



(REVIEW ARTICLE)



Digitally improving UK police surveillance and incidence response using real-time crowd reporting app: Digipolice

Onyemachi Joshua Ndubuisi ¹, Gift Adene ^{2,*}, Belonwu Tochukwu Sunday ³, Chinedu Emmanuel Mbonu ³ and Adannaya Uneke Gift-Adene ²

¹ Teesside University Middlesbrough, United Kingdom.

² Department of Computer Science, Akanu Ibiam Federal Polytechnic, Unwana, Nigeria.

³ Department of Computer Science, Nnamdi Azikiwe University, Awka, Nigeria.

Global Journal of Engineering and Technology Advances, 2024, 18(03), 124–138

Publication history: Received on 14 February 2024; revised on 19 March 2024; accepted on 22 March 2024

Article DOI: <https://doi.org/10.30574/gjeta.2024.18.3.0048>

Abstract

Overtime, evolving technologies has made ways of monitoring and controlling crime and action in the United Kingdom (UK) become more challenging which needs optimization with the help of technologies. Also with the rate of crime across cities in the UK like mass shootings, murder and gang gun battles, the UK Police system within localities are overwhelmed or may not efficiently cope with crime alerts and incidences coming in at increasing frequencies. However looking in the area of real time crowd reporting, there is need to leverage technology to enhance public safety, improve incident response, and foster community engagement. This necessitated the aim of this study to develop a digital policing surveillance app for crowd reporting in the UK. By utilizing digital policing surveillance apps, the aim is to improve crowd management capabilities by enabling real-time reporting of incidents and observations from both the public and law enforcement personnel. This was positioned to facilitate better situational awareness, efficient resource allocation, and effective response strategies. To achieve the objectives, Python Programming Language 3.8 were deployed together with a Django Framework and a Microsoft Visual Studio Build Tools. In the DP system, the Model-View-Controller (MVC) pattern, which divides an application into three interconnected components—the model, the view, and the controller—was used for the architectural design of the system. For the database, Django's ORM makes it simple to support many databases, and in this project straightforward SQLite3 database were utilised which is ideal for small-footprint applications. Using WebSockets to establish real-time, bidirectional communication between web clients and servers, the user interface and intelligence board for the system were developed. A clean designed frontend design system with Bootstrap version 5—an open-source, reusable design framework created by Twitter—was selected for its easy user interface. The system was put through 50 concurrent load tests with actual users swarming the app since the DP app was subjected to hundreds of people complaining in real time. 2. The host computer had a Core i5 vPro CPU running at 4GHz and 4GB of RAM. The result shows that 50 users successfully generated an average of 29.7 requests per second. The system scaled up nicely to 50 users and reached its highest response time of 2600 milliseconds (2.6 seconds) in total. A dramatic rise in activity and a longer response time occurred after there were more than 50 users. The system could not operate at its peak efficiency with 100 users because response times were inconsistent and exceeded 7000 milliseconds (7 seconds) for each request. The effectiveness of the online deployment was evaluated with respectable results, and the system requirements also underwent successful testing. The appropriate investigation of ethical issues was done to meet requirements. DigiPolice has the potential to develop into a cutting-edge programme for real-time crowd surveillance and incident reporting. UK Police can employ it to as advancement to its “My Police” with as fast-track real-time and collaborative crowd reporting enhancements.

Keywords: Crime; DigiPolice; United Kingdom; Surveillance; Policing

* Corresponding author: Gift Adene

1. Introduction

The most crucial responsibility for the police is criminal identification since it requires them to look everywhere for it [1]. Law enforcement organizations play a crucial role in maintaining peace, order, and security [2]. In the UK, separate organizations have been established to fulfill this purpose in different regions, namely England/Wales, Northern Ireland, and Scotland. These police forces are entrusted with specific authorities to safeguard peace, protect property, and prevent criminal activities. Notably, Ireland had the world's first organized police force, which later extended to a part of the United Kingdom [3]. The UK's police forces underwent modernization in the 1940s, setting a global benchmark for policing services [4]. However, in recent years, despite various initiatives and reforms, the UK's police forces have lost their status as the global gold standard [5]. Over the past five decades, there has been a shift towards more process-driven decision-making and the integration of technology in policing practices [5].

With changing times and evolving technology, the ways of monitoring and controlling crime and action in UK has become more challenging which needs optimization with the help technologies [6]. These technological advances though have opened new arenas of crime with prevalent cybercrime and fraud. These challenges are being met around the world with reforms and new teams being made to tackle the new crimes and challenges [5]. Furthermore with the rate of crime across cities in the UK like mass shootings, murder and gang gun battles, the UK Police system within localities maybe overwhelmed or may not efficiently cope with crime alerts and incidences coming in at increasing frequencies [7]; and thus bearing witness to the fact that the core among the challenges faced by UK police currently is prompt incidence monitoring and surveillance.

Police practice is increasingly reliant on digital surveillance technologies, which have become prominent in recent years [5]. Notable innovations that attract public attention include predictive policing, improved online surveillance, and portable digital police equipment, among others [8]. The rapid growth of these technologies led to the introduction of digital interventions by the England and Wales Police force, such as real-time facial recognition for monitoring criminal activity, particularly in densely populated public areas. However, not everyone finds these interventions acceptable. The UK police have faced accusations of violating ethical standards due to their use of live facial recognition technology as a crime-fighting tool [9]. [10] conducted a recent study on the implementation of facial recognition technology by police forces in London and south Wales. The report raised significant concerns regarding the "insufficient mechanisms for addressing grievances" in cases where harm is caused. It emphasized the importance of safeguarding human rights and enhancing accountability before expanding the use of facial recognition technology. [11] argues that the UK police should be prohibited from utilizing live facial recognition technology in any public spaces due to their violation of ethical standards and laws protecting human rights. These concerns have impeded the progress of the UK police force in swiftly implementing real-time crime surveillance and monitoring. However as a remedy to this challenge, this study is poised to bridge the gap by developing an application called DigiPolice (DP) which will help enhance Police response to crime incidences by applying crowdsourcing technique which have been established to help combat difficulty in monitoring crime in real-time over larger cities by leveraging community participation [5]. In the DP app, citizens report a crime or suspicious activity in real time to Police via the online public location-based app. The message notification is broadcasted in real time to the UK Police intelligence message board with the location coordinates of the reporter and nearest Police officers/stations are alerted in real time.

The aim of this study is to develop a digital policing surveillance app for crowd reporting in the United Kingdom (UK). However, the specific objective is to first analyse the existing policing surveillance system by performing an artefact analysis of the system to look at the present methods utilised by UK police to conduct incident monitoring.

