MICHIGAN STATE
UNIVERSITY

Project Plan Presentation

AI Cyberattack Early Warning System

The Capstone Experience

Team Vectra AI

Alex Fortsch
Graham Holley
Ajay Kumar
Morghane McAnelly
Alex Popovic
Jacob Sock

Department of Computer Science and Engineering
Michigan State University

Fall 2024

From Students…
…to Professionals

Project Sponsor Overview

- Cybersecurity Monitoring Company
  - Founded in 2011
- Pioneers of Generative AI
- Attack Signal Intelligence
  - Monitor attacks *WITHOUT* decryption
  - Machine learning to detect attacks and offer solutions
- Past Michigan State Capstone Sponsors
  - C2 simulator

VECTRA®

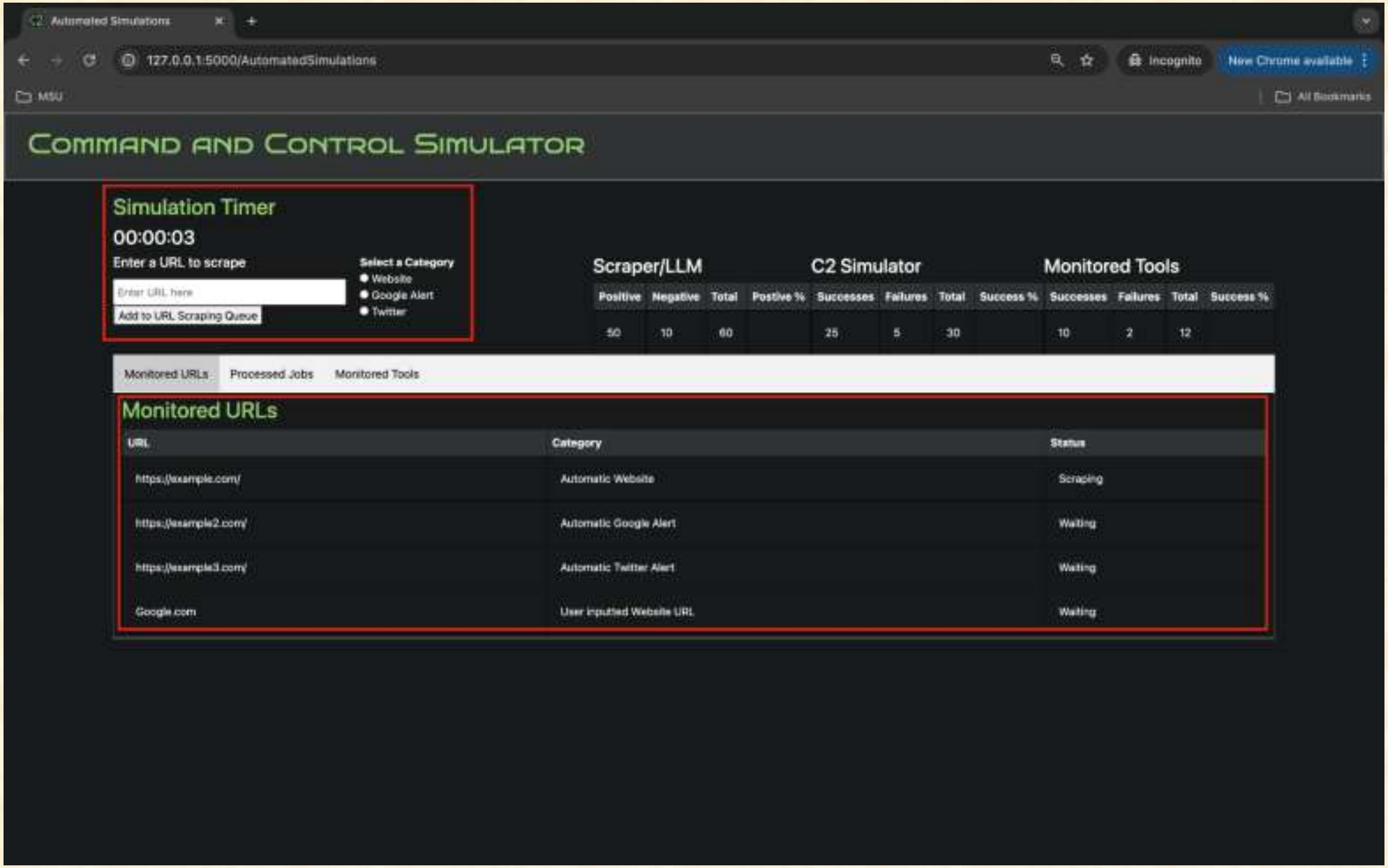
Project Functional Specifications

- Problem
  - Data scientists have to manually read reports and configure the C2 simulator
- Solution
  - Automate the process by web scraping threat intel resources, extrapolating C2 configs, and generate PCAP samples
- Result
  - Human intervention in the process is eliminated

