



राष्ट्रीय प्रौद्योगिकी संस्थान सिक्किम
National Institute of Technology Sikkim
An Institute of National Importance

PRESENTATION ON THE TOPIC
Online Click Fraud Detection and Prevention System

Submitted To:

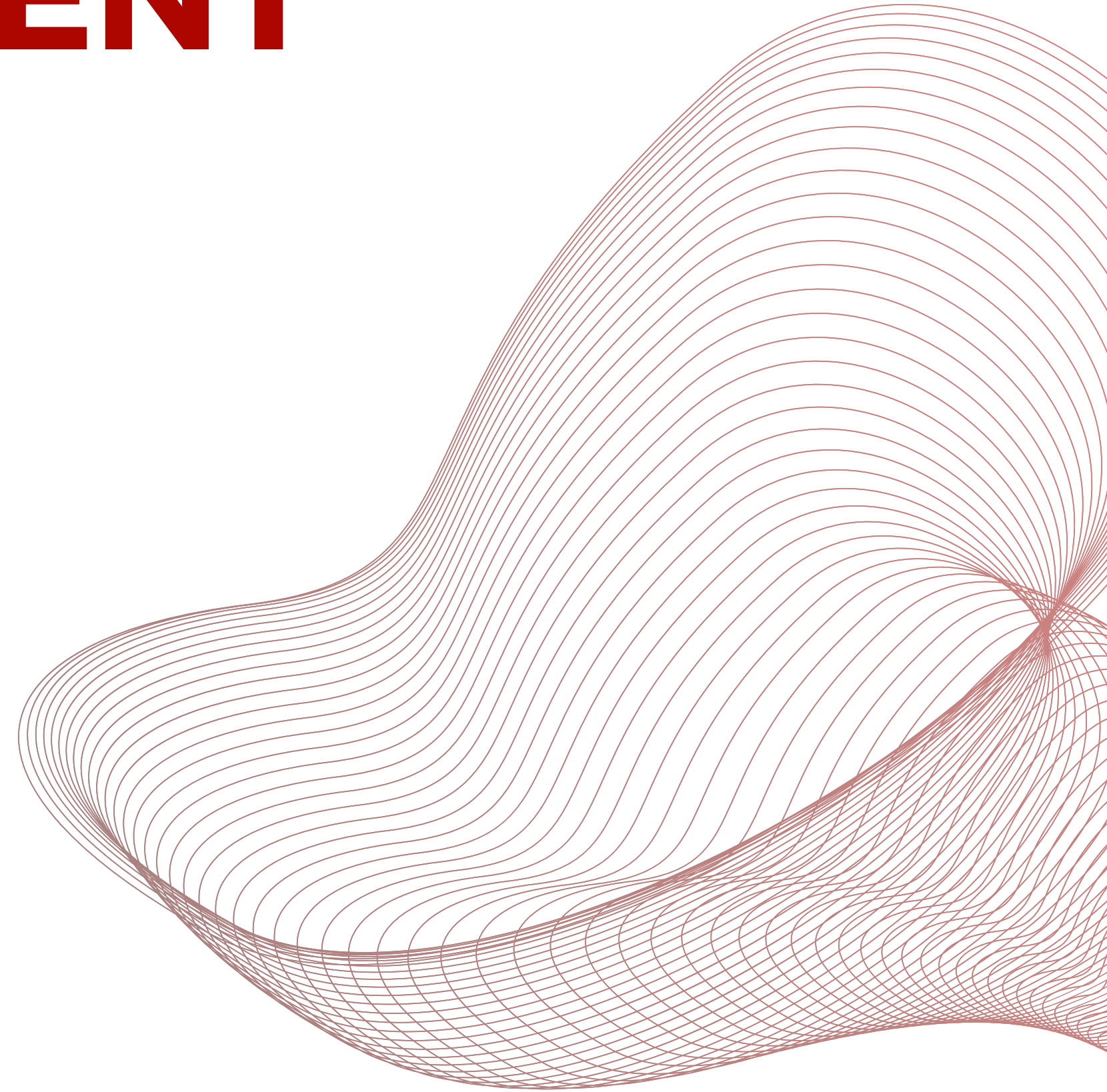
Dr. Pankaj Kumar Keserwani
Assistant Professor
Department of Compute Science and Engineering

Submitted By:

Rohit Mohan Roy (B200033CS)
Jigme Wangchuk Sherpa (B200044CS)
Prasan Subba (B200048CS)
Anurag Singh (B200058CS)
Vedaant Gajmer (B200069CS)