2. Review of Related Literature

Police surveillance refers to the monitoring and observation activities conducted by law enforcement agencies to gather information and maintain public safety [12]. It involves the use of various technological tools and techniques to collect data, monitor activities, and identify potential threats or criminal behaviour [13]. According to [14], the concept of police surveillance is rooted in the objective of preventing and deterring crime, protecting public order, and investigating criminal activities. However, the implementation and extent of police surveillance raise important considerations regarding privacy, civil liberties, and the balance between security and individual rights.

Moreover, surveillance comprises a variety of techniques used to acquire information about people in its widest meaning. These methods range from traditional practices of monitoring and tracking the activities of suspects to utilizing technological means and conducting covert undercover operations to gather detailed information on individuals [15]. With the advancement of technology, additional forms of surveillance are continually being integrated

into the toolkit of law enforcement agencies, particularly through electronic means [16]. While the primary motivation behind the development and use of these new technologies has been to prevent terrorist attacks, they also prove highly valuable in investigating crimes such as gambling, extortion, smuggling, bribery, loan sharking, drug trafficking, trafficking in stolen goods, and corruption of governmental authorities [17].

Considering that police surveillance is a recognized requirement in numerous investigations, and the constant introduction of new technologies can further enhance this function, the crucial question arises: How can law enforcement effectively manage surveillance operations and the utilization of devices without facing judicial resistance that may potentially lead to their prohibition? The key lies primarily in the manner in which a police department manages and supervises surveillance operations.

Firstly, the level of supervision and oversight necessary depends on the specific type of surveillance being conducted. In this thesis, three levels of surveillance have been identified in the model policy based on their common operational characteristics. It is important to acknowledge that some surveillance operations may not fit neatly into one classification, as certain aspects may overlap or be shared with others. With that said, the classifications utilized in this context are referred to as "monitoring," "covert surveillance," and "undercover operation."

2.1. Monitoring

This involves short-term and preliminary observation of the activities of an individual, group, or organization with the purpose of gathering information [18]. Typically, monitoring is conducted to determine whether a criminal investigation should be initiated and may encompass the transient observation of suspicious individuals or activities.

2.2. Covert Surveillance

In order to obtain data for a current criminal investigation, covert surveillance involves long-term or continuing monitoring of a targeted person, group, or organisation [19]. This form of surveillance often involves the use of invasive tactics, such as electronic eavesdropping or similar methods.

2.3. Undercover Operation

An authorised criminal investigation that involves an officer or officers assuming a false name or cover identity in order to infiltrate a group or organisation is referred to as an undercover operation. The goal is to learn more about people through the establishment of personal connections and other legal information-gathering techniques [20].

Monitoring represents the least intensive form of surveillance in terms of its potential impact on Fourth Amendment protections [18]. The observation and information gathering methods employed in monitoring are typically extensions of the types of observations and individual scrutiny carried out by patrol officers on a daily basis. When an individual or group becomes the target of focused surveillance, any information obtained as a result should be shared with a supervisor or other departmental personnel to determine whether sufficient grounds exist to initiate a criminal investigation [18]. Officers should not engage in prolonged, focused surveillance without informing their shift supervisor.

Since Leicestershire Police first utilized facial recognition technology at Download Festival in 2015, multiple UK police forces have adopted this technology. Its adoption has been spearheaded by South Wales Police and the Metropolitan Police. In a variety of contexts, including protests, sporting events, concerts, the Notting Hill Carnival, Remembrance Sunday, railway stations, major commercial areas and even beach locales, these forces have scanned hundreds of thousands of people using surveillance technologies. Police face recognition works by measuring and mapping a person's distinctive facial traits. This information is then transformed into a numerical code for comparison against other codes on secret watch lists. Up until recently, 'live' facial recognition—where a person's face is scanned and matched in real time with a watch list—had been the main technique used by the police. Usually, scanning was done using facial recognition cameras installed on police cars parked in populated places. Within their field of view, these cameras would take pictures of people, which the programme would then instantaneously compare to the database to identify.

However, there have been recent developments in the application of facial recognition by South Wales Police. They have announced the testing of facial recognition on officers' smartphones, enabling them to conveniently scan faces in public spaces. In addition to 'live' facial recognition, the Metropolitan Police has obtained software that enables 'retroactive' facial recognition. This refers to the scanning and comparison of faces in still images or previously recorded video footage with the watch list [21].

The Metropolitan Police employs Facial Recognition (FR) technology in various ways to ensure public safety, as stated by the [22]. The applications of FR technology in policing typically include the following:

- Real-time assistance: FR serves as a tool to aid officers in locating individuals on a "watchlist" who are being sought by the police. It provides immediate support to officers in identifying and apprehending such individuals.
- Officer-initiated usage: Officers can proactively capture an image of a person and utilize Facial Recognition software to determine their identity. This helps law enforcement even if the individual provides false or misleading information. Additionally, this application can be valuable in identifying unconscious or severely injured individuals who are unable to communicate their identity.
- Retrospective analysis: FR technology can be employed after an incident or event to assist officers in establishing a person's identity or determining whether their image matches other media stored in databases.

The deployment of Live Facial Recognition (LFR) cameras involves focusing on specific areas where individuals pass through. Their images are then streamed directly to the Live Facial Recognition system, which contains a watchlist consisting of wanted offenders, individuals sought by the courts, or those who pose a potential risk to themselves or others [22].

3. Methodology

Any sort of information system is built using a process known as the systems development methodology [23]. This study will make use of [24]'s concept of systems development methodologies. The four main aspects they concentrated on in their concept of systems development were:

- The first method is the development approach, which entails a number of guiding principles, aims, and ideas. Examples include object-oriented, structural, and information modelling strategies.
- The second method/phase is the development process model which is phase-by-phase breakdown of the system's development process. Examples include the spiral model and the linear life cycle model.
- The development method which is the third phase is a strategy for carrying out at least one system development phase methodically. This involves a group of practises, ideas, plans, and tools built around a certain philosophy and system of reference.
- Development Methodology are the steps to complete a development assignment. As an illustration, create entity relationship diagrams.

