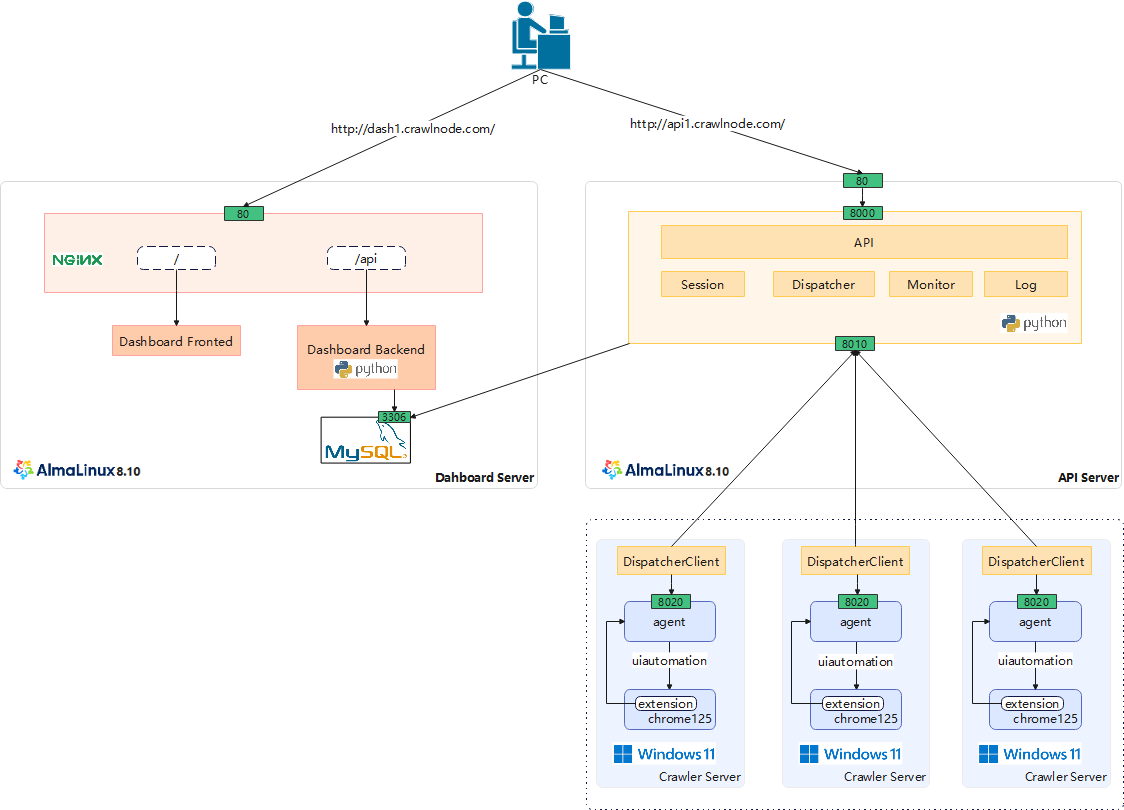
# Project Goals

Develop a web automation system that uses anti-anti-crawler technology to simulate user behavior and perform web browsing and interaction.

# System Design

## Architecture diagram



## Subsystem Description

### Crawler Server

Crawler Server is the host that performs crawler tasks, with Windows 11 operating system, Chrome browser (version 125), and agent software (to be developed). It supports running multiple isolated browser instances at the same time .

Agent is a Python-developed application. It opens port 8020 and accepts API requests over HTTP.

The main functions of Agent include:

* Host status monitoring
* Open Chrome (supports proxy and configuration isolation)
* Chrome operations: navigate to a url, click on element, type keystrokes into element, get http content
* Close Chrome

Development Environment:

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **Technology** | **Description** |
| Environment | Operating System | Windows 11 |  |
| Browser | Chrome 125 |  |
| Agent | Language | Python 3.8 |  |
| Framework | FastAPI |  |
| IDE | VS Code 1.100 |  |

