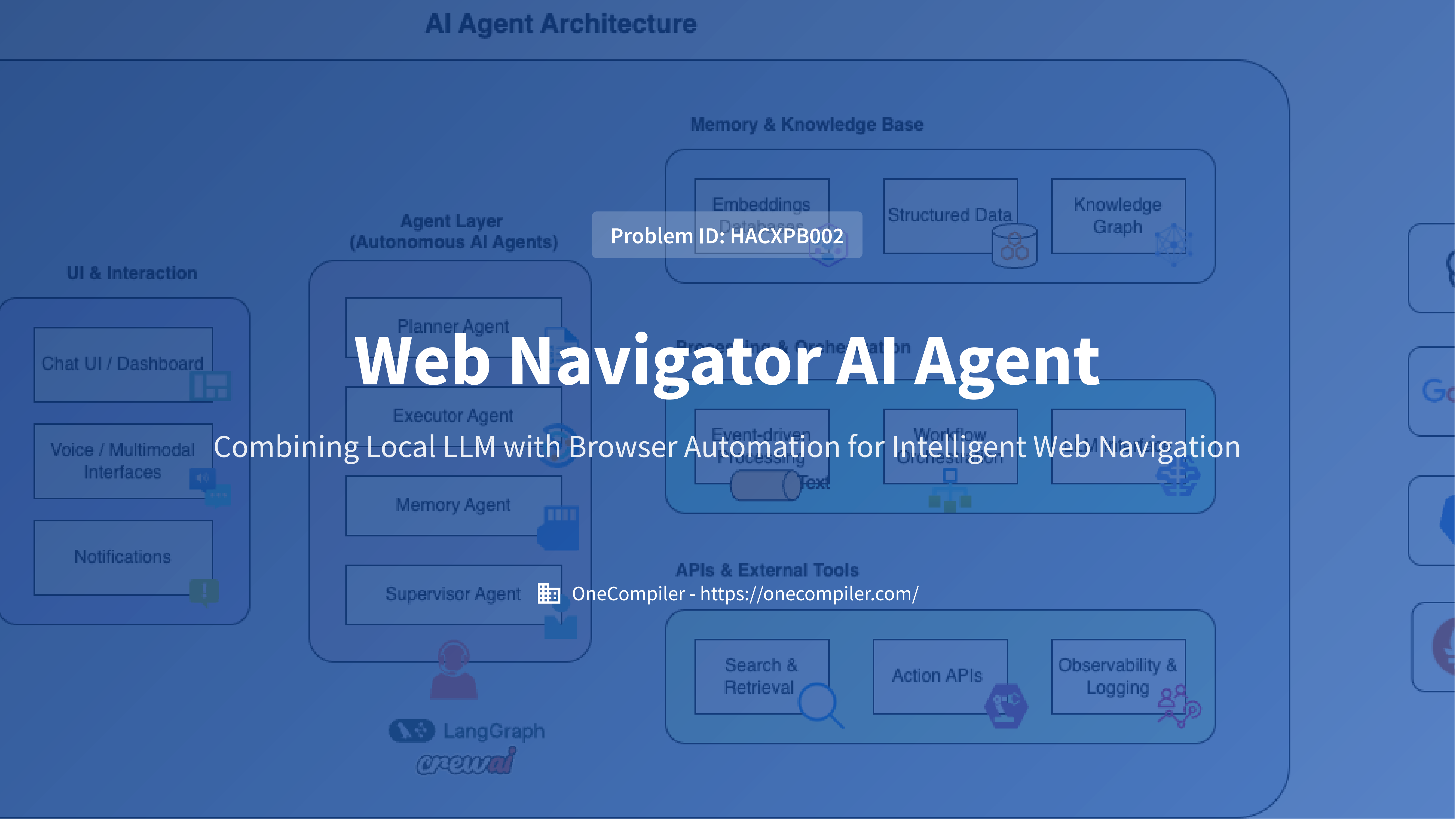


Problem ID: HACXPB002

Web Navigator AI Agent

Combining Local LLM with Browser Automation for Intelligent Web Navigation

OneCompiler - https://onecompiler.com/

Problem Statement

Building an Intelligent Web Navigation System

Core Challenge

Create an AI agent that combines **local LLM** with **browser automation** for autonomous web navigation