3.1. Functional Component Design of The System

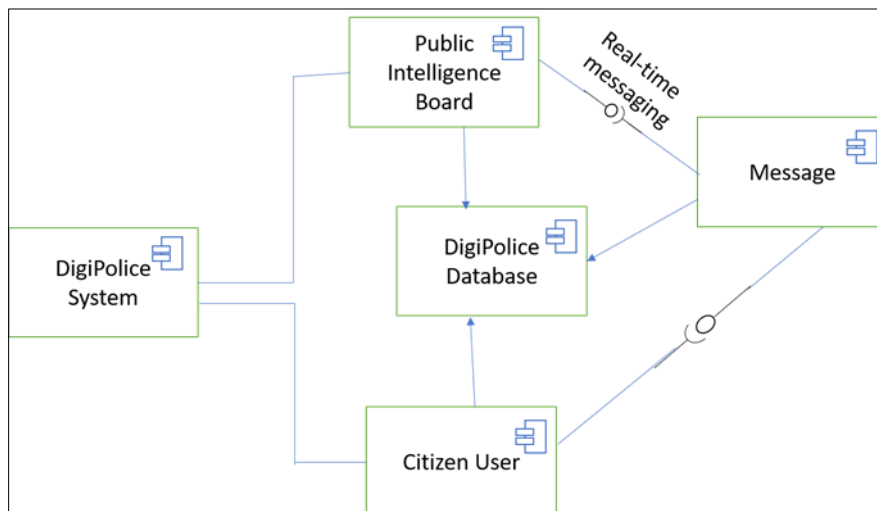


Figure 1 Component System Diagram of DigiPolice

Component design places an emphasis on building systems from reusable and interchangeable software components. In this system, it entails breaking down the DP system into smaller, functionally-complete elements known as components that may be independently created, tested, and maintained. See figure 1 for more.

An illustration of the sequential flow of choices, actions, and processes within the system or program will be represented via a flowchart diagram. The order of processes and the logical flow of data or control between distinct components or stages are represented using a variety of symbols and arrows. Using flowcharts helps simplify complex processes within the system so that they are simpler to comprehend, evaluate, and convey.

[25] stressed that flowchart diagrams are essential in software design because they show the logical organization and flow of algorithms, program control, and decision-making processes. They give engineers a high-level picture of the operation of the system, enabling them to spot any potential bottlenecks, inefficiencies, or design faults. Flowcharts also help in the documentation and dissemination of software designs to stakeholders, promoting teamwork and clear understanding.

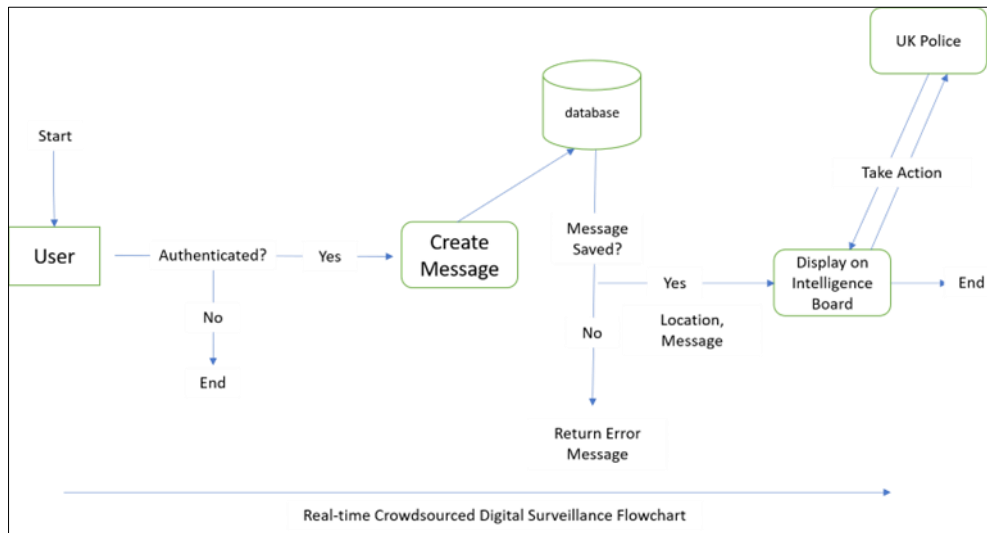


Figure 2 DigiPolice Algorithm Flowchart

3.2. Implementation

The implementation and development of a software system is covered in detail in this part, with an emphasis on assuring scientific rigor and following accepted best practices. The methodology covers every stage of the software development lifecycle, from gathering requirements and designing a solution through putting it into practice, testing it, and deploying it.

Furthermore, this can guarantee the dependability, effectiveness, and maintainability of the produced software system by using this methodology. In order to build a successful DP software system, the approach is backed by a synthesis of pertinent academic research and industry best practices. We follow this by focusing on standard implementation methodology and system design.

3.3. DigiPolice (DP) Software Development Practices

Software development practices are a collection of approaches, strategies, and best practices that are used when developing software systems. These procedures cover a range of software development activities, such as gathering requirements, designing, coding, testing, deployment, and maintenance. They seek to increase the productivity, caliber, and dependability of software development initiatives. These procedures foster iterative development, high-quality coding, and prompt delivery of software solutions. [26], [27], [28].

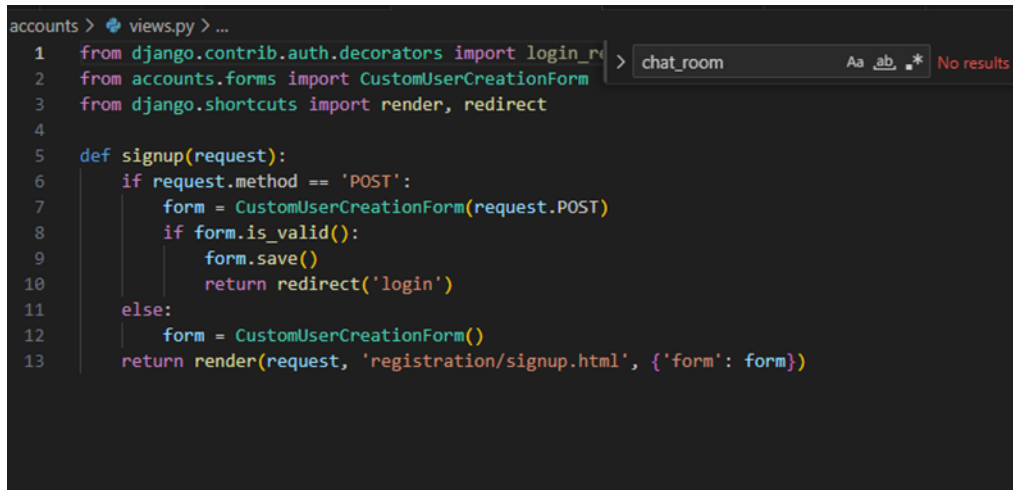
3.4. Version Control

The iterative development of the DP system required versioning using Git as a versioning tool and Github as repository tool. In order to effectively manage code changes and guarantee the integrity and traceability of software releases, version control is essential in software systems. It becomes easy to identify changes, bug fixes, and rollbacks by tracking and storing various versions of the source code. Version control systems also make cooperation easier by allowing different developers to work simultaneously on the same codebase. Conflicts are greatly reduced as a result, and code modifications can be merged effectively. Academic publications, such as those by [29] and [30] underline the value of version control systems in software development.