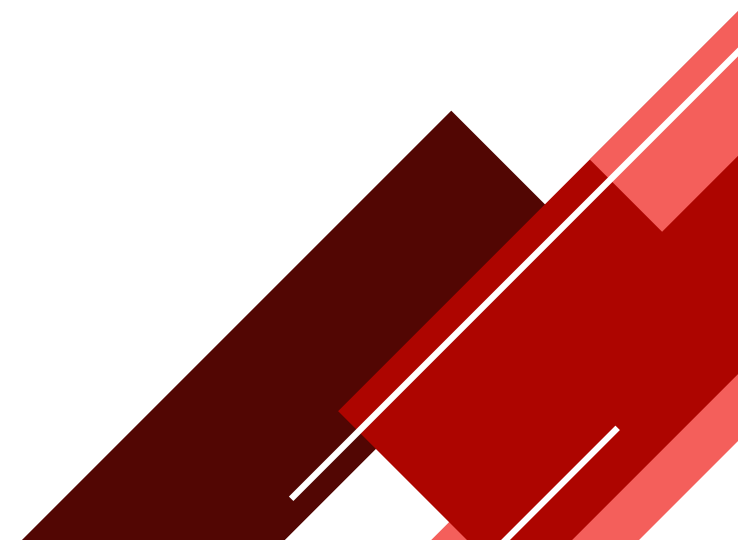
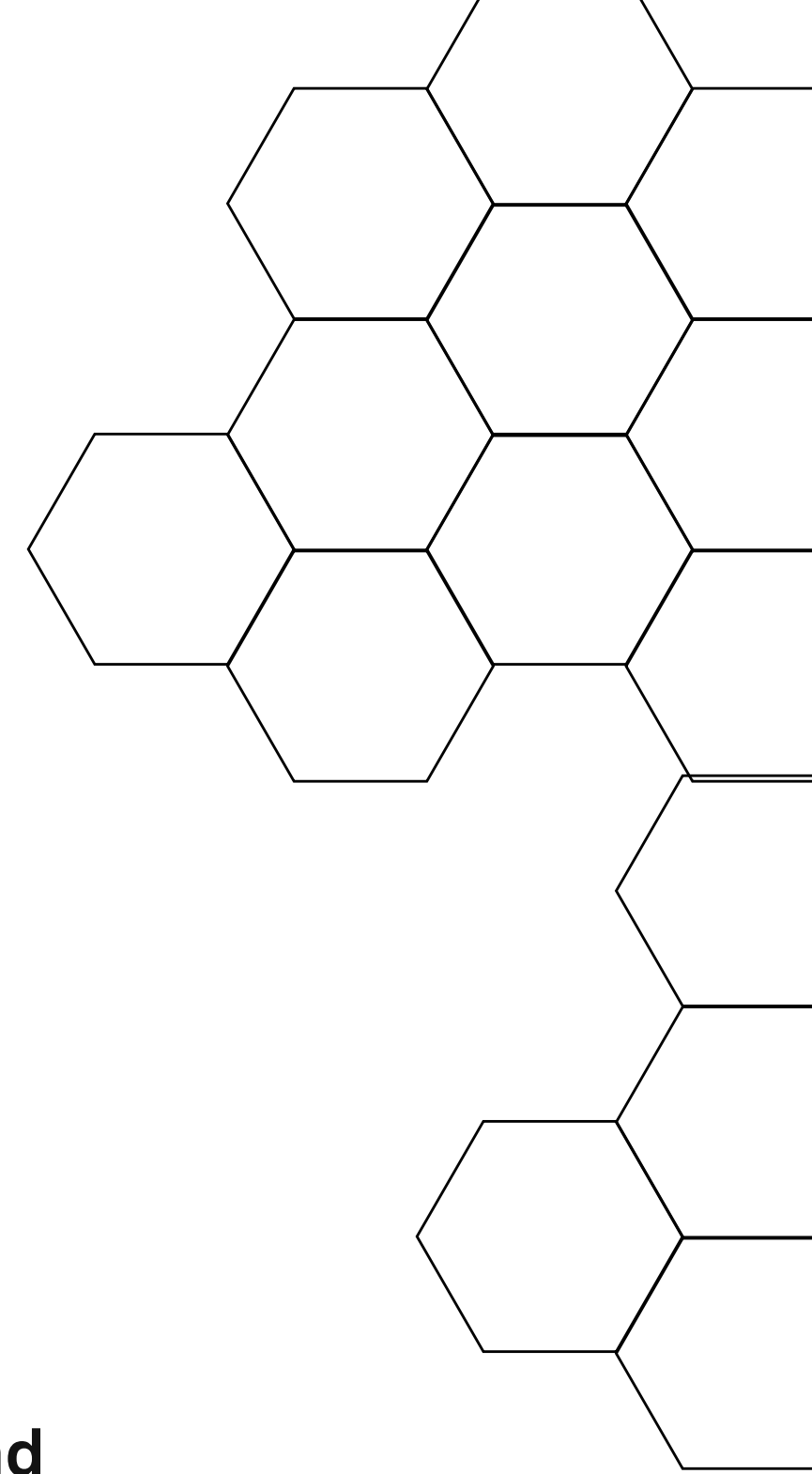
TABLE OF CONTENT

- Introduction
- Objective
- Technical Review
- Problem statement
- Proposed Framework
- List of attributes.
- Rules
- Methodology
- Implementation
- Future Works



