

# Privacy Policy Assistant Instruction

This project is a browser extension designed to help users better understand website privacy policies. Below is a step-by-step guide to setting up and using the plugin:

## Environment Setup

Install Docker and a virtual machine (e.g., WSL + Ubuntu, or any other Linux VM)

### MongoDB Local Setup

1. Install MongoDB (we recommend [MongoDB Compass](#) as a graphical interface):

Go to the MongoDB download page: <https://www.mongodb.com/try/download/community>

2. Create a Database and collection

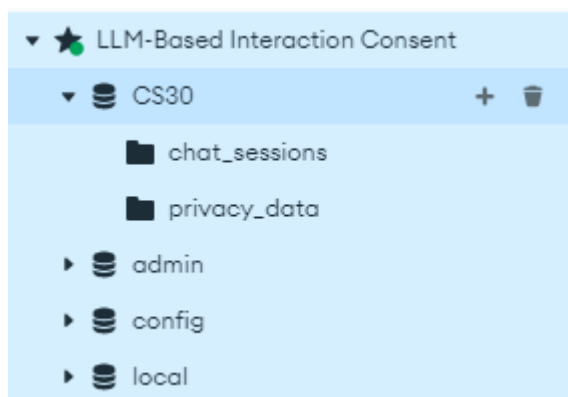
Our project uses a MongoDB container configured in `docker-compose.yml`.

Before opening MongoDB Compass, please ensure Docker is running by input command:  
docker-compose up --build -d (only input this command first time) If you want to know more about docker command please check the “README” under “flaskproject”.

Once Docker is running, it will start a mongodb container (exposing localhost:27017), create a volume to persist database data, and allow external tools (like Compass) to connect to container.

3. View the database in MongoDB Compass

Open Compass and click “connect” to show the collection.



4. Match Local Settings with the Python Code

The Python application reads MongoDB connection parameters from environment variables (or uses defaults):

```

host = os.getenv('MONGODB_HOST', 'localhost')
port = 27017
database_name = os.getenv('MONGODB_DB', 'CS30')











connection_string = f'mongodb://{host}:{port}/'

```

Make sure MongoDB Docker container is running and exposing port 27017, and no local MongoDB instance is occupying the same port.

## Step 1: Start the Main Project (FlaskProject)

Open the main project directory: FlaskProject

 .git	2025/5/30 17:17	文件夹
 Backend Development	2025/5/25 18:04	文件夹
 Cloud	2025/5/25 17:44	文件夹
 Data	2025/5/25 17:44	文件夹
 <b>FlaskProject</b>	2025/5/30 16:40	文件夹
 FlaskProject_crawler	2025/5/25 17:44	文件夹
 FlaskProject_MQ	2025/5/25 17:44	文件夹
 Frontend	2025/5/30 16:40	文件夹
 Frontend_Archived	2025/5/25 17:44	文件夹
 literature	2025/5/25 17:44	文件夹

Ensure that Docker is installed and running

```

PS D:\cs30-1\code\capstone\FlaskProject> wsl -d Ubuntu
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 5.15.153.1-microsoft-standard-WSL2 x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Fri May 30 16:47:36 AEST 2025
Usage of /: 1.0% of 1006.85GB Users logged in: 0
Memory usage: 7% IPv4 address for eth0: 172.28.149.57
Usage of /: 1.0% of 1006.85GB Users logged in: 0
Memory usage: 7% IPv4 address for eth0: 172.28.149.57
Usage of /: 1.0% of 1006.85GB Users logged in: 0
Memory usage: 7% IPv4 address for eth0: 172.28.149.57
Usage of /: 1.0% of 1006.85GB Users logged in: 0
Memory usage: 7% IPv4 address for eth0: 172.28.149.57

```

Open a terminal and run the following command:

Docker-compose up -d

```
root@LAPTOP-04FLTDDA:/mnt/d/cs30-1/code/capstone/FlaskProject# docker-compose up -d
[+] Running 2/2
  ✓ Container privacy-policy-mongodb   Running
0.0s
[+] Running 2/2
```