3.5. SOLID Principles in DP Development

A collection of recommendations known as the SOLID principles encourages modularity, adaptability, and maintainability in software architecture. Robert C. Martin first proposed these ideas, which have since acquired universal acceptance as essential ideas for developing reliable and scalable software systems. We describe some of the principles that the DICA system followed. While it may not be realistic to strictly follow the SOLID Principle depending on system approaches and use cases, some are however implementable as described below.

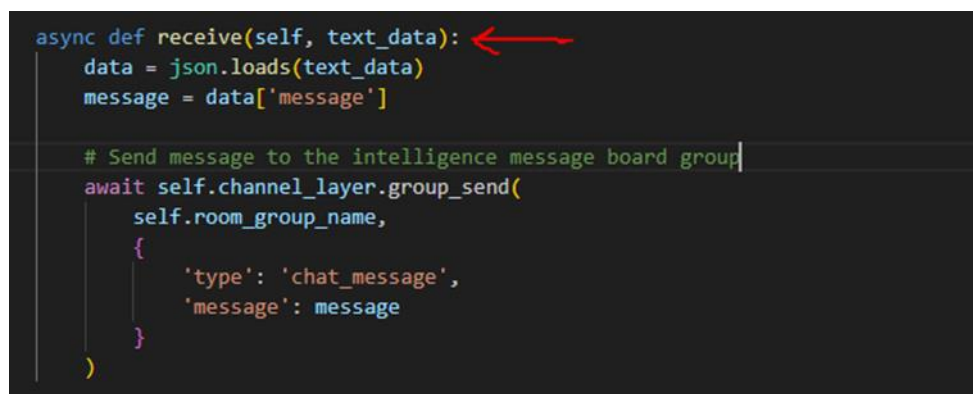
- **Single Responsibility Principle (SRP):** The SRP highlights that each class should be in charge of a single, clearly defined part of the system and that there should be no more than one cause for a class to change. Software engineers can boost code maintainability and reusability by following SRP [31]. The DP system followed the SRP principle in which each class is responsible for one function.



```
accounts > views.py > ...
1 from django.contrib.auth.decorators import login_required
2 from accounts.forms import CustomUserCreationForm
3 from django.shortcuts import render, redirect
4
5 def signup(request):
6     if request.method == 'POST':
7         form = CustomUserCreationForm(request.POST)
8         if form.is_valid():
9             form.save()
10            return redirect('login')
11        else:
12            form = CustomUserCreationForm()
13            return render(request, 'registration/signup.html', {'form': form})
```

Figure 3 DP SRP showing the user signup view with only one responsibility which is registration of potential UK citizen users

- **The Open/Closed Principle (OCP):** According to the Open/Closed Principle, software entities should be open to addition but closed to modification. This idea enables creation of systems that can easily be enhanced with new features without changing the present code. The OCP encourages code stability, reuse, and makes it simple to include new features [31]. The DP app followed this principle in which the ChatConsumerview responsible for receiving and delivering messages in real-time to the intelligence board can be further enhanced with additional multi-media features like videos and pictures which is beyond the features of the system.



```
async def receive(self, text_data):
    data = json.loads(text_data)
    message = data['message']

    # Send message to the intelligence message board group
    await self.channel_layer.group_send(
        self.room_group_name,
        {
            'type': 'chat_message',
            'message': message
        }
    )
```

Figure 4 DP OCP principle showing only *text_data* message type, additional message type like *video_data*, *audio_data* can be further added to the receive function as a parameter and processed

The remaining principles include the Liskov Substitution Principle (LSP) which emphasizes that objects of a superclass should be interchangeable with objects of their subclasses without compromising the program's validity [32]. The Clients shouldn't be made to rely on interfaces they don't utilize, according to the Interface Segregation Principle which

is the fourth principle while by placing an abstraction layer between them, the Dependency Inversion Principle which is the fifth SOLID principle encourages the decoupling of high-level modules from low-level modules.

3.6. How Message is Transmitted in Real-Time From User to Police Intelligence Board in the DP System using Django Channels' WebSocket Technology

Using WebSockets to establish real-time, bidirectional communication between web clients and servers, Django Channels is a potent Django plugin used in this system. For applications requiring real-time updates or interactive features, WebSockets offer a persistent connection that enables effective and low-latency communication. WebSockets in Django Channels as used in this system are explained in detail here:

- **Configuration:** Channels library was installed and included it in the project in order to use WebSockets with Django Channels. In order to manage WebSocket connections, routing and ASGI (Asynchronous Server Gateway Interface) configuration were required and hence configured.
- **Protocol Upgrade:** The initial handshake that happens when a client requests a WebSocket connection adheres to the HTTP protocol. In order to upgrade the connection to a WebSocket, the client (the DP system) sends an HTTP request with a particular header. When the server receives this request, it determines whether WebSocket connections are supported.

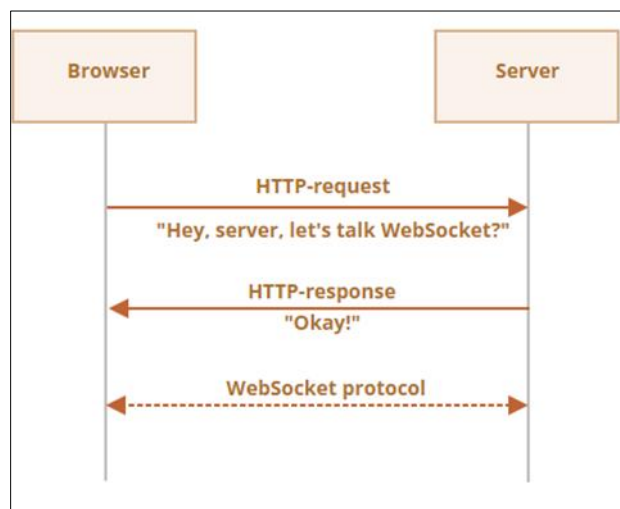


Figure 5 Websocket Handshake and Upgrade to Websocket (101) (Source: [33])

- **ASGI Application:** The Django Channels ASGI application responds to the upgrade request as soon as the server acknowledges it. By serving as a bridge between the server and the Channels layer, the ASGI application enables the asynchronous processing of requests and responses.

```

chatapp > asgi.py > ...
1  import os
2  from django.core.asgi import get_asgi_application
3  from channels.routing import ProtocolTypeRouter, URLRouter
4  from chatapp.routing import urlpatterns
5
6  os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'chatapp.settings')
7
8  application = ProtocolTypeRouter({
9      'http': get_asgi_application(),
10     'websocket': URLRouter(urlpatterns),
11 })
12
    
```