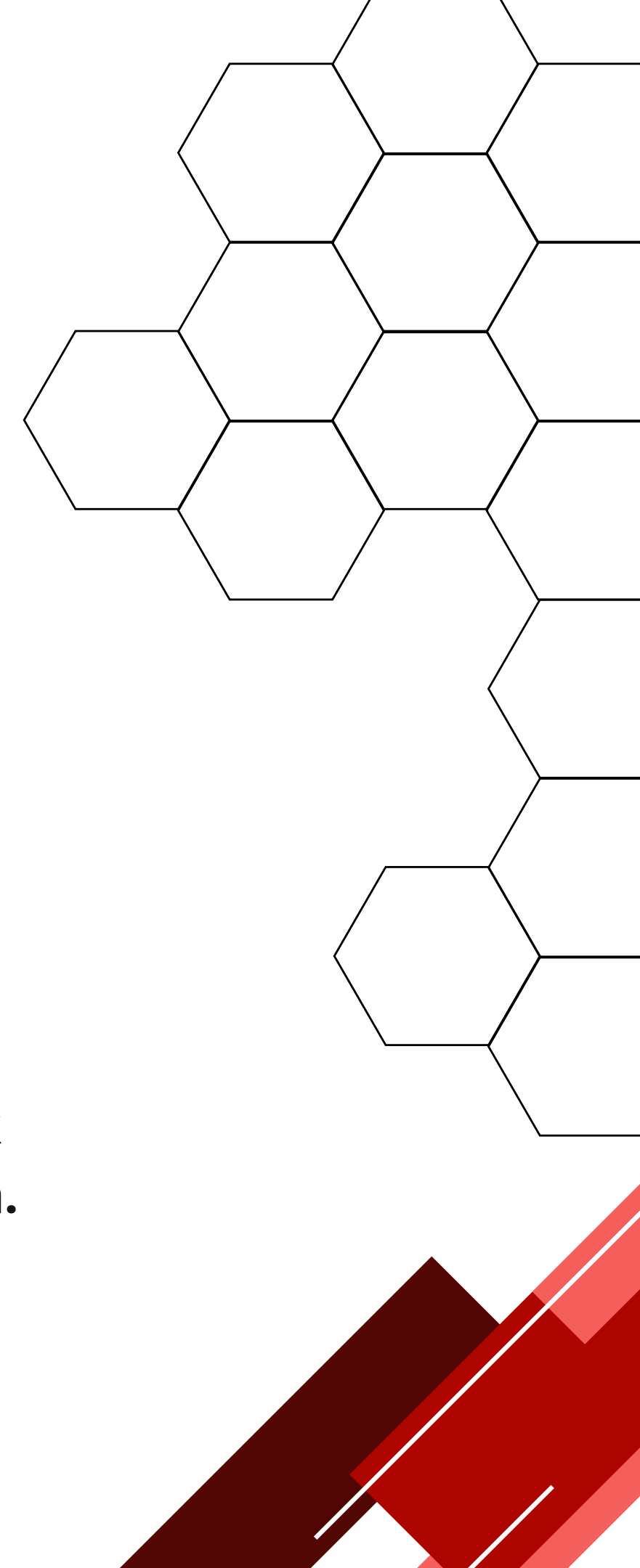
INTRODUCTION

- Online advertising is a rapidly growing industry, with substantial investments, particularly in digital advertising.
- Click fraud detection is a complex task involving the identification of click intent based on technical data and context information.
- Existing literature primarily focuses on advertisers, leaving a gap in research on click fraud detection within ad networks.
- The motivation for this work is to propose a system for click fraud detection and prevention within ad networks, aiming to protect advertisers' interests.
- This project introduces the concept of implementing a click fraud detection and prevention system , emphasizing design choices and system architecture.



OBJECTIVES

- The main objective is to propose and implement a click fraud detection and prevention system.
- This system is designed for application within an ad network, one of the key players in the online advertising environment.
- The primary goal of the system is to protect the interests of advertisers by ensuring the quality of clicks and preventing click fraud.
- Click fraud is a critical challenge in online advertising, and the system aims to address this issue.
- The objective involves developing a system that can identify and prevent click fraud, thus enhancing the overall integrity of the online advertising ecosystem.





CLICK AD FRAUD TECHNIQUES

Click Farms

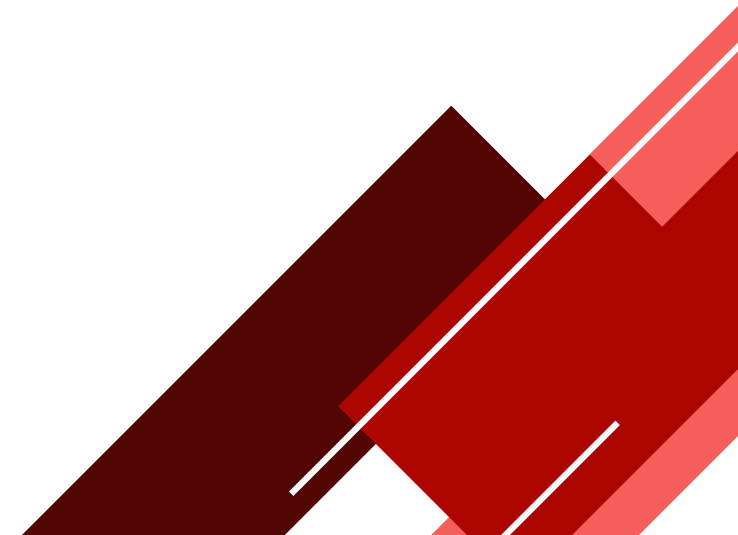
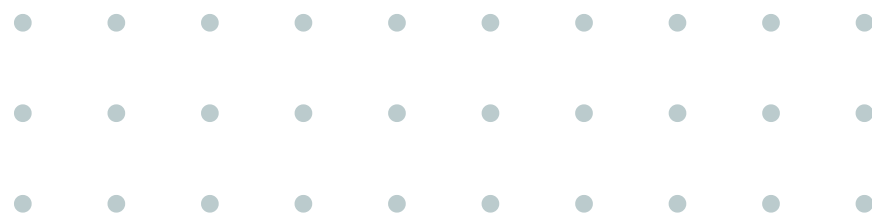
Click farms are operations where individuals or automated scripts are employed to generate a large volume of clicks on online advertisements.

Bots and Automated Scripts

Bots and automated scripts are computer programs designed to mimic human behavior online, including clicking on ads.

Ad Stacking

Ad stacking involves placing multiple ads on top of each other within a single ad placement, with only the top ad visible to users.