Project Design Specifications

- Users should be able to pass in a URL in through a user interface
- Users should know the current URLs in the queue and ones already being monitored
- Users should be able to see the results of run C2 Simulator
- Users should be able to run other detection tools with valid configurations
- Users should be able to see the statistics of how the application is working as well as success rates

Screen Mockup: Opening Screen



Command and Control Simulator

Simulation Timer

00:00:03

Enter a URL to scrape

Enter URL here

Add to URL Scraping Queue

Select a Category
- ● Website
- ● Google Alert
- ● Twitter

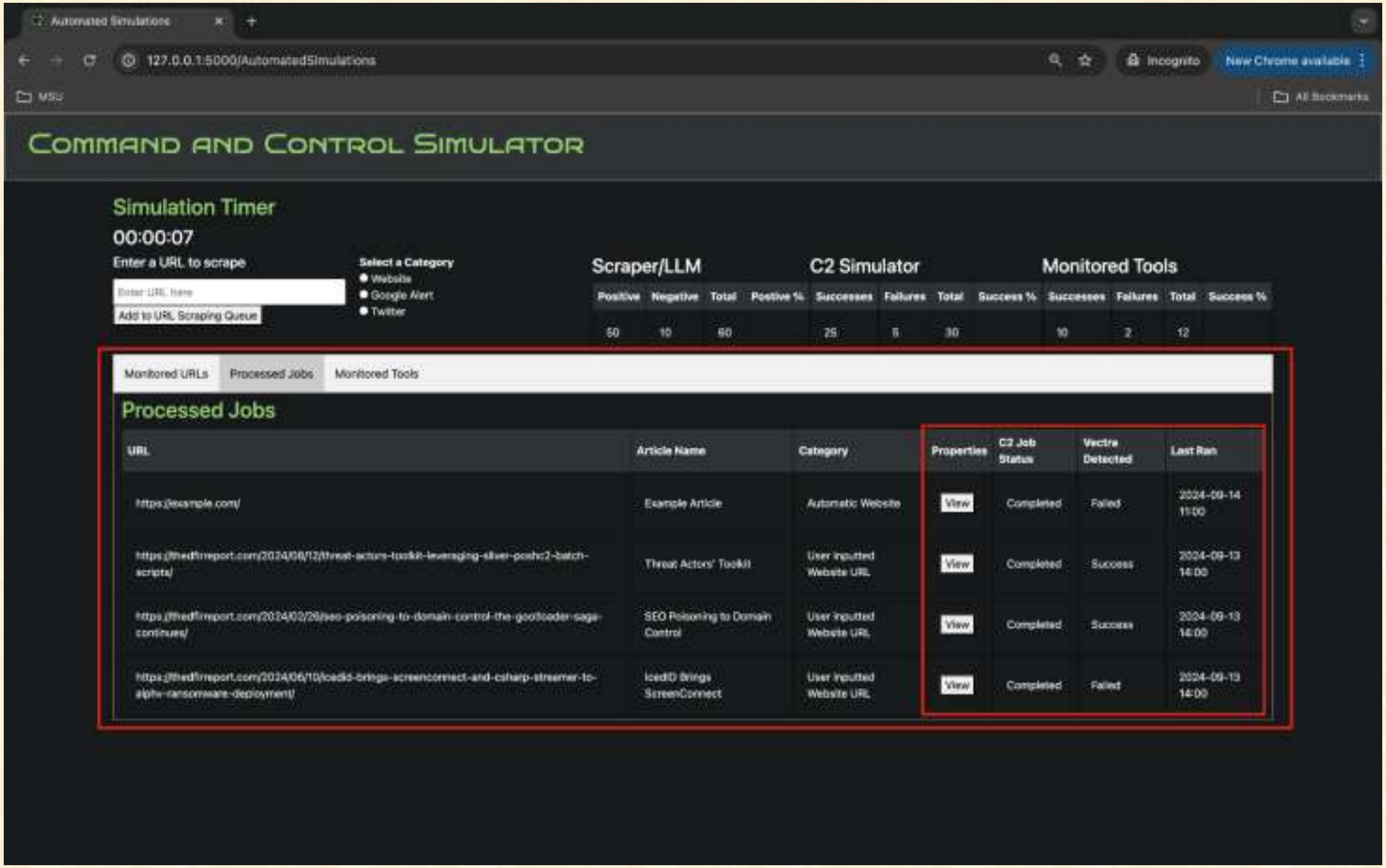
Scraper/LLM				C2 Simulator				Monitored Tools			
Positive	Negative	Total	Postive %	Successes	Failures	Total	Success %	Successes	Failures	Total	Success %
50	10	60		25	5	30		10	2	12	