This will launch the core backend services in Docker containers

## Step2: Start the Login Service (login module)

Open another terminal window or code editor

Navigate to the FlaskProject/services/backend/login/ directory

Run the following command (as indicated in the README file): `uvicorn main:app --reload`



```
Problems Output Debug Console Terminal Ports
INFO: 127.0.0.1:8942 - "GET /api/get_visibility?feature=extension HTTP/1.1" 200 OK
INFO: 127.0.0.1:11262 - "OPTIONS /api/get_visibility?feature=extension HTTP/1.1" 200 OK
INFO: 127.0.0.1:11262 - "GET /api/get_visibility?feature=extension HTTP/1.1" 200 OK
PS D:\cs30-1\code\capstone\Backend Development\Login> uvicorn main:app --reload
INFO: Will watch for changes in these directories: ['D:\cs30-1\code\capstone\Backend Development\Login']
INFO: Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO: Started reloader process [23888] using StatReload
INFO: Started server process [17420]
INFO: Waiting for application startup.
o INFO: Application startup complete.
INFO: 127.0.0.1:14188 - "OPTIONS /api/get_visibility?feature=extension HTTP/1.1" 200 OK
INFO: 127.0.0.1:14188 - "GET /api/get_visibility?feature=extension HTTP/1.1" 200 OK
INFO: 127.0.0.1:14733 - "GET /api/get_visibility?feature=extension HTTP/1.1" 200 OK
```

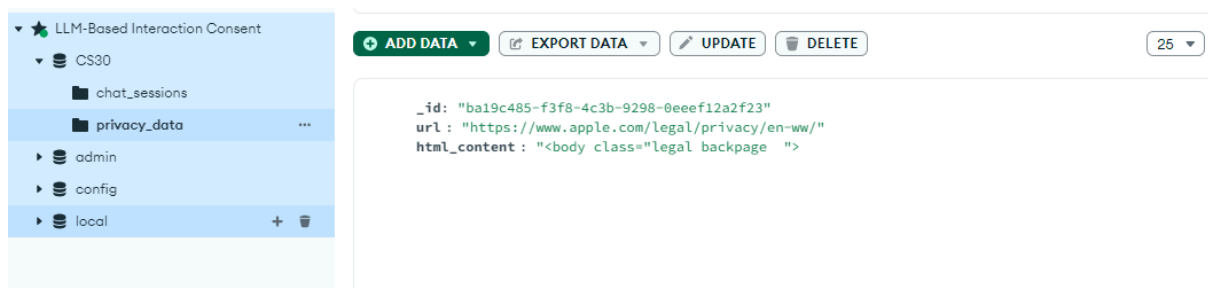
This will start the login service, enabling user authentication

## Step 3: Configure and Connect to MongoDB

Open MongoDB Compass

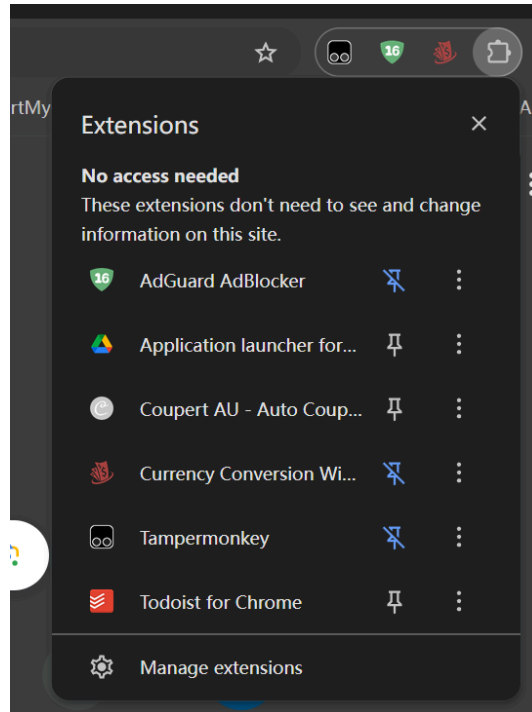
Start a local MongoDB instance (default URL: `mongodb://localhost:27017`)

Modify the MongoDB connection configuration in your project (e.g., `.env` or `config.py`) to match your local setup