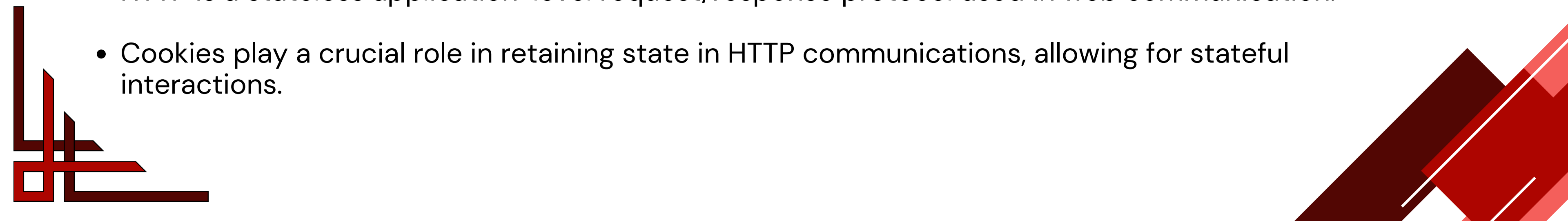


TECHNICAL REVIEW

Online Advertising Concepts

- Online advertising involves four primary agents: advertisers, publishers, users, and ad networks.
- Payment methods include Pay-Per-Impression (PPI), Pay-Per-Mille (PPM), Pay-Per-Click (PPC), and Pay-Per-Conversion (PPA), with a focus on PPC.
- Advertisers seek to reach their target audience through publishers' platforms, facilitated by ad networks.

HTTP Protocol:

- Understanding the HTTP protocol is crucial for grasping how the system operates.
 - HTTP is a stateless application-level request/response protocol used in web communication.
 - Cookies play a crucial role in retaining state in HTTP communications, allowing for stateful interactions.
- 

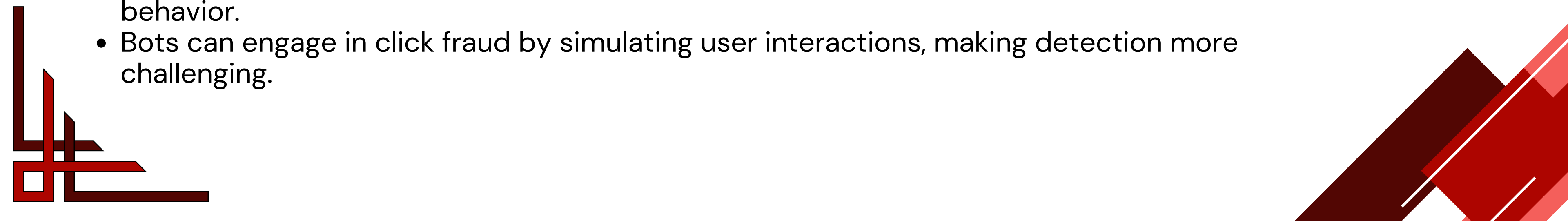


TECHNICAL REVIEW CONTD.

Related Work:


- Existing literature mentions some related work on click fraud detection, including the Microsoft adCenter system and techniques used on the advertiser's side.
- However, there is a notable gap in literature concerning systems used by ad networks, and limited elaboration on technical implementation details.

Click Fraud Attack Methods:

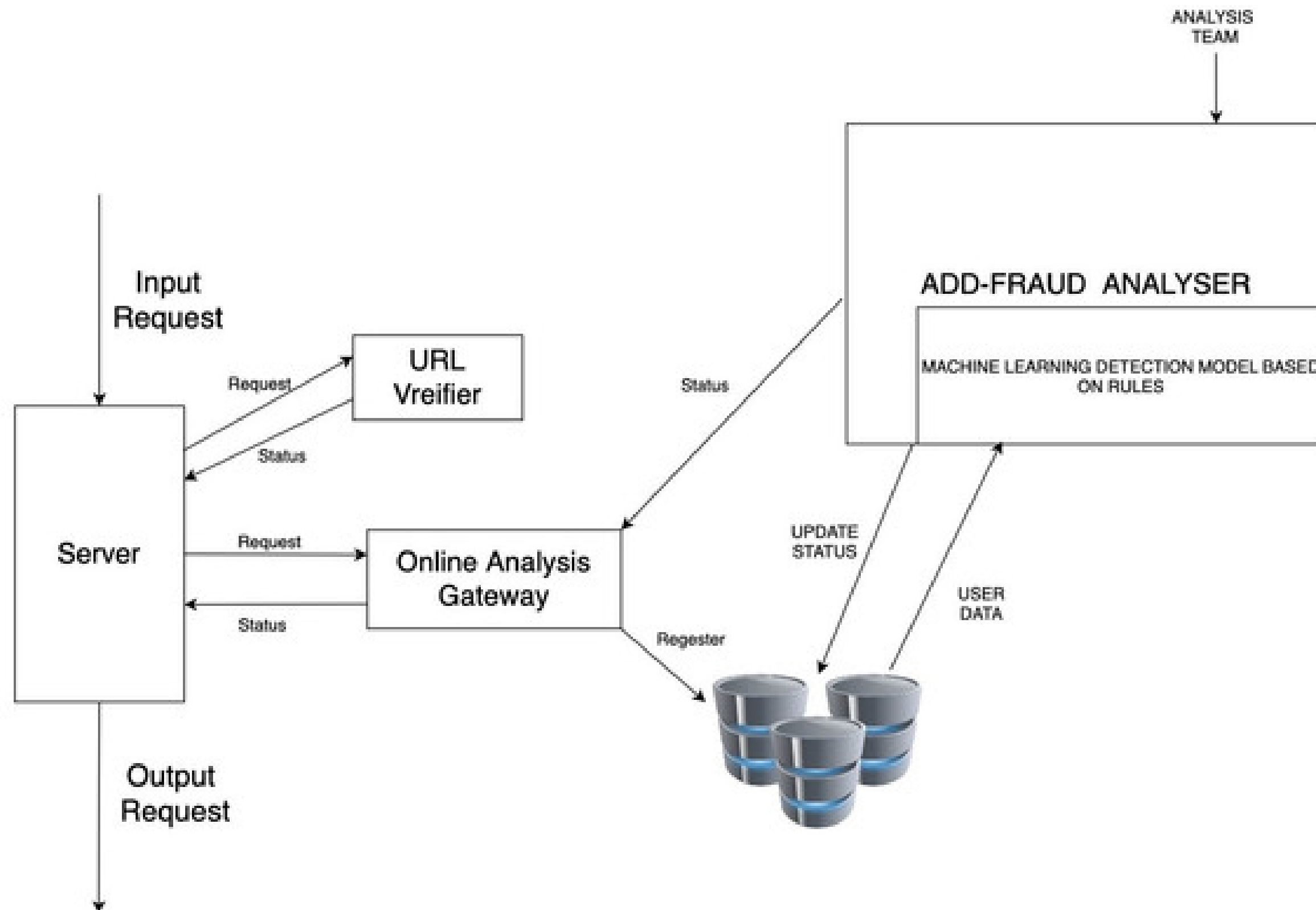
- Click fraud can be committed by both people and automated bots.
 - Basic human-initiated click fraud involves accessing a publisher's page and clicking on ads.
 - More sophisticated attacks may employ multiple human attackers, mimicking legitimate user behavior.
 - Bots can engage in click fraud by simulating user interactions, making detection more challenging.
- 