Figure 6 ASGI Application configuration in the system

- **Routing:** Which consumer manages the incoming WebSocket connection is determined by routing in Django Channels. Although consumers are made expressly for WebSocket connections, they are comparable to Django views. A given consumer is assigned a URL path by the routing setup.

```

chat > routing.py > ...
1 # chat_app/routing.py
2
3 from django.urls import re_path
4
5 from . import consumers
6
7 urlpatterns = [
8     re_path(r'ws/chat/$', consumers.ChatConsumer.as_asgi()),
9 ]
10

```

Figure 7 DP Routing connection to consumer

- **Consumers:** In Django Channels, handling WebSocket connections and controlling client-server communication falls to consumers. Consumers are able to connect to WebSockets, read and write messages, and carry out various tasks based on the information they have received.
- **The Channels Layer:** The communication between Django and the underlying WebSocket server is facilitated by the Channels layer, which serves as an intermediate. It offers a high-level API that abstracts away the difficulties of handling messages and managing WebSocket connections in the DP system.
- **Event Driven Communication/Broadcasting/Sending and Receiving Messages:** Clients and servers can send and receive messages using the WebSocket protocol once the WebSocket connection has been made. Messages may be in text, JSON, or binary format, among others. The server has the ability to send messages to clients or reply to requests from clients.

Events prompt customer behaviors in the event-driven approach used by Django Channels. Events can be the receiving of a message, the opening or closure of a WebSocket connection, or other specific events. Customers can specify event handlers to carry out particular operations in response to these events.

With Django Channels, messages can be broadcast to numerous clients or grouped together. While groups offer targeted communication with particular client subsets, broadcasts enable real-time updates to all connected clients.

```

import json
from channels.generic.websocket import AsyncWebsocketConsumer

class MessageConsumer(AsyncWebsocketConsumer):
    async def connect(self):
        self.room_group_name = 'intelligence_room'

        # Join the room group
        await self.channel_layer.group_add(
            self.room_group_name,
            self.channel_name
        )

        await self.accept()

    async def disconnect(self, close_code):
        # Leave the room group
        await self.channel_layer.group_discard(
            self.room_group_name,
            self.channel_name
        )

```

Figure 8 DP Consumer using Asynchronous connection

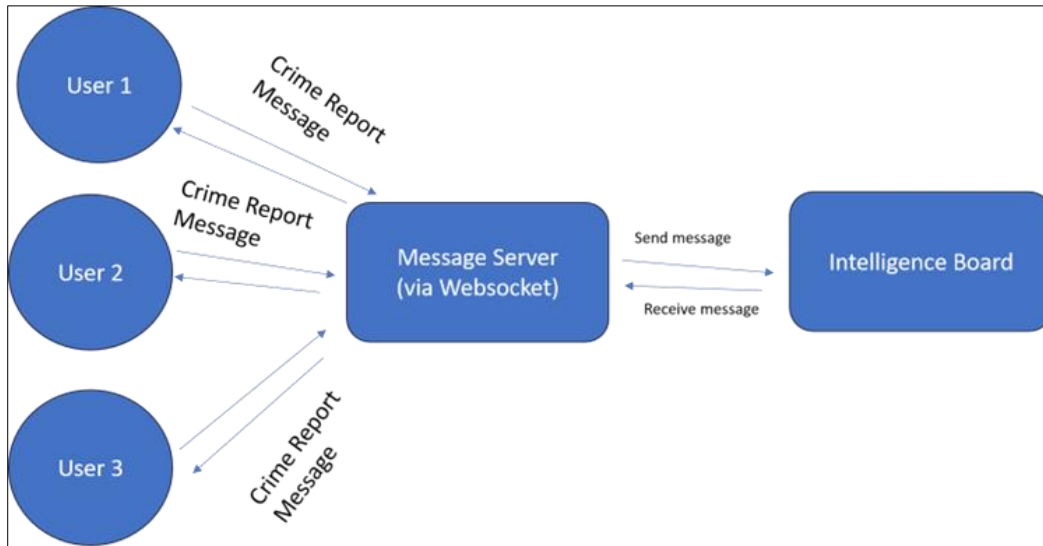


Figure 9 Real-time event driven communication in the DP web application

For local development in this work, we employ the Windows 10 operating system, a piece of software produced by Microsoft Corporation. This is due to the use's adaptability, affordability, and widespread acceptance in Nigeria. Few people utilise Apple's Mac OS, whereas the majority of people use Windows OS. This was selected to be simple to use in the DigiPolice app's offline mode. Linux was utilised as the operating system for the online version. Due to its numerous beneficial features for both corporate and residential users, Linux has substantially developed and is now one of the most popular operating systems in the IT industry [34].

3.7. User Interface

The visual and interactive components of an application's user interface (UI) are how users engage with the software. It includes the graphical elements of the application, such as buttons, menus, and forms, as well as their structure, functionality, and design. A user-friendly and intuitive experience is crucially dependent on the user interface (UI). It affects user experience, productivity, and engagement. Usability criteria, such as simplicity, consistency, and feedback systems, should be prioritized in a well-designed user interface [35]. The DP web application followed UI design best practices.

3.8. The User Login Interface

This is the user authentication interface of the application

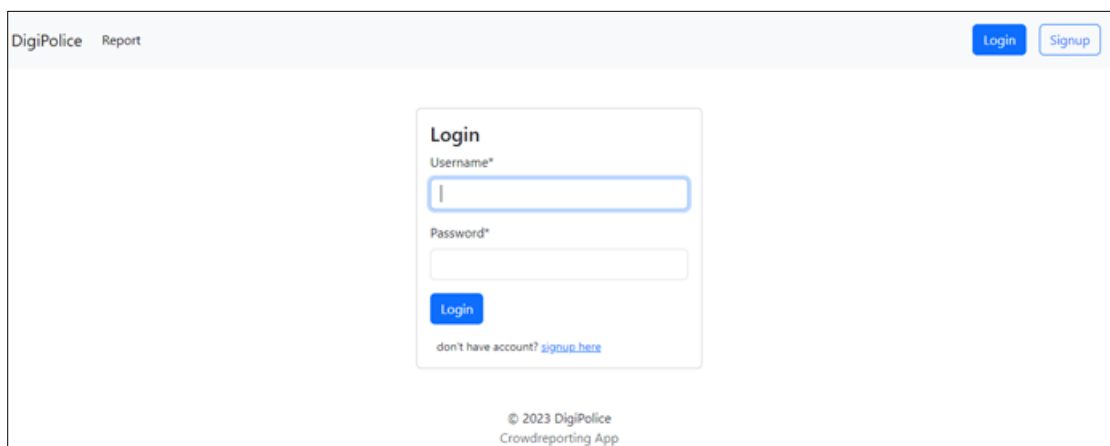


Figure 10 User Login Interface

3.9. The User Message Input Interface

This interface takes in the crime alert or message to be sent to UK police.