Monitored URLs | Processed Jobs | Monitored Tools

Monitored URLs

URL	Category	Status
https://example.com/	Automatic Website	Scraping
https://example2.com/	Automatic Google Alert	Waiting
https://example3.com/	Automatic Twitter Alert	Waiting
Google.com	User inputted Website URL	Waiting

Screen Mockup: Processed Jobs



Command and Control Simulator

Simulation Timer

00:00:07

Enter a URL to scrape

Enter URL here

Add to URL Scraping Queue

Select a Category

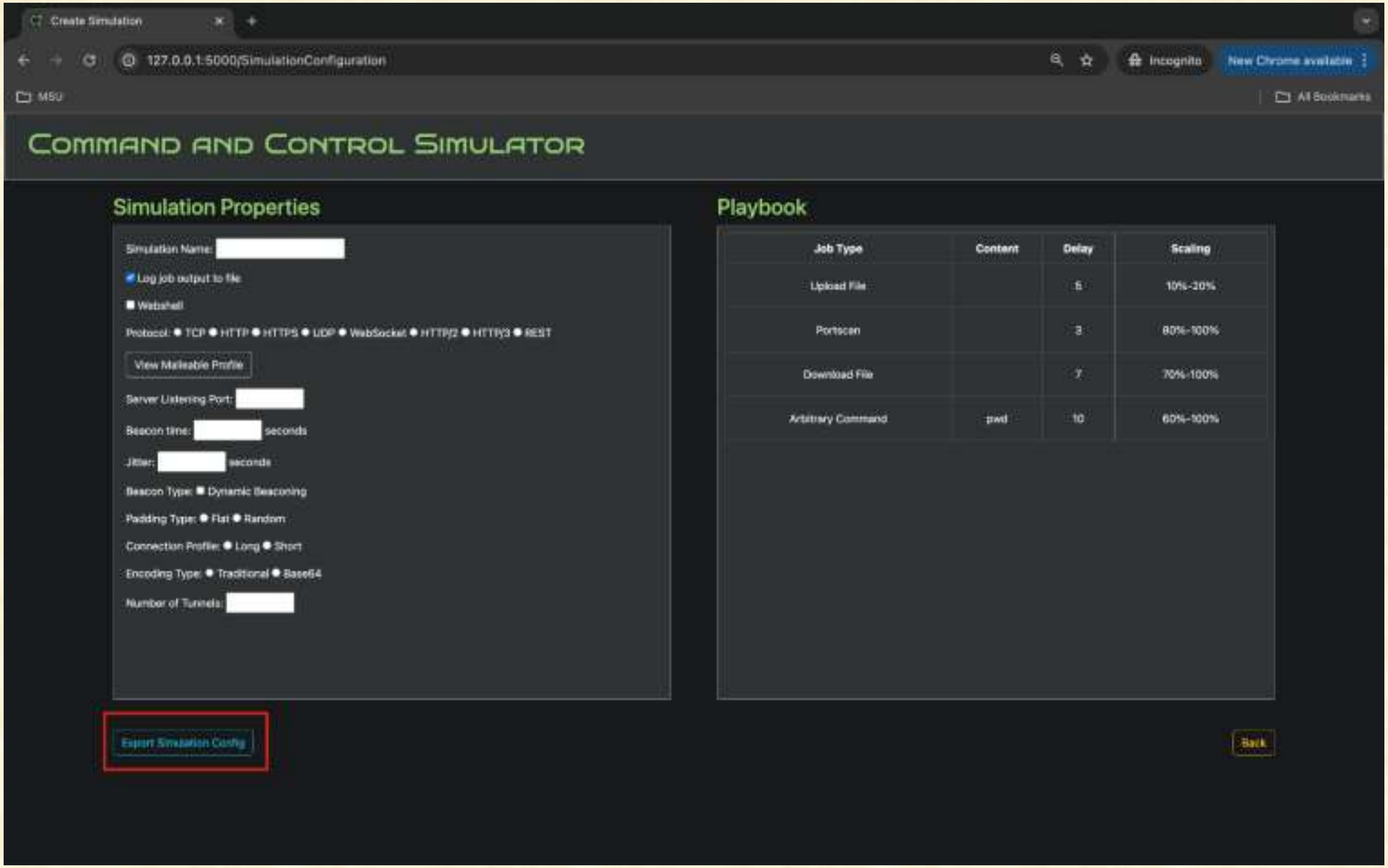
- ● Website
- ● Google Alert
- ● Twitter