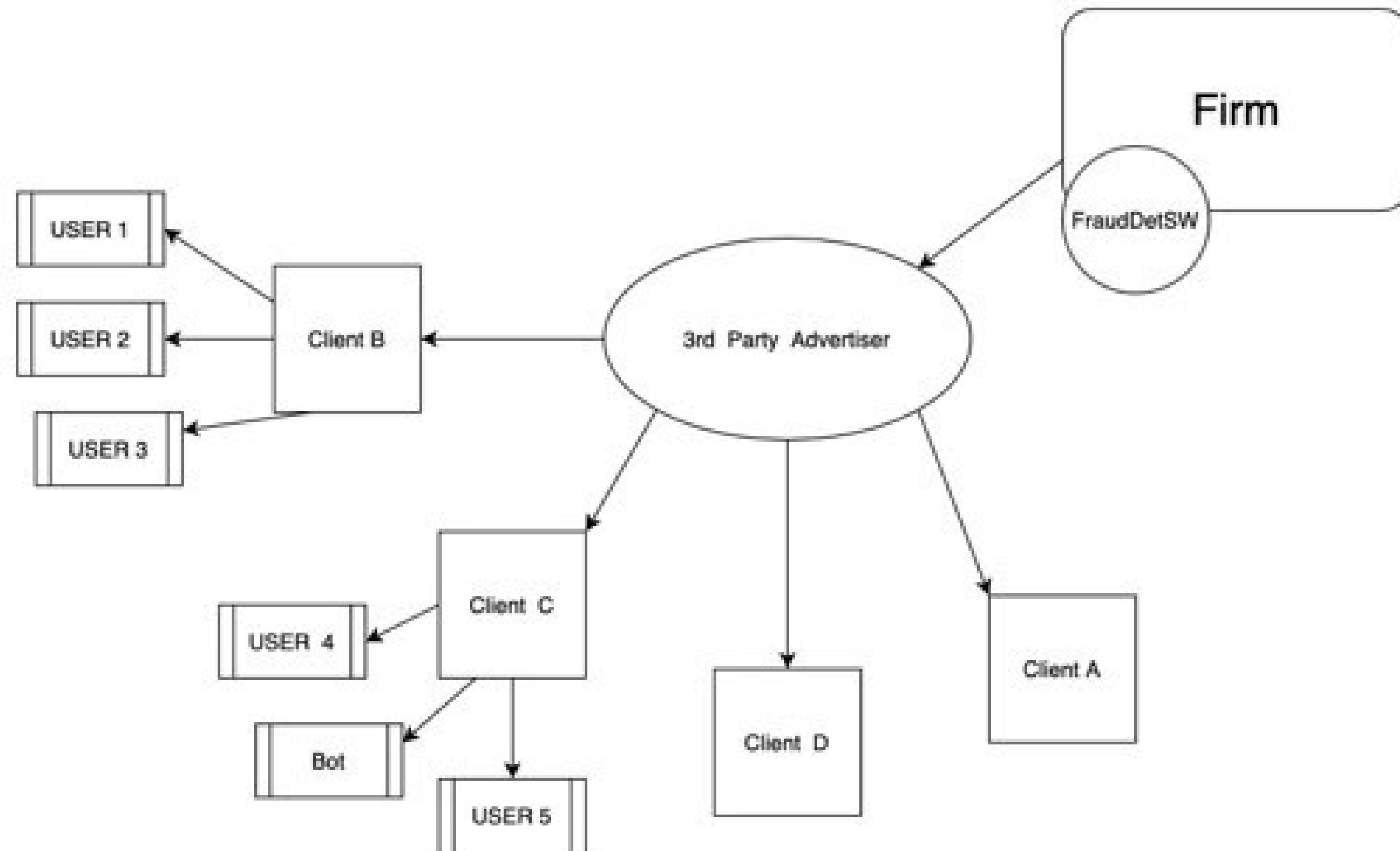
PROBLEM STATEMENT

- Click fraud in online advertising is a complex problem involving both manual and automated fraudulent activities.
 - Existing research primarily focuses on advertisers, creating a gap in understanding and addressing click fraud within ad networks.
 - Effective solutions require a deep technical understanding, including the HTTP protocol and cookies.
 - Evolving tactics like botnets and low-frequency attacks make click fraud detection more challenging.
 - Click fraud compromises the integrity of ad networks and directly affects advertisers and firms.
- 

