stack Developer**
Builds scraping and automation features using Selenium and APIs.
- **Ayush Poojari – Frontend and AI Developer**
Frontend Developer and ML Engineer skilled in building responsive UIs, integrating machine learning models, and developing LLM systems.
- **Tarun Kumar – Scrum Master and Backend Developer**
Leads sprint planning and oversees backend architecture and integration.
- **Soham Deo – AI Engineer and Developer**
Builds the chatbot UI and connects it to the backend with logging support.
- **Preet Raut – Cloud Engineer and Risk Analyst**
Manages cloud data flow and structures LLM outputs with risk analysis.

4. Architecture Setup

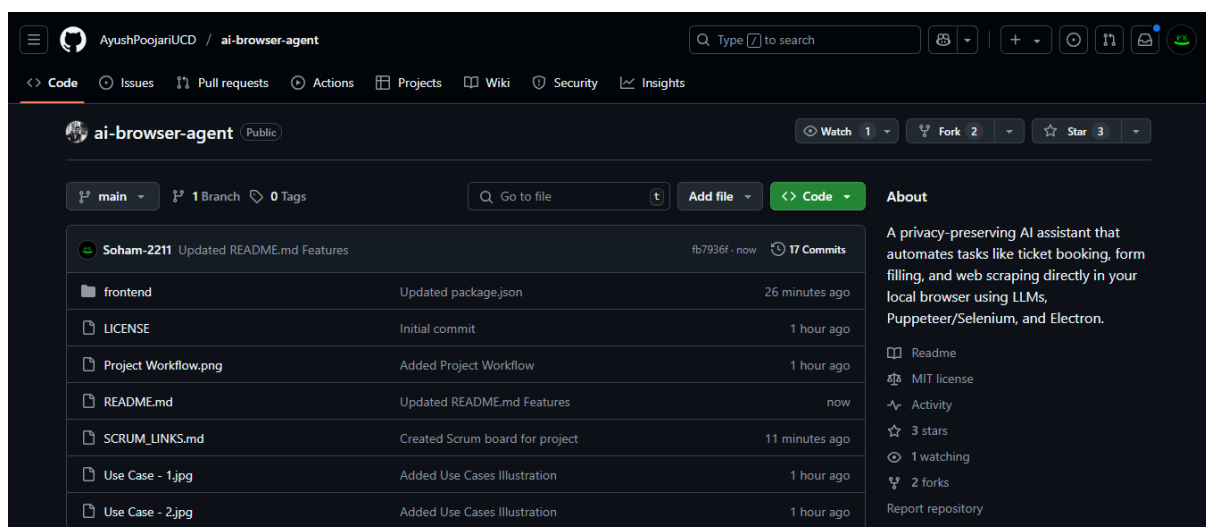
- **Cloud Platform:** Azure
- **Desktop App:** Built with Electron/React + Python backend
- **LLM:** Anthropic Claude 3.5 and Azure Cloud
- **Automation:** Selenium
- **Semantic Kernel:** MS Semantic Kernel for step breakdown