Scraper/LLM				C2 Simulator				Monitored Tools			
Positive	Negative	Total	Positive %	Successes	Failures	Total	Success %	Successes	Failures	Total	Success %
50	10	60		25	5	30		10	2	12	

Processed Jobs

URL	Article Name	Category	Properties	C2 Job Status	Vectre Detected	Last Ran
https://example.com/	Example Article	Automatic Website	View	Completed	Failed	2024-09-14 11:00
https://thedfirreport.com/2024/08/12/threat-actors-toolkit-leveraging-sliver-poshc2-batch-scripts/	Threat Actors' Toolkit	User Inputted Website URL	View	Completed	Success	2024-09-13 14:00
https://thedfirreport.com/2024/02/26/seo-poisoning-to-domain-control-the-gootloader-saga-continues/	SEO Poisoning to Domain Control	User Inputted Website URL	View	Completed	Success	2024-09-13 14:00
https://thedfirreport.com/2024/06/10/icedid-brings-screenconnect-and-csharp-streamer-to-alphv-ransomware-deployment/	IcedID Brings ScreenConnect	User Inputted Website URL	View	Completed	Failed	2024-09-13 14:00

Screen Mockup: Simulation Config



Command and Control Simulator

Simulation Properties

Simulation Name:

☑ Log job output to file

☐ Webshell

Protocol: ● TCP ● HTTP ● HTTPS ● UDP ● WebSocket ● HTTP/2 ● HTTP/3 ● REST

View Malleable Profile

Server Listening Port:

Beacon time: seconds

Jitter: seconds

Beacon Type: ☐ Dynamic Beaconing

Padding Type: ● Flat ● Random

Connection Profile: ● Long ● Short

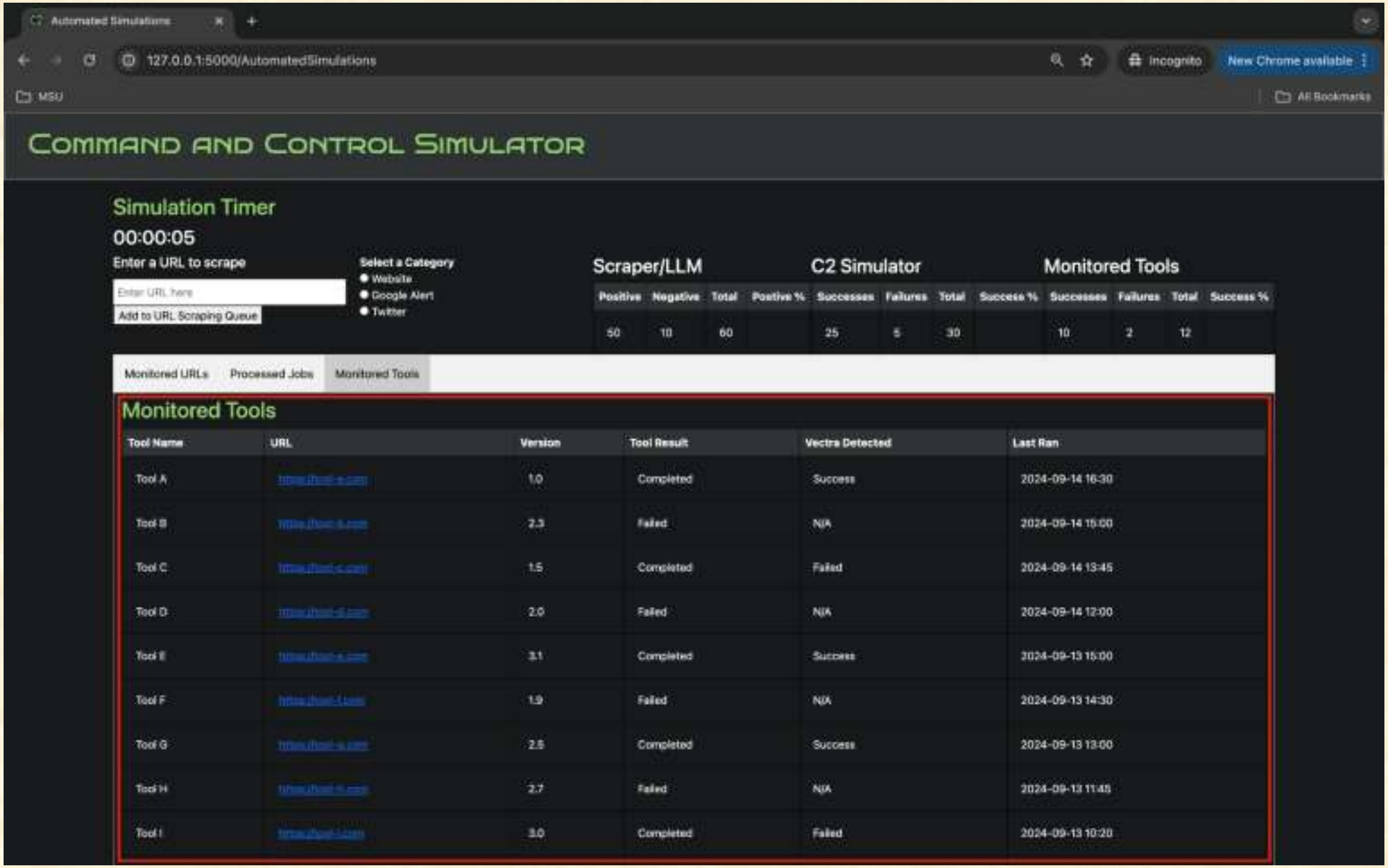
Encoding Type: ● Traditional ● Base64

Number of Tunnels:

Playbook

Job Type	Content	Delay	Scaling
Upload File		5	10%-20%
Portscan		3	80%-100%
Download File		7	70%-100%
Arbitrary Command	pwd	10	60%-100%