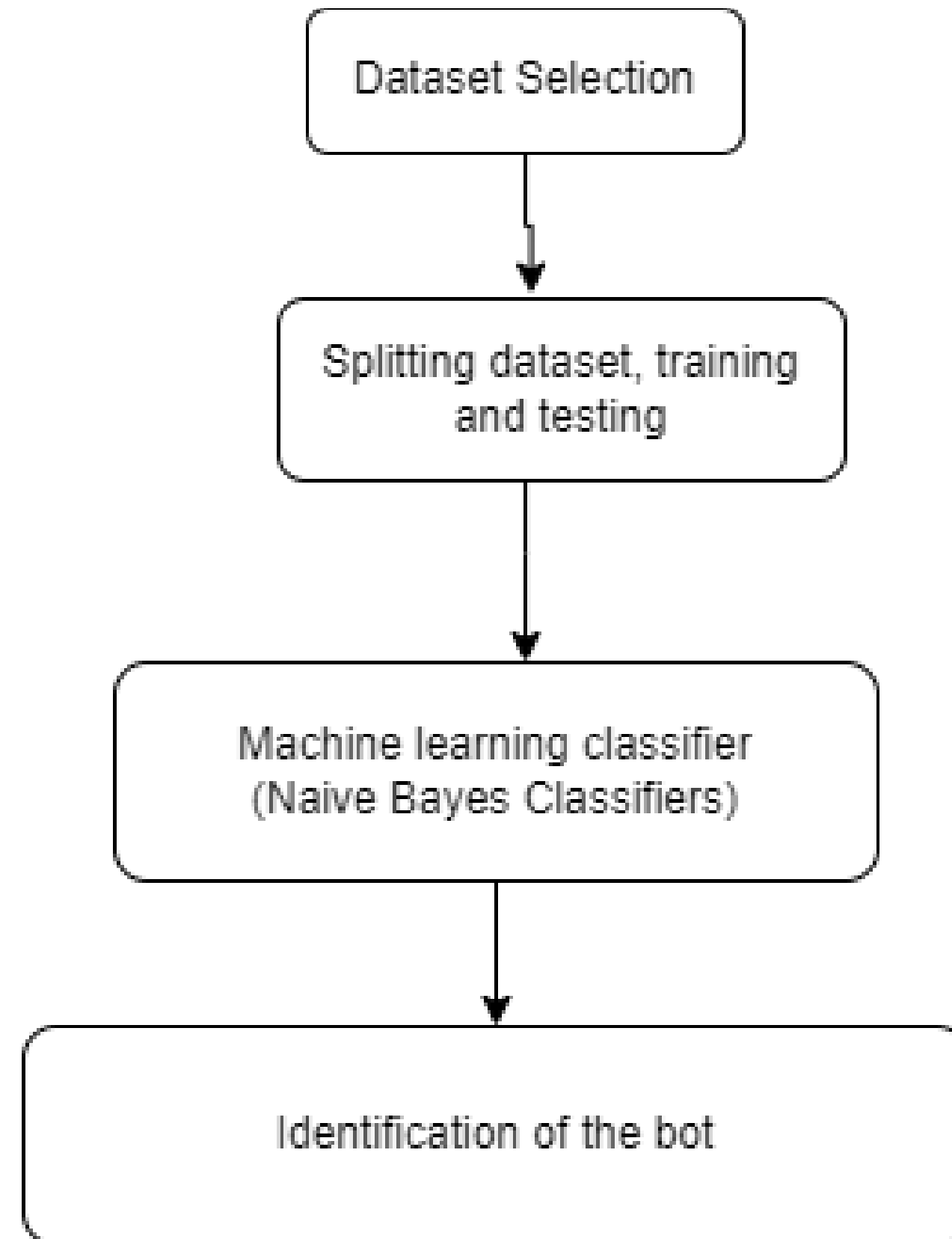
PROPOSED FRAMEWORK



PROPOSED FRAMEWORK CONTD.



PROPOSED FRAMEWORK CONTD.



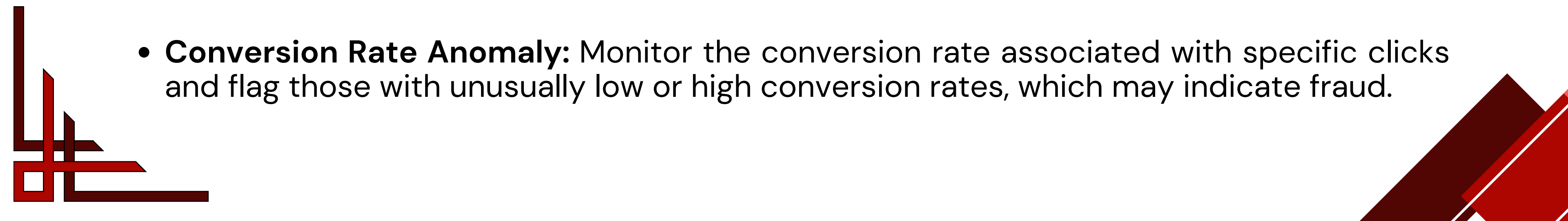


List of Attributes

- Timestamp
- IP Address
- Country Code
- Country
- Continent Code
- Continent
- Region
- Region Name
- City
- District
- Latitude
- Longitude
- Timezone
- Offset
- Internet Service Provider
- Organization
- AS
- AS Name
- Mobile Flag
- Proxy Flag
- Hosting
- Bot Flag
- Host ID
- User Agent
- Fetch Mode
- Forwarded Host ID
- Forwarded_to_host_ID
- Currency