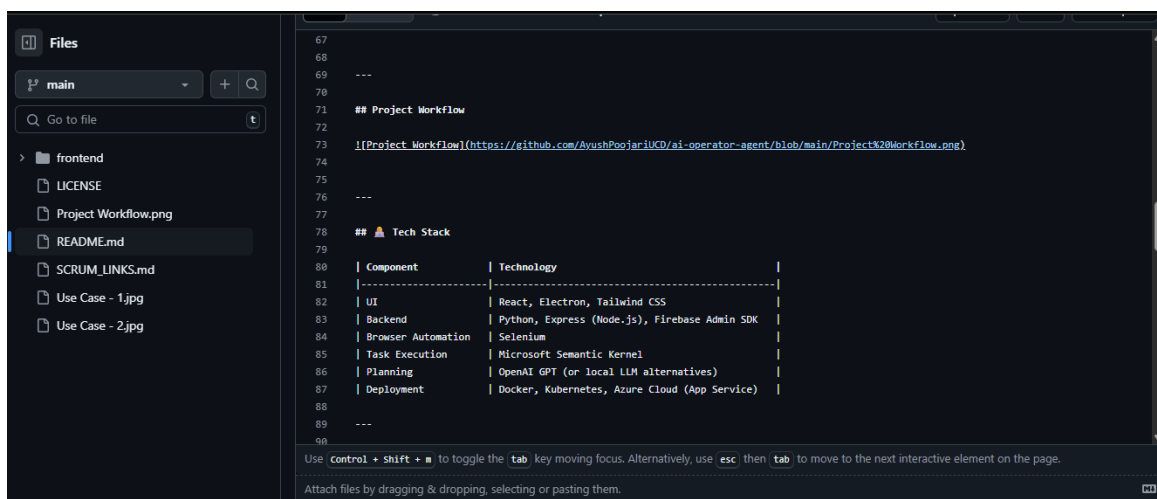
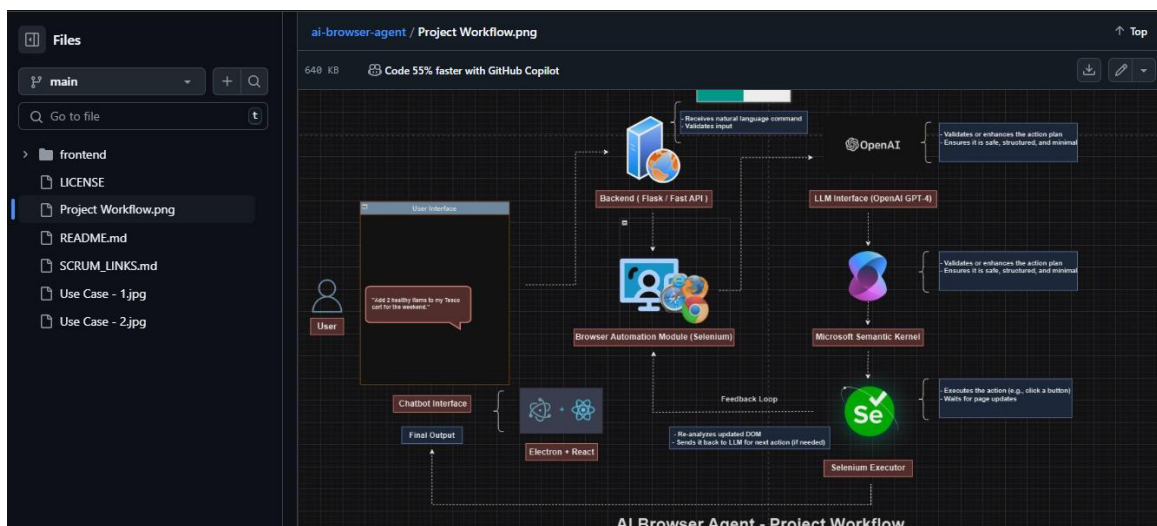
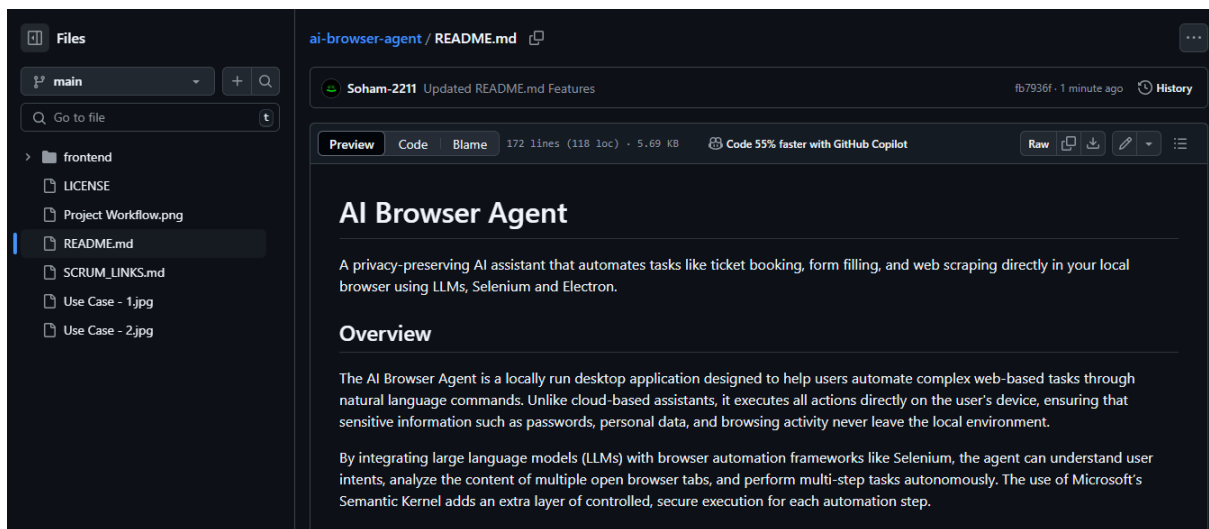
5. Data Plan