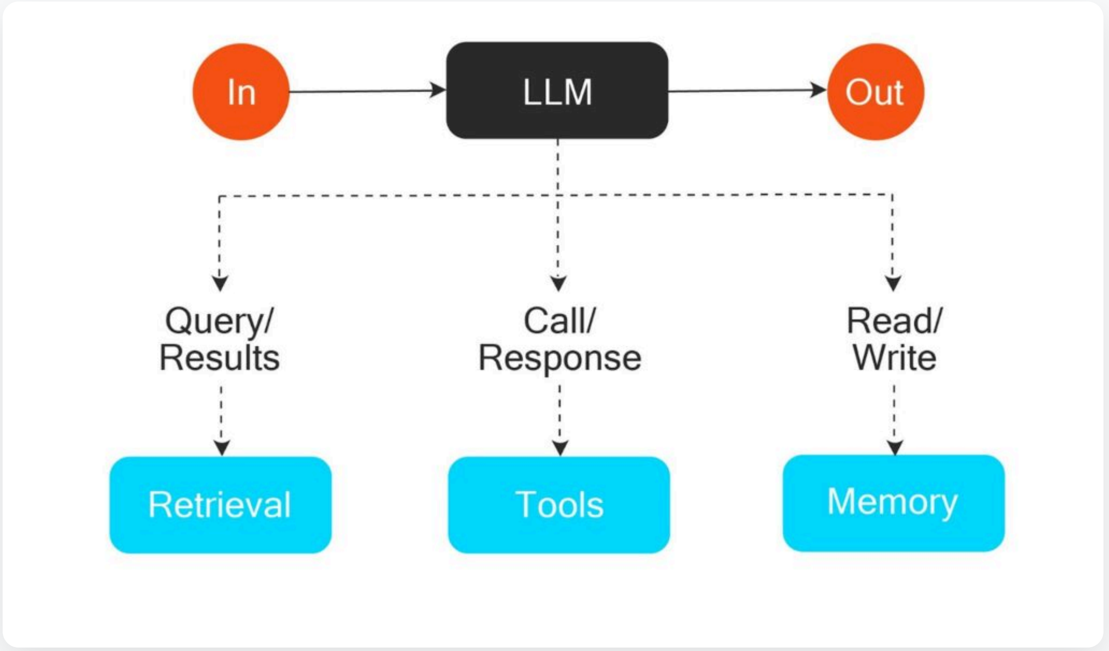
Key Requirements

- Process **natural language instructions**
- Navigate web pages **autonomously**
- Extract relevant information
- Return **structured output**

Technical Context

Problem ID: **HACXPB002**

Company: **OneCompiler**



Objectives

What the Web Navigator AI Agent Needs to Accomplish

Natural Language Processing

Interpret **user instructions** in natural language and translate them into actionable web navigation tasks

Autonomous Web Navigation

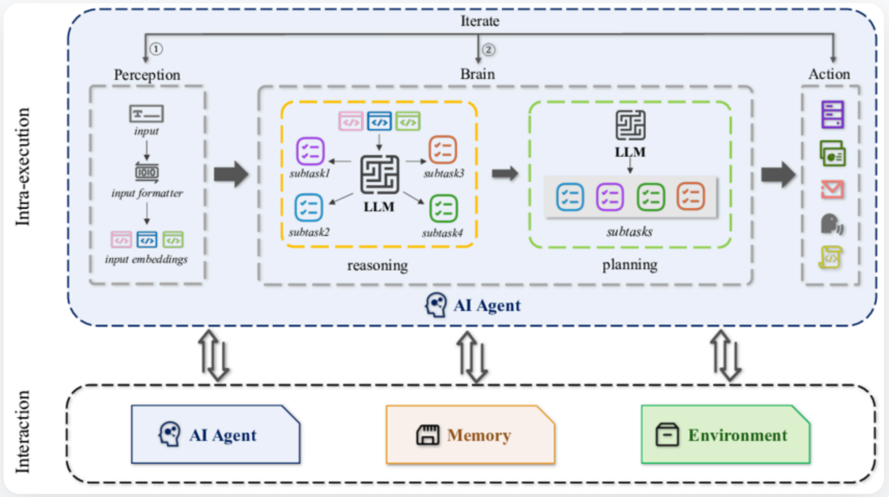
Navigate websites **independently** without human intervention, handling dynamic content and page changes

Data Extraction & Structuring

Extract relevant information from web pages and return it in a **structured format** for easy consumption