RULES

- **Click Frequency Rule:** Monitor and flag excessive clicks from the same IP address or device within a short time frame, which may indicate click fraud.
 - **Geographic Anomaly Rule:** Detect clicks originating from unusual or unexpected geographic locations, as this can be a sign of click fraud.
 - **Click Time Patterns:** Analyze click time patterns to identify irregularities, such as constant clicks at odd hours or consistent, unnatural intervals.
 - **User Agent Analysis:** Examine user agents to identify bots or fake devices, which can be used to generate fraudulent clicks.
 - **Click Through Rate (CTR) Rule:** Set thresholds for CTR and flag instances where the CTR is significantly higher or lower than expected, as extreme values may suggest click fraud.
 - **Conversion Rate Anomaly:** Monitor the conversion rate associated with specific clicks and flag those with unusually low or high conversion rates, which may indicate fraud.
- 

METHODOLOGY

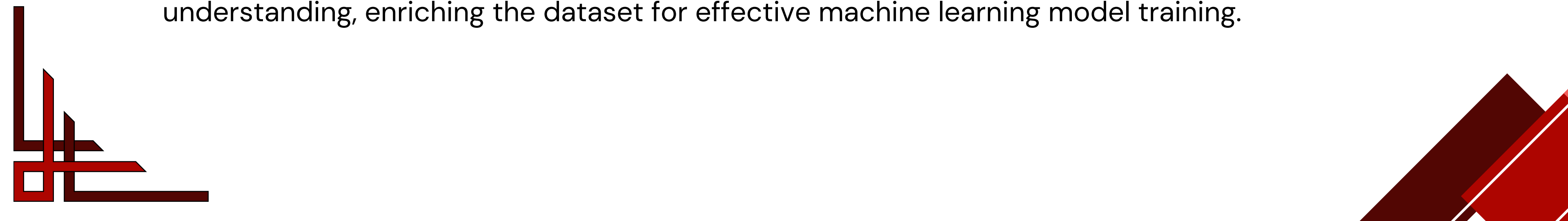
The primary objective of this project is to establish a robust methodology for the detection of fraudulent activities, with a specific focus on discerning bot-driven interactions via a provided ngrok link. This comprehensive approach combines the utilization of the Express.js framework for web server creation and management, integration with the IP-API service for detailed IP address analysis, and a structured data logging mechanism for subsequent analysis. The following extended report delves deeper into each step of the data collection process.

Data Collection Process:

- **Express.js Setup:** Express.js is employed as the foundation for the web server, streamlining the handling of incoming HTTP requests. Leveraging Express's middleware functionality, the system efficiently parses incoming JSON data and serves static files, including an HTML page that enhances user interaction.
- **IP Retrieval:** The ngrok IP address is extracted from the incoming request headers, with careful consideration given to both the 'x-forwarded-for' field and the remote address. This dual approach ensures the acquisition of accurate and reliable information about the public IP address associated with the entity interacting with the ngrok link.



METHODOLOGY CONTD.

- **IP-API Service Integration:** The ngrok IP address is the key for requests to the IP-API service, providing detailed IP information. The resulting dataset covers geographical location, ISP details, and flags for proxy or mobile association.
 - **Data Structuring and Logging:** Information is structured into a JSON format, including timestamp, coordinates, ISP details, and bot activity flags. This dataset is logged for analysis and model training.
 - **Error Handling:** Robust error-handling manages issues with log file operations and the IP-API service, ensuring data reliability.
 - **Historical Data Storage:** Existing logs enable continuous data collection, aiding trend analysis for identifying fraudulent behavior.
 - **Manual Attribute Assignment:** A 'bot' attribute is manually assigned based on contextual understanding, enriching the dataset for effective machine learning model training.
- 

IMPLEMENTATION

```
import requests
import time

def fetch_data(url, num_requests=10, delay_seconds=1):
    for _ in range(num_requests):
        try:
            response = requests.get(url)

            if response.status_code == 200: #in case status is success
                print("Data fetched successfully:")
                print(response.text)
            else:
                print(f"Failed to fetch data. Status code: {response.status_code}")

        except Exception as e:
            print(f"An error occurred: {str(e)}")

        # Add a delay between requests
        time.sleep(delay_seconds)

if __name__ == "__main__":
    url = 'https://d220-2a09-bac5-3e64-18c8-00-278-cc.ngrok-free.app/'

    num_requests = 100
    delay_seconds = 1

    fetch_data(url, num_requests, delay_seconds)
```

IMPLEMENTATION CONTD.

```
import webbrowser
import time

def open_link_in_tab(url, num_times=100, delay_seconds=1):
    for _ in range(num_times):
        webbrowser.open_new_tab(url)
        time.sleep(delay_seconds)

if __name__ == "__main__":
    # Replace 'your_url_here' with the actual URL you want to open
    url = 'https://d220-2a09-bac5-3e64-18c8-00-278-cc.ngrok-free.app/'

    # Set the number of times to open the link and the delay between each open
    num_times = 10
    delay_seconds = 1

    # Call the open_link_in_tab function with the specified URL, number of times, and delay
    open_link_in_tab(url, num_times, delay_seconds)
```

IMPLEMENTATION CONTD.