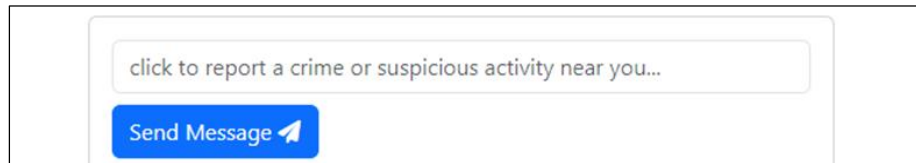


Figure 11 User input interface made simple and easy

3.10. The Intelligence Board

This interface displays all the incoming messages from multiple destinations and users presumably in the UK in realtime. This interface displays the message, the anonymized user (for purpose of safety of whistle blowers and reporters), the IP address and Location (city) for rapid response from the Police Force. This gives a very wide surveillance and efficiency to Police operations.

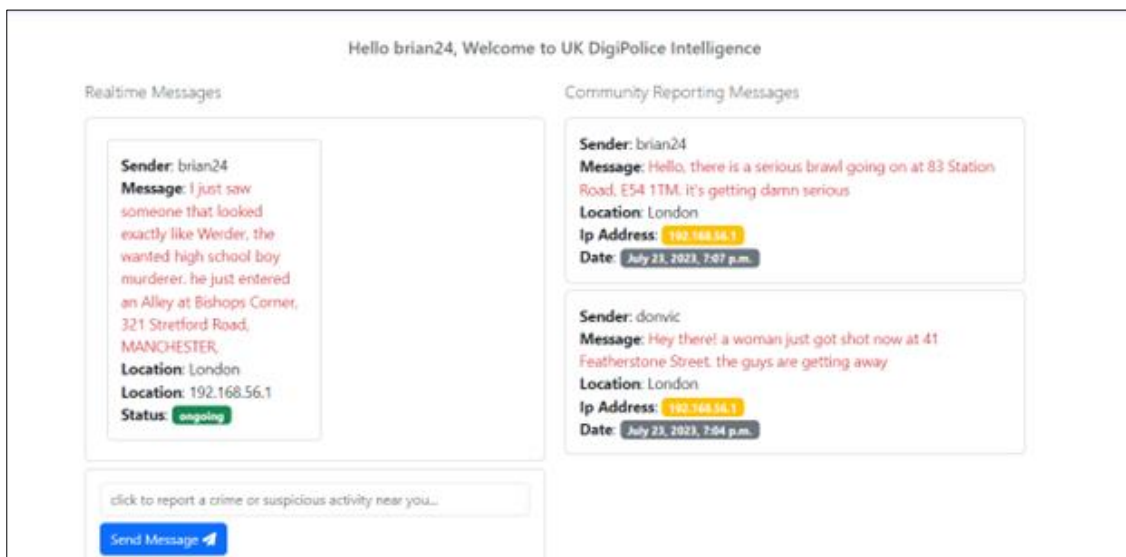


Figure 12 The Real Time Intelligence Board

3.11. Testing and Evaluation

The Pytest framework was used to create and run tests for the DP code. Pytest is a well-liked testing framework that offers a straightforward and understandable technique to build tests, enabling one to guarantee the accuracy and dependability of any software product.

Furthermore, while there are existing test packages, Pytest have lots of advantages suitable for use as a testing suite tool for DP app.

The ability of Pytest to automatically find and run tests without the need for explicit configuration is one of its important advantages. In order to find and run the test functions or methods, it recursively analyzes the test directories and files. Pytest employs a naming scheme for its test files and functions, which makes it simple to locate and execute the tests.

Additionally, Pytest offers a reliable framework for test fixtures. Fixtures are procedures or functions that give the tests a stable and well-defined state. They can be utilized to create test data, preconditions, or initialize resources needed for testing. Fixtures' lifecycles are automatically managed by Pytest, which makes sure they are properly put up and taken down before and after each test.

To assess a design that has been put into practice, researchers frequently use both quantitative and qualitative methodologies. Response time, throughput, and error rates are examples of quantitative metrics that provide unbiased

assessments of performance and functionality [36]. User perceptions and software usability can be better understood through qualitative methodologies like user surveys and expert evaluations [37].

Quantitative metrics with a special focus on performance testing and response time evaluation were focused on here. Furthermore, understanding the system objectives is useful when benchmarking them with functional which we discuss briefly.

3.12. Load Performance Test and Response Time

Given that DP app will be subjected to hundreds of users reporting in real time, the system was subjected to 50 concurrent load tests of real users swarming the app. 2 pages were chosen, the login and the signup page to evaluate load performance and response time as number of users increased. The processor speed of the host machine was 4GHz and a RAM of 4GB running on Intel's Corei5 vPro.

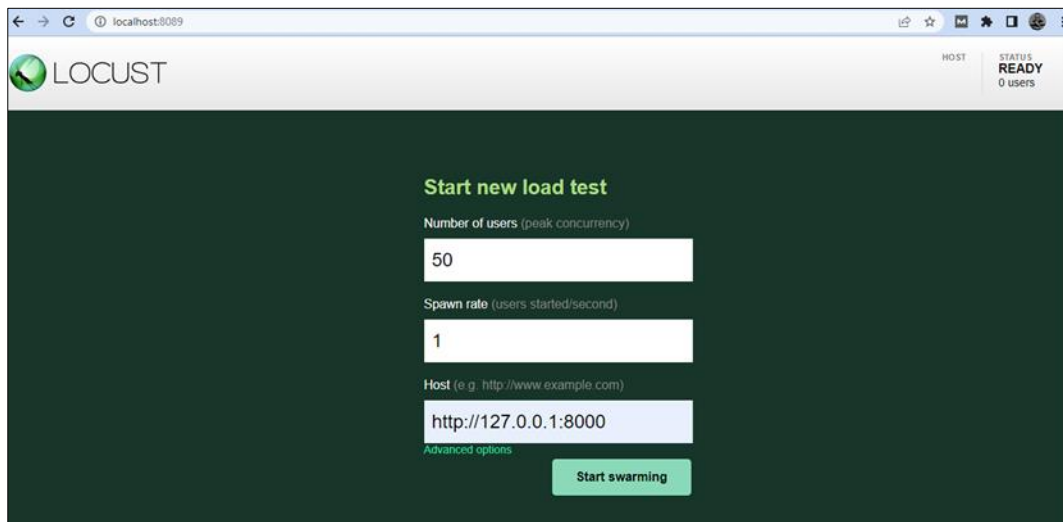


Figure 13 Performance Evaluation: 50 concurrent user load test at 1 spawn rate

From the result chart below, 50 users generated average of 29.7 requests per second without any failures. The system scaled up well up to 50 users generating and peaked at total of 2600 milliseconds (2.6 seconds to respond to user requests).