## Step 4: Install and Load the Extension (Frontend)

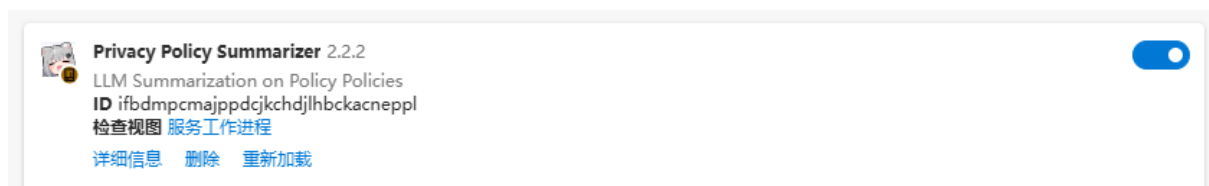
Open your browser and click the extensions icon in the upper-right corner, then select "Manage Extensions"



Make sure to turn on the “Developer Mode” and then click "Load unpacked extension"

Select the frontend-extension folder (we recommend version v2.2.2 for localhost deployment; v2.5.4 for direct use)

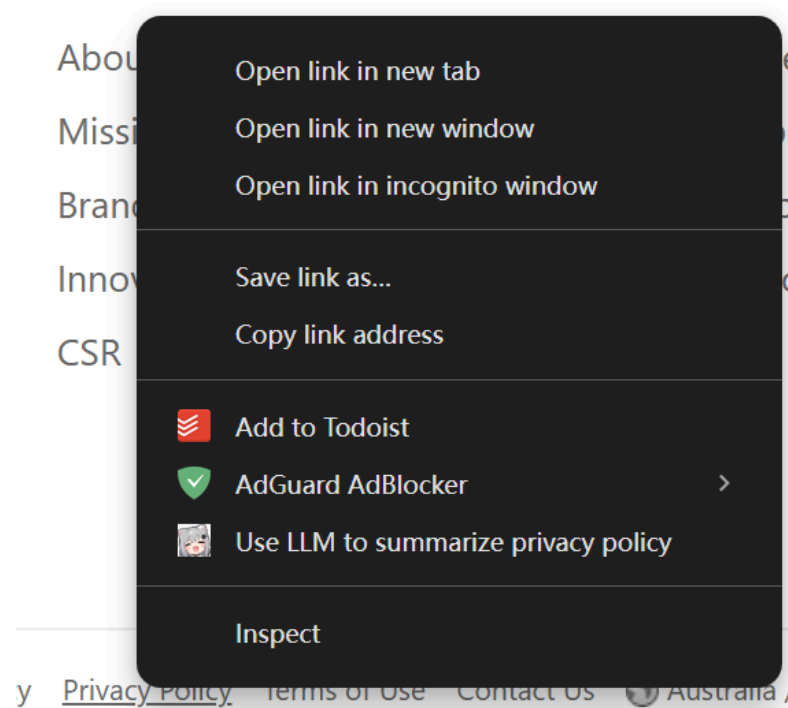
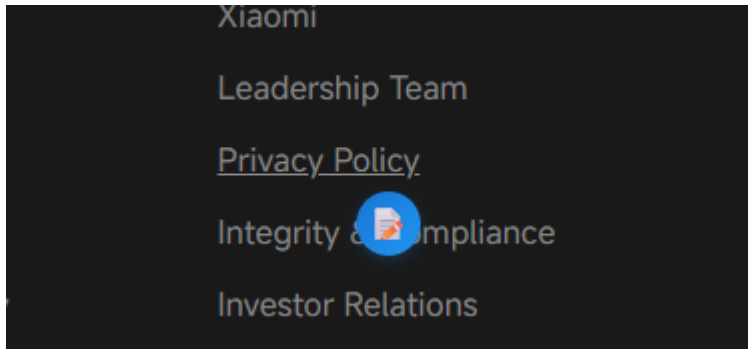
Once loaded, the plugin will be installed successfully.