Screen Mockup: Monitored Tools



Command and Control Simulator

Simulation Timer

00:00:05

Enter a URL to scrape

Enter URL here

Add to URL Scraping Queue

Select a Category

● Website
● Google Alert
● Twitter

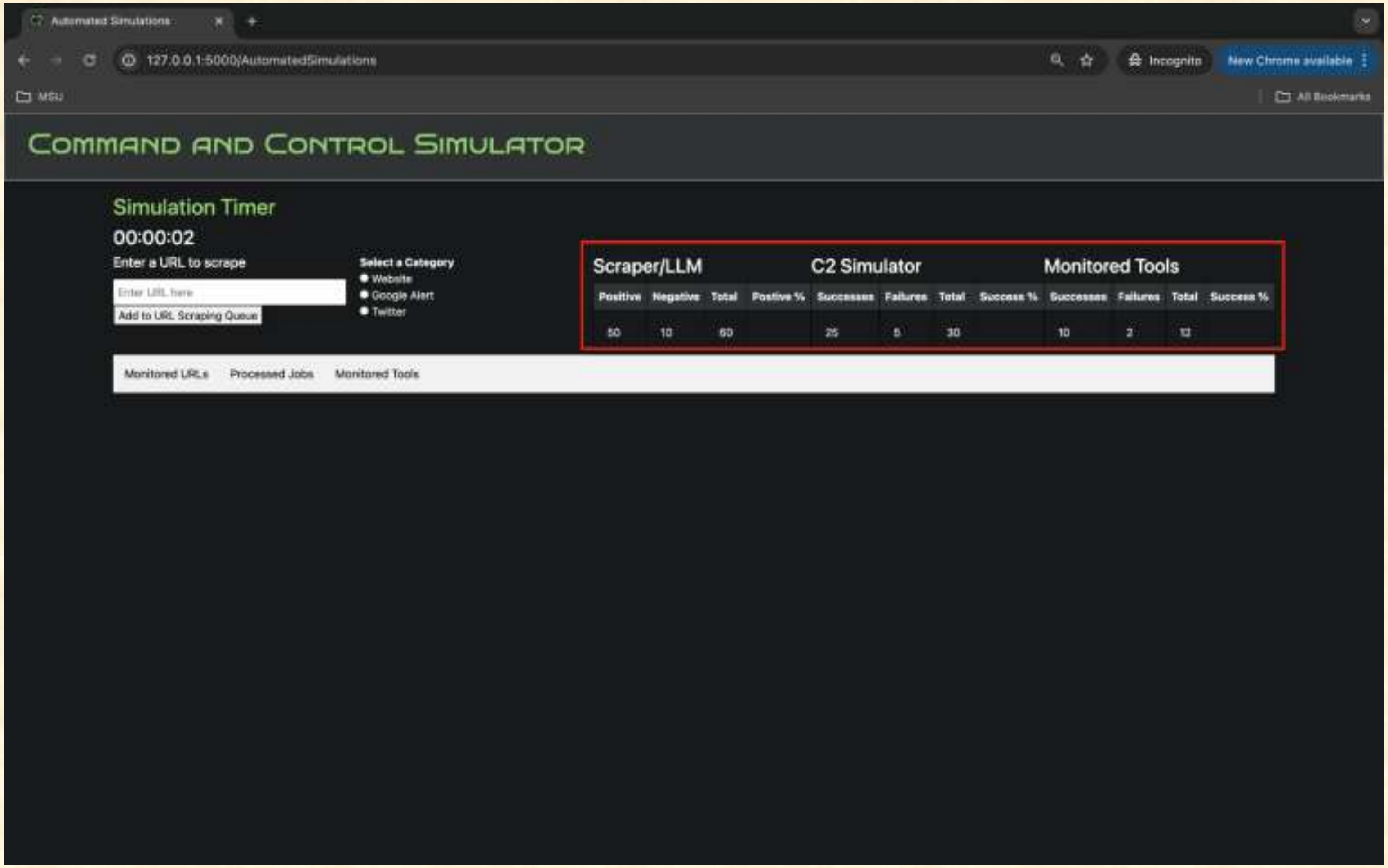
Scraper/LLM				C2 Simulator				Monitored Tools			
Positive	Negative	Total	Postive %	Successes	Failures	Total	Success %	Successes	Failures	Total	Success %
50	10	60		25	5	30		10	2	12	

Monitored URLs | Processed Jobs | Monitored Tools

Monitored Tools

Tool Name	URL	Version	Tool Result	Vectra Detected	Last Ran
Tool A	https://tool-a.com	1.0	Completed	Success	2024-09-14 16:30
Tool B	https://tool-b.com	2.3	Failed	N/A	2024-09-14 15:00
Tool C	https://tool-c.com	1.5	Completed	Failed	2024-09-14 13:45
Tool D	https://tool-d.com	2.0	Failed	N/A	2024-09-14 12:00
Tool E	https://tool-e.com	3.1	Completed	Success	2024-09-13 15:00
Tool F	https://tool-f.com	1.9	Failed	N/A	2024-09-13 14:30
Tool G	https://tool-g.com	2.5	Completed	Success	2024-09-13 13:00
Tool H	https://tool-h.com	2.7	Failed	N/A	2024-09-13 11:45
Tool I	https://tool-i.com	3.0	Completed	Failed	2024-09-13 10:20