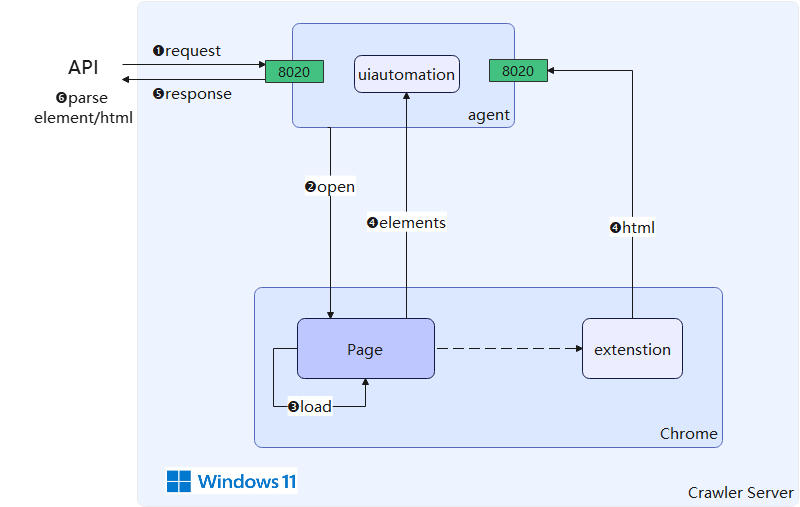
Runtime Environment:

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **Technology** | **Description** |
| Environment | Operating System | Windows 11 |  |
| Browser | Chrome 125 |  |
| Agent | Runtime | Python 3.8 |  |

Crawler Server can support the following interfaces:

* + Query host status
  + Query session list
  + For other interfaces, refer to [4](#_接口设计) API Server [Interface Design](#_接口设计)

API request/response data flow diagram：



### API Server

Provides APIs for crawling website over HTTP, and provides monitoring and logging functions, as follows:

* Monitoring: Online/offline status of Crawler Server
* Session Management
* Viewing logs

Development Environment:

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **Technology** | **Description** |
| Environment | Operating System | Windows 11 |  |
| Browser | Chrome 125 |  |
| API Server | Language | Python 3.8 |  |
| Framework | FastAPI |  |
| IDE | VS Code 1.100 |  |

Runtime Environment:

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **Technology** | **Description** |
| Environment | Operating System | AlmaLinux 8.10 |  |
| API Server | Runtime | Python 3.8 |  |

### Dashboard

Dashboard provides management functions for the Crawler, Session and Log:

* Crawler management: support load/refresh table、modify alias
* Session management: support filtering query by crawler server, with paginated results display
* Log management: support filtering query by crawler server, with paginated results display

Development Environment:

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **Technology Stack** | **Description** |
| Environment | Operating System | Windows 11 |  |
| Browser | Chrome 125 |  |
| frontend | Language | Vue |  |
| UI Framework | Ant Design |  |
| IDE | VS Code 1.100 |  |
| backend | Language | Python 3.8 |  |
| Framework | FastAPI |  |
| IDE | VS Code 1.100 |  |

Runtime Environment:

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **Technology Stack** | **Description** |
| Environment | Operating System | AlmaLinux 8.10 |  |
| Web Server | Nginx 1.20 |  |
| Dashboard | Runtime | Python 3.8 |  |

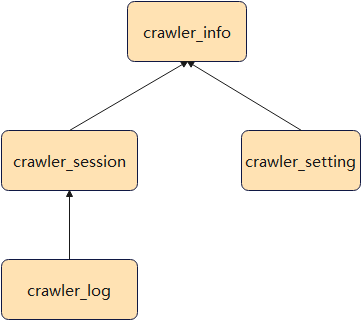
### Data storage

The systems share data storage:

|  |  |  |
| --- | --- | --- |
|  | **Technology** | **Description** |
| Database | MySQL 8.0 |  |

# Database Design

## Data Model



## Data Structure

### Common Fields

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description​** |
| id | int | Primary key |
| create\_time | datetime |  |
| update\_time | datetime |  |

### crawler\_info

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description​** |
| uuid | varchar |  |
| host\_name​​​ | varchar |  |
| internal\_ip | varchar |  |
| external\_ip | varchar |  |
| os | varchar |  |
| agent | varchar |  |
| last\_heartbeat | datetime |  |
| status | int | 10:online, 20:offline, 30:shutdown |
| cpu\_usage | float |  |
| memory\_usage | float |  |

### crawler\_setting

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description​** |
| crawler\_id | int | Foreign key |
| alias | varchar |  |
| max\_browser\_count | int |  |

### crawler\_session

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description​** |
| crawler\_id | int | Foreign key |
| uuid | varchar |  |
| init\_time | datetime |  |
| url | varchar |  |
| destory\_time | datetime |  |

### crawler\_log

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description​** |
| crawler\_session\_id | int | Foreign key |
| url | varchar |  |
| request\_time | datetime |  |
| response\_time | datetime |  |
| status\_code | int |  |

# API Server interface design

The API server provides HTTP interfaces that use the POST method with JSON-formatted request and response bodies.