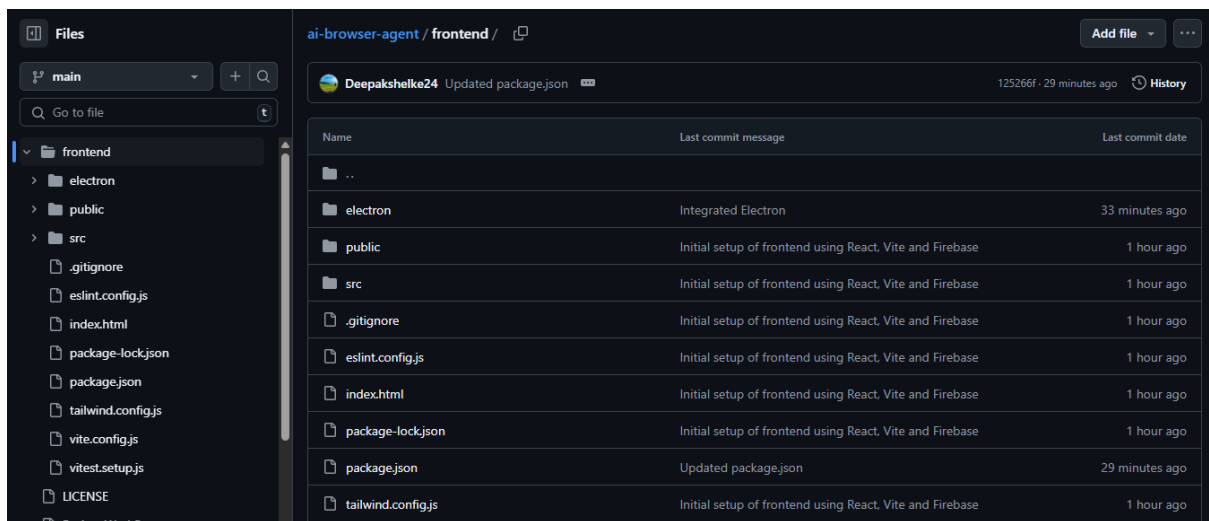
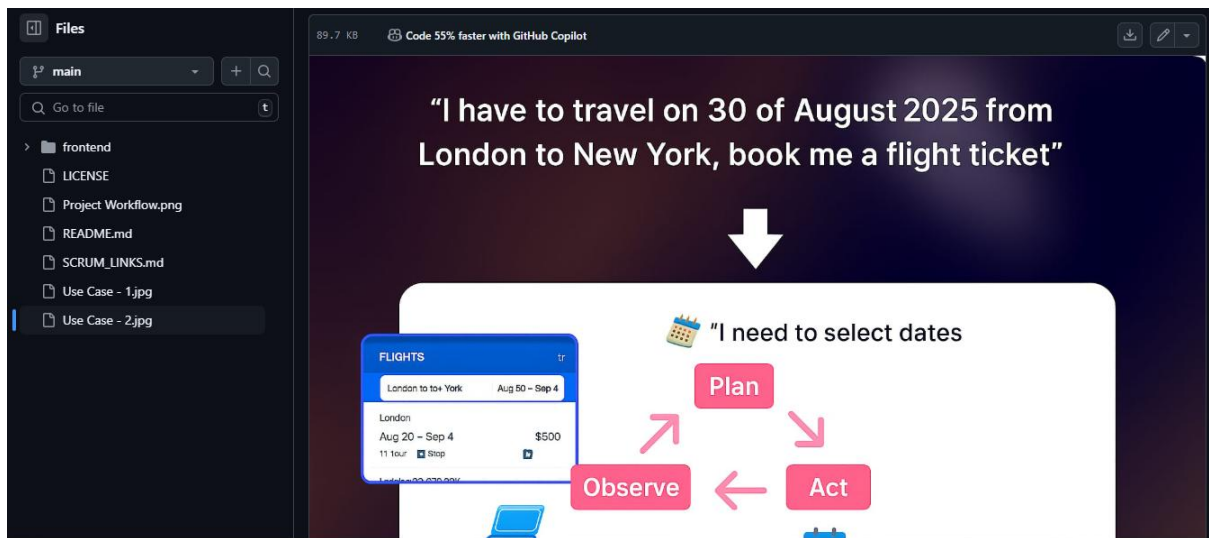
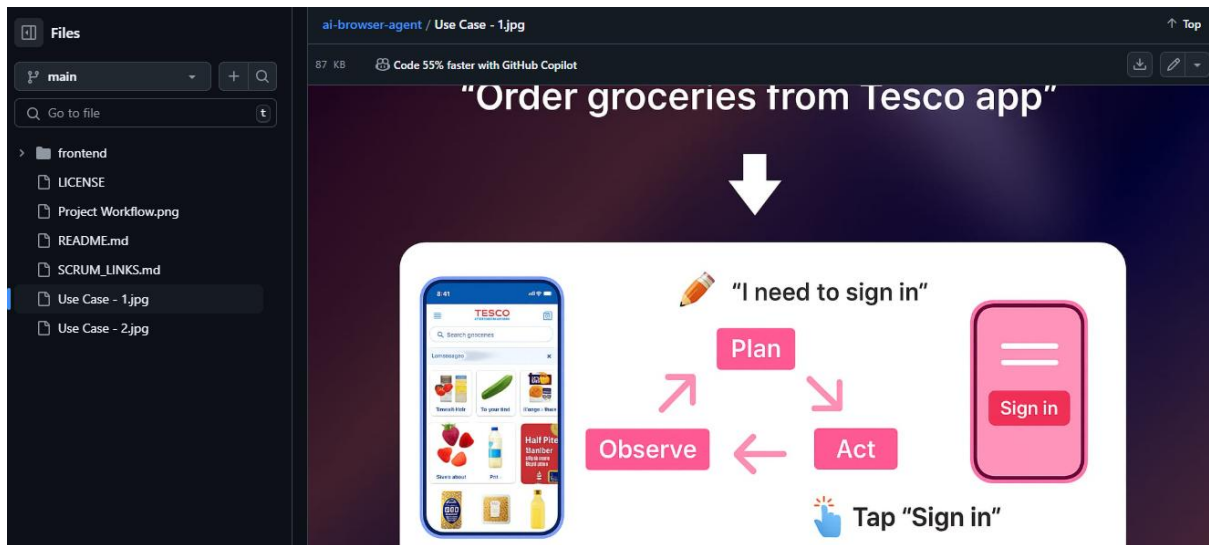
- **Input Data:** Live content from browser tabs (HTML/DOM)
- **Generated Data:** Structured web actions from LLM, logs of agent decisions
- **Evaluation Data:** User test cases (e.g., shopping task), success/failure logs, task completion time