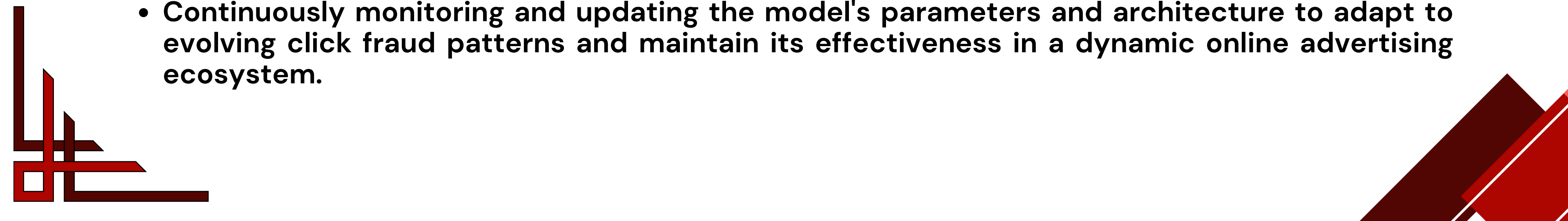
```
async function fetchData(url, numRequests = 10, delayMilliseconds = 1000) {  
  for (let i = 0; i < numRequests; i++) {  
    try {  
      const response = await fetch(url);  
  
      if (response.ok) {  
        const data = await response.text();  
        console.log("Data fetched successfully:");  
        console.log(data);  
      } else {  
        console.log(`Failed to fetch data. Status code: ${response.status}`);  
      }  
    } catch (error) {  
      console.error(`An error occurred: ${error}`);  
    }  
  
    await new Promise(resolve => setTimeout(resolve, delayMilliseconds));  
  }  
}  
  
const url = 'https://d220-2a09-bac5-3e64-18c8-00-278-cc.ngrok-free.app/';  
  
const numRequests = 50;  
const delayMilliseconds = 100;  
  
fetchData(url, numRequests, delayMilliseconds);
```

IMPLEMENTATION CONTD.

```
{
  "timestamp": "2023-11-14T10:46:14.152Z",
  "ip": "2a09:bac5:3da6:101e::19b:7",
  "status": "success",
  "continent": "Asia",
  "continentCode": "AS",
  "country": "India",
  "countryCode": "IN",
  "region": "KL",
  "regionName": "Kerala",
  "city": "Kochi",
  "district": "",
  "zip": "",
  "lat": 9.9312,
  "lon": 76.2673,
  "timezone": "Asia/Kolkata",
  "offset": 19800,
  "currency": "INR",
  "isp": "Cloudflare, Inc.",
  "org": "Cloudflare WARP",
  "as": "AS13335 Cloudflare, Inc.",
  "asname": "CLOUDFLARENET",
  "reverse": "",
  "mobile": false,
  "proxy": true,
  "hosting": false,
  "headers": {
    "host": "425f-2a09-bac5-3da6-101e-00-19b-7.ngrok-free.app",
    "user-agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36 Edg/119.0.0.0",
    "accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7",
    "accept-encoding": "gzip, deflate, br",
    "accept-language": "en-US,en;q=0.9,en-IN;q=0.8",
    "cache-control": "max-age=0",
    "cookie": "abuse_interstitial=425f-2a09-bac5-3da6-101e-00-19b-7.ngrok-free.app",
    "sec-ch-ua": "\"Microsoft Edge\";v=\"119\", \"Chromium\";v=\"119\", \"Not?A_Brand\";v=\"24\"",
    "sec-ch-ua-mobile": "?0",
    "sec-ch-ua-platform": "\"Windows\"",
    "sec-fetch-dest": "document",
    "sec-fetch-mode": "navigate",
    "sec-fetch-site": "same-origin",
    "sec-fetch-user": "?1",
    "upgrade-insecure-requests": "1",
    "x-forwarded-for": "2a09:bac5:3da6:101e::19b:7",
    "x-forwarded-host": "425f-2a09-bac5-3da6-101e-00-19b-7.ngrok-free.app",
    "x-forwarded-proto": "https"
  }
},
```



FUTURE WORKS

- Establishing a comprehensive data collection system to gather a diverse and high-quality dataset for training the Click Fraud Detection and Prevention System. Developing an advanced ML model with deep learning capabilities to enhance the accuracy and efficiency of the system.
 - Implementing sophisticated algorithms for anomaly detection and behavioral analysis to identify and mitigate emerging patterns of fraudulent activities in real-time.
 - Employing advanced feature engineering techniques and data preprocessing methods to optimize the model's performance and detect complex click fraud schemes effectively.
 - Conducting rigorous testing and validation procedures, including A/B testing and simulated real-world scenarios, to assess the model's robustness and reliability under various operational conditions and potential adversarial attacks.
 - Continuously monitoring and updating the model's parameters and architecture to adapt to evolving click fraud patterns and maintain its effectiveness in a dynamic online advertising ecosystem.
- 

The background features a white canvas with several red geometric elements. In the top right, there is a solid red square. In the bottom left, there is another solid red square. A large, solid red rectangle is positioned on the right side, partially cut off by the edge. Two intricate, wavy line patterns made of thin red lines are located on the left and right sides, framing the central text. The text 'THANK YOU' is centered in a bold, dark red, sans-serif font with a slight 3D effect.

THANK YOU