Beyond 50 users, there is a sharp increase of activities, with an increased response time. At 100 users, the system took over 7000 milliseconds (7 seconds) to respond to each request, the system could not perform optimally, response times kept fluctuating.

This shows that a higher host machine or cluster of host machines operating in front of load balancers is needed to keep response time as low as 1.5 seconds for better user experience.



Figure 14 Performance evaluation at 100 concurrent users

3.13. Error Rate of DP System

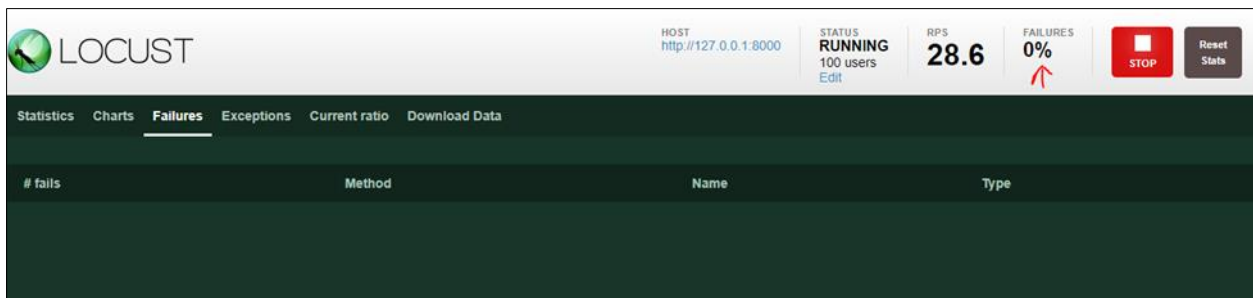


Figure 15 DP System Error rate

The system was sturdy, with no occurrence of failures or errors during 100 concurrent user performance test.

For the purpose of accounting for thousands of citizens in the UK, a cluster of higher compute power machines with 32GB RAM and faster processor speed up to 16GHz can improve performance tentatively.

3.14. System Software Version and Backward Compatibility Testing

Software versioning is a methodical process for designating distinctive identities or labels to various software versions or iterations. It makes it easier for users and developers to identify and keep track of the improvements, bug fixes, and modifications introduced in later versions. Semantic versioning (e.g., 1.0.0, 2.3.1) and date-based versioning (e.g., YYYY.MM.DD) are two common versioning techniques.

In order to verify that newer versions of software are compatible with older versions and do not damage current functionality, backward compatibility testing is a crucial component of software versioning. To ensure that all crucial features and user interfaces continue to function as intended, the most recent version must be tested against current use cases, data, and integrations [38]. Testing for backward compatibility enables users who might not be able or willing to upgrade right away avoid potential problems and guarantees a seamless transition between versions.

To effectively meet the standard objectives and ensure the maintainability and backward compatibility of DP app from initial deployment, the following software versions and backward compatibility testing were checked. The ones untested was due to the constraint of time and availability of compute power and machine resources to run web app across multiple instances.

Table 1 Requirements Version status

Requirement Version	Status
Python 3.8 (current version)	Passed
Python 3.9	Untested
Django 1.0	Failed
Django 2.1.5	Passed
Django 3.2 (current version)	Passed
Django 4 (forward compatibility)	Untested
Library Dependencies	Passed (for both Django 2, Django 3, Python 3.8)

4. Conclusion

This study shows that the DigiPolice (DP) apart from its real time non LFR solution also facilitates improved engagement between the police and the community. It provides a platform for citizens to report incidents and share information, fostering a sense of participation and collaboration in crime prevention efforts. Thereby dousing the public fears of LFR in terms of accuracy rates, the possibility of prejudice and discrimination, privacy violations and data exploitation, as well as a lack of regulation and transparency.

The DigiPolice as a software application was built on the principles of SDLC in which the components are created to follow to follow the MVC pattern. Thus in creating intuitive interface, efficient workflows, and reliable feedback systems, a prototype, user-centred design was adopted, which was planned to increase user pleasure and productivity. For intuitive user interface, clean coded frontend design system was chosen which included Bootstrap version 5 developed by Twitter as an open-source reusable design system. For efficient workflow, Django's inbuilt user authentication system was used. In order to achieve speed and quick response between user message input to public intelligence board, Django channels package which is reliably used for real-timewebsocket connections was used.

Given that DP app was subjected to hundreds of users reporting in real time, the system was subjected to 50 concurrent load tests of real users swarming the app. 2. The processor speed of the host machine was 4GHz and a RAM of 4GB running on Intel's Corei5 vPro. From the result 50 users generated average of 29.7 requests per second without any failures. The system scaled up well up to 50 users generating and peaked at total of 2600 milliseconds (2.6 seconds to respond to user requests). Beyond 50 users, there was a sharp increase of activities, with an increased response time. At 100 users, the system took over 7000 milliseconds (7 seconds) to respond to each request, the system could not perform optimally, response times kept fluctuating.

The effectiveness of the online deployment was evaluated with respectable results, and the system requirements also underwent successful testing. The appropriate investigation of ethical issues was done to meet requirements. DigiPolice has the potential to develop into a cutting-edge programme for real-time crowd surveillance and incident reporting. UK Police can employ it to as advancement to its “My Police” with as fast-track real-time and collaborative crowd reporting enhancements.

Recommendation

The created prototype system indicates possible opportunities for innovation in the area of interest of the study while also demonstrating how a gap may be filled. A prototype called DigiPolice was developed with an emphasis on the components of real-time criminal crowd reporting. However, the UK Police, particularly the Metropolitan Police, should create an efficient large-scale system from the prototype. A more dependable system architecture and model may thus be created to load more users in subsequent experiments. As an enhancement, this may also be included into current systems.

Additionally, the prototype can be extended by adding Redis server (a key-value datastore) for real time chat persistent without relying only on database and in-memory temporary session storage. Additional areas for further improvement and research could be adding multimedia like images, videos and even livestreaming. Also, data privacy in which users accounts are anonymized to protect whistleblowers.

Compliance with ethical standards

Disclosure of conflict of interest

No conflict of interest to be disclosed.