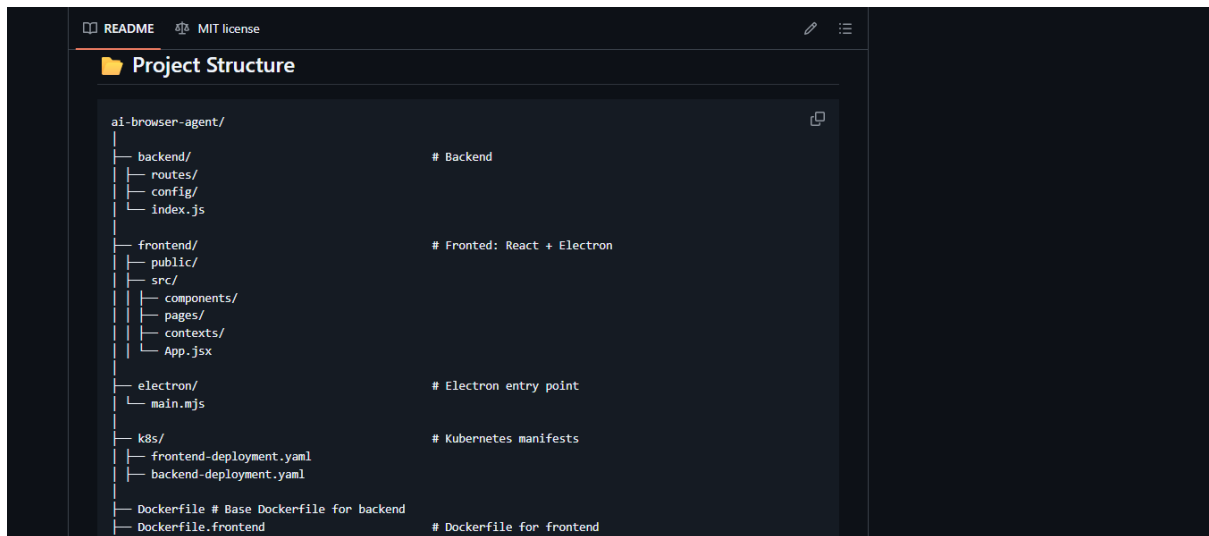
6. GitHub Evidence

- All team members contribute via Git branching and pull requests
- Repo Link: [Repository Link](#)
- Repo includes:
 - /code: UI, backend, kernel, LLM, testing modules
 - /sprints: weekly goals, completed tasks
 - GitHub Insights shows individual commit history (non-trivial)









7. Team Management

- **Tool Used:** Jira (for sprint planning & task assignment), GitHub Projects
- **Meeting Format:** Twice weekly (With Microsoft Coordinators on MS Teams on tuesday)
- **Minutes Structure:**
 - Attendance
 - Sprint progress
 - Bottlenecks
 - Assigned tasks with deadlines

8. Architecture Diagram and Tech Stack