Screen Mockup: Statistics



Command and Control Simulator

Simulation Timer

00:00:02

Enter a URL to scrape

Enter URL here

Add to URL Scraping Queue

Select a Category

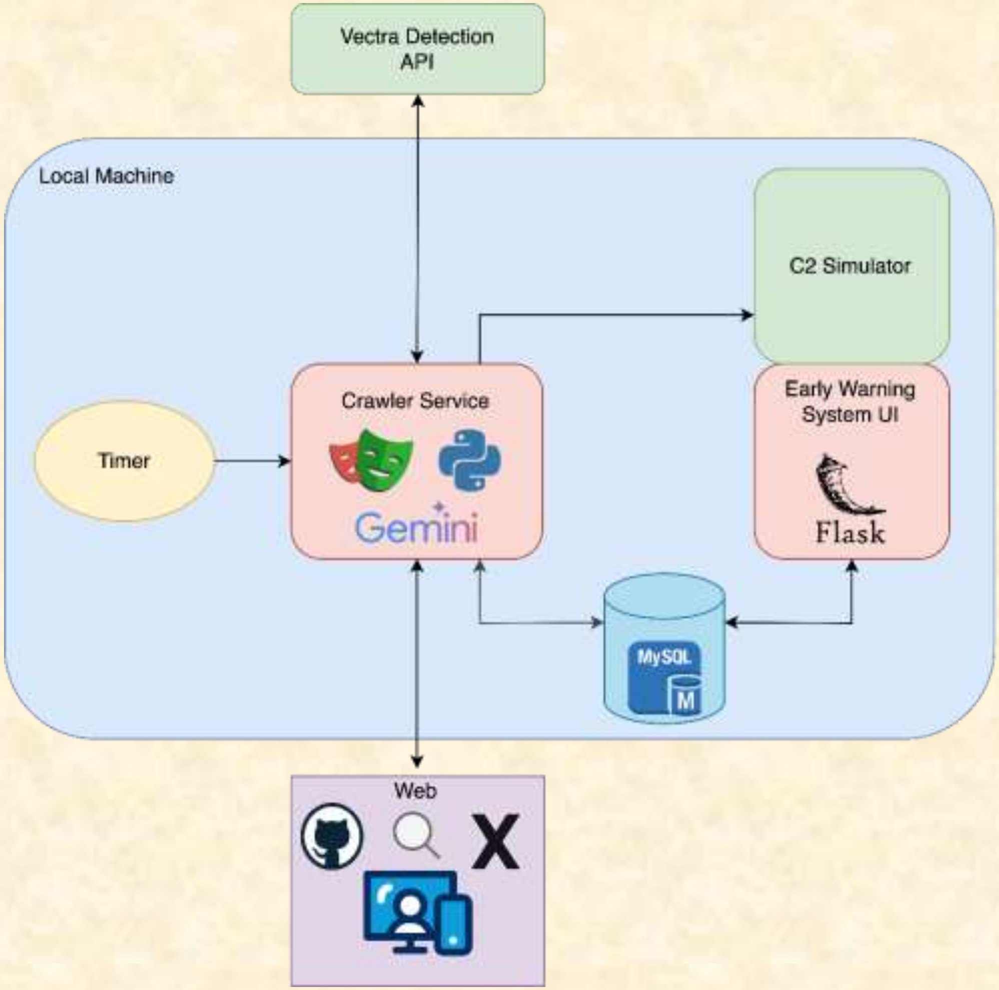
- ● Website
- ● Google Alert
- ● Twitter

Scraper/LLM				C2 Simulator				Monitored Tools			
Positive	Negative	Total	Postive %	Successes	Failures	Total	Success %	Successes	Failures	Total	Success %
50	10	60		25	5	30		10	2	12	

Project Technical Specifications

- Playwright and Python to scrape HTML content
- Gemini LLM to extract C2 configuration parameters
- Frontend built using Flask, HTML, CSS, and Jinja
- Backend and connector code written in Python
- MySQL for data storage

Project System Architecture



Project System Components

- Hardware Platforms
  - Computers
- Software Platforms / Technologies
  - Playwright
  - Gemini
  - Flask
  - VSCode
  - Pyshark
  - MySQL
  - Python

Project Risks

- Website Accessibility
  - Some URLs require authorization we don’t have
  - Human manned accounts as well as utilizing alternative data sending from sources that have it
- Website Content Filtration
  - Certain websites contain tags that will not be standardized
  - We can utilize our LLM to ignore the tags in the summary
  - We can develop a filtration system within our webscraper
- Automation of Cyberattack Tools
  - Cyberattack tools that are parsed in by the user need to be ran without being known
  - We can make use our LLM to find out how to run any given tool and then use Argparse to run the commands that the LLM returns
- High Cost of LLM Model Tiers
  - Calling the API for LLMs can be very pricey and our project triggers the “high risk” filters
  - We need to prompt engineer and change data so that our prompts won’t get an invalid call

Questions?

? ? ? ?

? ?

? ? ?