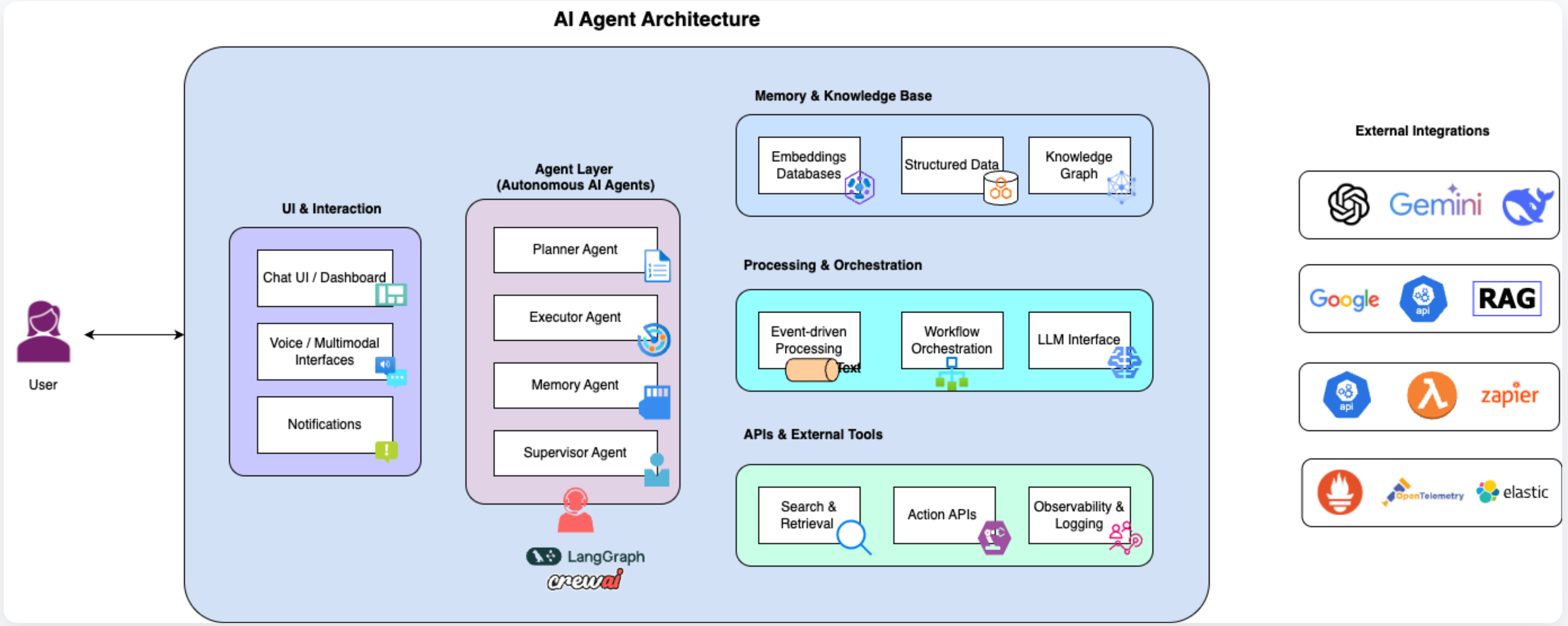
Technology Integration

Seamlessly combine **local LLM** capabilities with **browser automation** tools for optimal performance



Architecture Overview

High-Level System Design



User Interface

Natural language input and structured output display

LLM Processing

Local LLM for instruction interpretation and decision making

Browser Automation

Playwright/Selenium for web navigation and interaction

Data Processing

Extraction and structuring of web content

Technical Components

Technology Stack Breakdown

Orchestration Layer

- **Python**- Extensive libraries & AI ecosystem
- **Node.js**- JavaScript runtime for web tasks

Manages workflow between LLM and browser automation

LLM & NLP

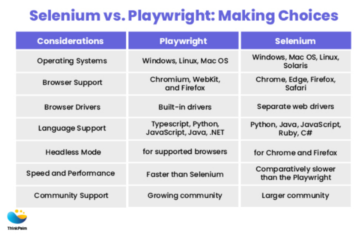
- **LangChain**- LLM application framework
- **Ollama**- Local LLM management

Processes natural language instructions and makes decisions

Browser Automation

- **Playwright**- Modern, reliable automation
- **Selenium**- Established web automation
- **Puppeteer**- Chrome DevTools Protocol