References

- [1] Onyemachi Joshua Ndubuisi, Gift Adene, Belonwu Tochukwu Sunday, Chinedu Emmanuel Mbonu, & Adannaya Uneke Gift-Adene. (2024). Digital Criminal Biometric Archives (dica) and Public Facial Recognition System (FRS) for Nigerian criminal investigation using Haar Cascades classifier technique. *World Journal of Advanced Engineering Technology and Sciences*, 11(2), 029–043. doi:10.30574/wjaets.2024.11.2.0077
- [2] Brooks R., Lopez C. 2020. Policing in a Time of Pandemic: Recommendations for Law Enforcement. <https://ethics.harvard.edu/files/center-for-ethics/files/whitepaper7a.pdf> Retrieved from. [Google Scholar]
- [3] Bright, D., Brewer, R. and Morselli, C. (2021) Using social network analysis to study crime: Navigating the challenges of criminal justice records. *Social Networks*, 66, pp.50-64.
- [4] Jasserand, C. (2023) Experiments with facial recognition technologies in public spaces: in search of an EU governance framework. *Handbook on the Politics and Governance of Big Data and Artificial Intelligence*, 1, pp.315-331
- [5] Hill, D., O'Connor, C. D. and Slane, A. (2022) Police use of facial recognition technology: The potential for engaging the public through co-constructed policy-making. *International Journal of Police Science and Management*, 24(3), pp. 325-335.
- [6] Andrejevic, M. (2019). Automating surveillance. *Surveillance and Society*, 17(1/2), pp.7-13.
- [7] Peng, Z., He, J., Yang, F. and Zhao, X. (2022) An overview and forecast of biometric recognition technology used in forensic science. In *Chinese Conference on Biometric Recognition* (pp. 32-41). Cham: Springer Nature.
- [8] Pisaric, M. (2022) Communications encryption as an investigative obstacle. *J. Crimin. and Crim. L.*, 60, pp.61-78
- [9] Petersen, K. (2022) Looking at the big picture: Using systems theory to understand the impact of body-worn cameras on police accountability. *Critical Criminology*, 30(4), pp. 861-878.
- [10] Alani, B. F. A. (2022) Anomaly detection for video surveillance in crowded environments for police (Master's thesis, Altınbaş Üniversitesi/Lisansüstü Eğitim Enstitüsü).
- [11] Dauvergne, P. (2022) Facial recognition technology for policing and surveillance in the Global South: A call for bans. *Third World Quarterly*, 43(9), pp. 2325-2335.
- [12] Slobogin, C. and Brayne, S. (2023) Surveillance technologies and constitutional law. *Annual Review of Criminology*, 6, pp. 219-240.

- [13] Brayne, S. (2020) *Predict and surveil: Data, discretion, and the future of policing*. Oxford: Oxford University Press
- [14] Piza, E. L., Welsh, B. C., Farrington, D. P. and Thomas, A. L. (2019) CCTV surveillance for crime prevention: A 40-year systematic review with meta-analysis. *Criminology and public policy*, 18(1), pp.135-159.
- [15] Loftus, B. (2019) Normalizing covert surveillance: the subterranean world of policing. *The British Journal of Sociology*, 70(5), pp. 2070-2091.
- [16] Stardust, Z., Gillett, R. and Albury, K. (2023) Surveillance does not equal safety: Police, data and consent on dating apps. *Crime, Media, Culture*, 19(2), pp. 274-295
- [17] Fontes, C., Hohma, E., Corrigan, C. C. and Lütge, C. (2022) AI-powered public surveillance systems: why we (might) need them and how we want them. *Technology in Society*, 71,.
- [18] Andrejevic, M. (2019).Automating surveillance. *Surveillance and Society*, 17(1/2), pp.7-13.
- [19] Harper, D. J., Ellis, D. and Tucker, I. (2022) Covert aspects of surveillance and the ethical issues they raise. *Ethical Issues in Covert, Security and Surveillance Research Advances in Research Ethics and Integrity*, 35(2), pp. 177-197.
- [20] Malloch, M. S. (2023) Policing controversies: Undercover policing. In *Forensic psychology, crime and policing* (pp. 263-268). New York: Policy Press.
- [21] Radiya-Dixit, E. and Neff, G. (2023) A sociotechnical audit: Assessing police use of facial recognition. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency* (pp. 1334-1346).
- [22] Fussey, P. and Murray, D. (2019) *Independent report on the London Metropolitan Police Service’s trial of live facial recognition technology*. London: MET Press
- [23] Johnson, C., Smith, A. and Williams, B. (2020) Evaluating usability in software development. In *Proceedings of the 12th International Conference on Human-Computer Interaction* (pp. 245-251).
- [24] Ratcliffe, J. (2006) *Video surveillance of public places*. Washington, DC: US Department of Justice, Office of Community Oriented Policing Services.
- [25] Sardeshpande, D. and Gokhale, V. (2022) “Legibility” a product of obligatory processes in parametric architectural design: A study of implications of associative modeling on design thinking in a parametric architectural design studio. *International Journal of Architectural Computing*, 20(4), pp. 728-741.
- [26] Jackson, M. and Green, S. (2017) A code review of software maintainability metrics. In *Proceedings of the 9th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement* (pp. 116-119).
- [27] Bright, D., Brewer, R. and Morselli, C. (2021) Using social network analysis to study crime: Navigating the challenges of criminal justice records. *Social Networks*, 66, pp.50-64.
- [28] Chetty, J. S. (2019) *An evaluation of electronic surveillance as a covert technique in the investigation of organised crime* (Doctoral dissertation).
- [29] Monahan, T. (2012) Surveillance and terrorism. *Routledge handbook of surveillance studies*, 21 pp. 285-291.
- [30] Shinguto, L. N. (2021) *An evaluation of physical surveillance in the investigation of robberies* (Doctoral dissertation).
- [31] Smith, P., Johnson, L. and Brown, R. (2018) Functional testing in software development. In *Proceedings of the 17th International Symposium on Software Engineering* (pp. 275-281).
- [32] Barbara Liskov, *Data Abstraction and Hierarchy*, ACM SIGPLAN Notices, 23(5):17-34, May 1987.
- [33] JavaScript Info. <https://javascript.info>
- [34] Borum, R. (2020) Scientific and technological advances in law enforcement intelligence analysis. *Science Informed Policing*, 56(1), pp. 99-121.
- [35] Shneiderman, B. (1998). *Designing the user interface: Strategies for effective human (3rd ed.)*. Boston, MA: Addison Wesley Longman, Inc.
- [36] Chen, L., Wang, Y. and Li, H. (2016) Quantitative evaluation of software design. In *2016 IEEE International Conference on Software Quality, Reliability and Security (QRS)* (pp. 455-461). IEEE.
- [37] Lee, J., Kim, M. and Park, S. (2018) Qualitative evaluation of software usability through user experience. In *Proceedings of the 20th International Conference on Human-Computer Interaction* (pp. 132-137).
- [38] Huang, Y. and Kuo, C. (2019) Software backward compatibility testing: State of the art and challenges. *Journal of Systems and Software*, 147, pp. 106-123.