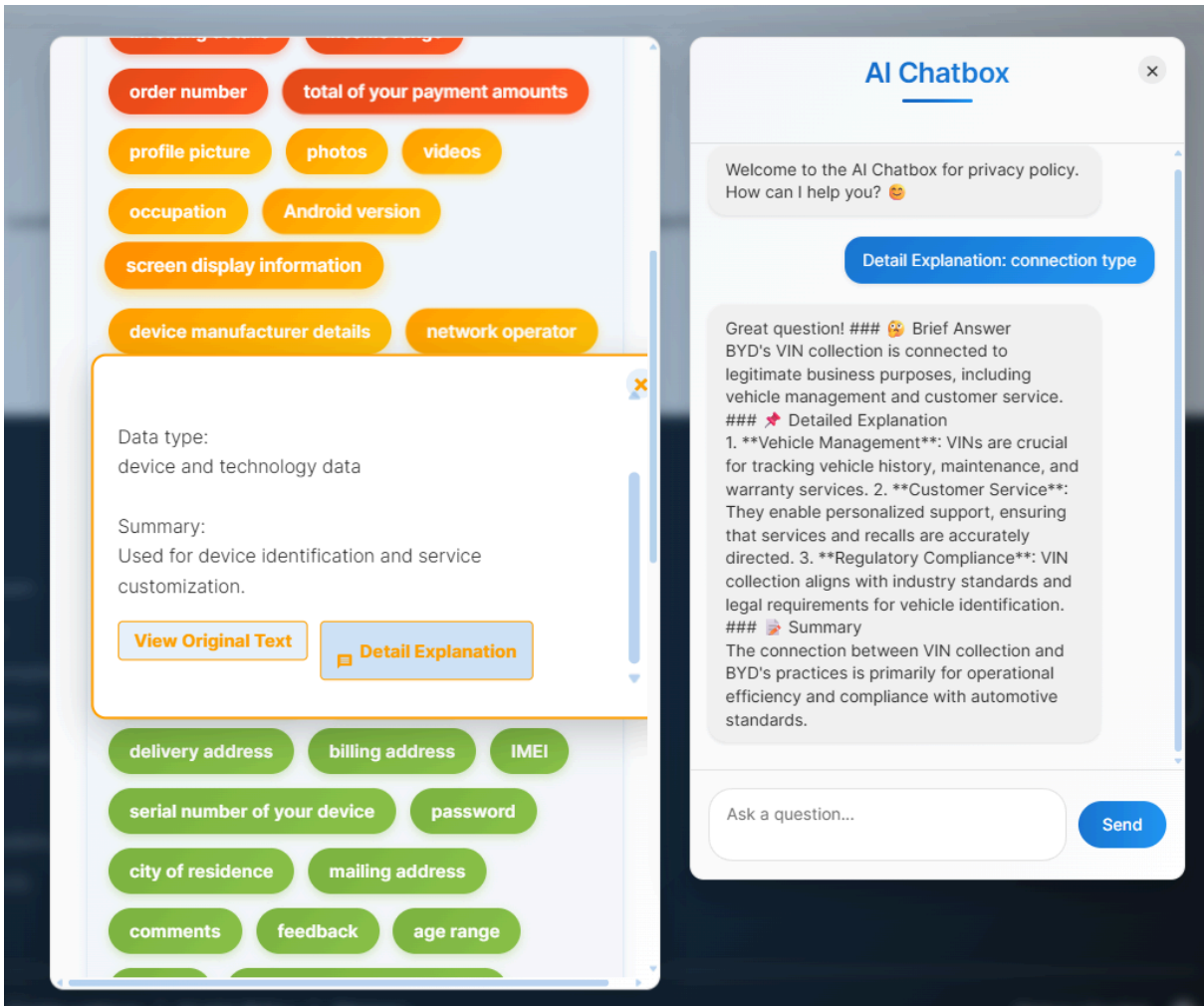
## Plugin Usage Instructions

The plugin will automatically detect privacy policy pages and display simplified summaries in pop-up bubbles. Just move the mouse hover over the link containing phases like “privacy policy”, it will show a floating icon displays “Use LLM to summarize the privacy policy”

In case if such operations don't show the floating icon, you can also right click the link to trigger the plugin.



Users can also type keywords or questions into the plugin, and it will return intelligent explanations based on LLM (Large Language Model) responses.



Usually, when clicking the floating icon, the service worker will be refreshed automatically to wait for the backend to return the summary and update the page display. If the plugin shows "service worker (Inactive)" during the loading state, click Reload and refresh the target webpage. It should resume normal operation.