Controls browser actions and extracts web content



Selenium vs. Playwright: Making Choices

Considerations	Playwright	Selenium
Operating Systems	Windows, Linux, Mac OS	Windows, Mac OS, Linux, Solaris
Browser Support	Chromium, WebKit, and Firefox	Chrome, Edge, Firefox, Safari
Browser Drivers	Built-in drivers	Separate web drivers
Language Support	Typescript, Python, JavaScript, Java, .NET	Python, Java, JavaScript, Ruby, C#
Headless Mode	for supported browsers	for Chrome and Firefox
Speed and Performance	Faster than Selenium	Comparatively slower than the Playwright
Community Support	Growing community	Larger community

ThinkPalm

Data Processing

- **BeautifulSoup**- HTML parsing
- **Pandas**- Data structuring
- **JSON/XML**- Output formats

Structures extracted information for user consumption

Implementation Workflow

Step-by-Step Process of the Web Navigator AI Agent

1 **Natural Language Input**

User provides **instructions** in natural language

2 **LLM Processing**

Local LLM interprets instructions and generates navigation plan

3 **Browser Automation**

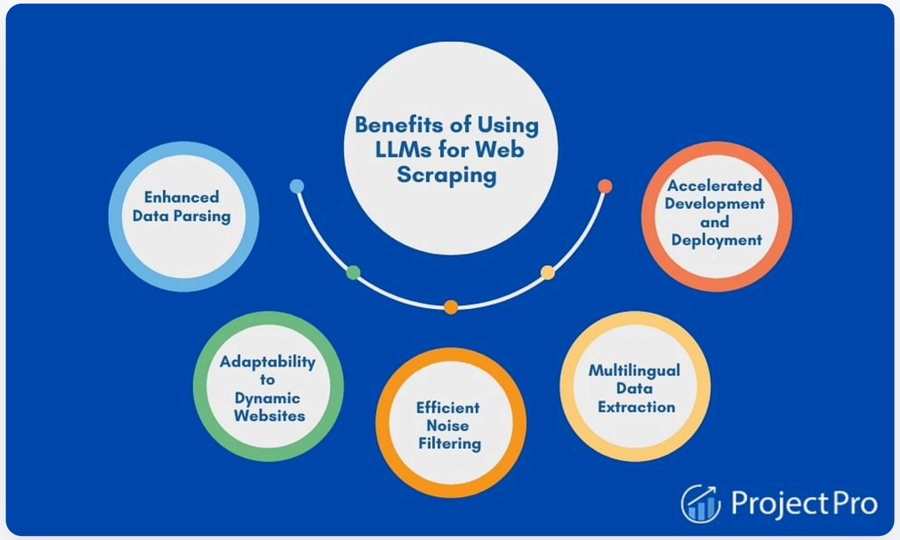
Playwright/Selenium executes web navigation actions

4 **Data Extraction**

System **identifies** and extracts relevant information

5 **Structured Output**

Results formatted and returned in **structured format**


Benefits of Using LLMs for Web Scraping
Enhanced Data Parsing
Accelerated Development and Deployment
Adaptability to Dynamic Websites
Efficient Noise Filtering
Multilingual Data Extraction
ProjectPro

Benefits & Applications

Use Cases and Advantages of Web Navigator AI Agent

Key Benefits

Efficiency

- Reduces **manual effort** in web navigation
- Processes tasks **faster** than human interaction
- Operates **24/7** without fatigue

Privacy

- **Local LLM** keeps data on device
- No **third-party** data exposure
- Complete control over **data handling**

Accessibility

- **Natural language** interface
- No technical knowledge required
- Democratizes **web automation**

Applications

E-commerce

- **Price monitoring** across sites
- Product **comparison**
- Automated **purchasing** workflows

Research

- **Data collection** for analysis
- Academic **literature review**
- Competitive **intelligence** gathering

Business

- **Lead generation**
- Market **trend analysis**
- Content **aggregation**

Conclusion & Next Steps

Summary and Future Directions

Key Takeaways

- Web Navigator AI Agent combines **local LLM** with **browser automation**
- Enables **autonomous web navigation** through natural language
- Technology stack: Python/Node.js, LangChain/Ollama, Playwright/Selenium
- Provides **structured output** from web content extraction
- Benefits include efficiency, privacy, and accessibility

This technology will transform how we interact with and extract information from the web

Next Steps

1. **Prototype Development**
   Create initial **working prototype** with core functionality
2. **Testing & Refinement**
   Evaluate with **real-world scenarios** and improve performance
3. **Advanced Features**
   Add **multi-site navigation** and complex task handling
4. **Integration & Deployment**
   Connect with **OneCompiler platform** and release to users