## Navigate to a URL

* URL: /go
* Method: Post
* Header:
* session: session id (default: start new browser session)
* RequestBody:
* url: destination URL
* timeout: the maximum number of seconds to wait (default: 60 seconds)
* user: the HTTP basic authentication username (optional)
* pass: the HTTP basic authentication password (optional)
* ResponseBody:
* session : Session ID
* url: Main url requested
* status\_code: Main page HTTP status code
* response\_headers: List of response headers
* html: Raw HTML content for page.
* elements: Associative array of interactive elements on page:
  + element\_id: Element ID
  + name: Element Name
  + class: Element Class
  + value: Element Value
* responses : List of all response URLs for sub-URLs requested (eg, images, AJAX, etc.)
  + response\_id
  + url
  + content\_type
  + status\_code
  + size

## Click on element

* URL: /click
* Method: Post
* Header:
* session: session id (default: start new browser session)
* RequestBody:
* element: element selector
* wait: 10s / page\_load / none
* timeout: the maximum number of seconds to wait (default: 60 seconds)
* ResponseBody:

## Type keystrokes into element

* URL: /type
* Method: Post
* Header:
* session: session id (default: start new browser session)
* RequestBody:
* element: element selector (or none)
* keystrokes: sequence of [keystrokes](https://www.autohotkey.com/docs/v1/KeyList.htm) to type (eg, david@vinaudit.com{TAB}12346{ENTER} )
* wait: 10s / page\_load / none
* timeout: the maximum number of seconds to wait (default: 60 seconds)
* ResponseBody:

## Download response content

* URL: /download
* Method: Post
* Header:
* session: session id (default: start new browser session)
* RequestBody:
* session: session id
* response: response id
* ResponseBody:

## Destory session

* URL: /destory
* Method: Post
* Header:
* session: session id
* ResponseBody:

# Dashboard Page

## Crawler List Page

The Crawler List Page shows a paginated table of the crawler servers

* Query: No filtering/search functionality (default view).
* Refresh button: Click to reload the page and update the crawler list
* Table: Host\_name, alias, last heartbeat time, status
* operate:
* Edit button: Modifies the alias of the selected crawler server
* Session link: Click to navigate to the session list page, automatically filtering sessions by the selected crawler
* Log link: Click to navigate to the Log List Page, automatically filtering sessions by the selected crawler

## Session List Page

The Session List Page shows a paginated table of the sessions.

* Query: filtering/search by crawler server
* Table: crawler server, session\_id, url
* operate: no interactive actions available

## Log List Page

The Log List Page shows a paginated table of the logs.

* Query: filtering/search by crawler server
* Table: crawler server, api, request\_time, response\_time, status\_code
* operate: no interactive actions available

# Test cases

* Test script for requesting top 100 domains: https://radar.cloudflare.com/domains
* Test script for requesting 60,000 known dealer websites (to detect which websites are active)
* Test script for crawling all known links on given dealer websites:
  + thruwaynissan.com, motorwerkshonda.com, capitalfordrockymount.com, cityautotrucks.com, kirklandhonda.com, 1autoliquidators.com, 0to60motorsportsct.com, 039autosale.com, 1autodmv.com, 1oakautos.com, 1ownerautosales.net, my1stopauto.com, 1800autoapproved.com, 1easyrideautosale.com, 1uniquemotorsllc.com

# Development Plan

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | W1 | W2 | W3 | W4 | W5 | W6 | W7 | W8 | W9 |
| Requirements Analysis |  |  |  |  |  |  |  |  |  |
| System Design |  |  |  |  |  |  |  |  |  |
| Crawler Server Development |  |  |  |  |  |  |  |  |  |
| API+Dashboard Development |  |  |  |  |  |  |  |  |  |
| Test Cases |  |  |  |  |  |  |  |  |  |