# PROJECT REPORT

# Topic: VIRTUAL PRIVATE NETWORK (VPN)

*Operating Systems Lab*

**PROJECT TITLE:**

**BCE – 5A**

*Group Members*

| Name | Enrollment |
|---|---|
| Huzaifa Ahmed | 02-132232-014 |
| Muhammad Arslan | 02-132232-009 |
| Rana Muhammad Talha Umer | 02-132232-003 |

**Submitted to:**

**Muhammad Muzammil**

## ABSTRACT:

This project explores the design and implementation of a Virtual Private Network (VPN), a crucial technology for secure and private communication over the internet. In an era where data privacy is increasingly at risk, VPNs offer a practical solution for individuals and organizations to protect their information. This project focuses on understanding the underlying principles of VPNs, such as tunneling protocols, encryption standards, and authentication mechanisms. Through this report, we aim to demonstrate how VPNs function, their real-world applications, and how a basic VPN setup can be configured to ensure secure data transmission across potentially unsafe networks.

## INTRODUCTION:

With the rapid growth of internet usage, the need for data protection and secure online communication has become more important than ever. A Virtual Private Network (VPN) is a technology that allows users to create a secure connection to another network over the internet. VPNs are commonly used to shield browsing activity, access restricted websites, and safeguard sensitive data from cyber threats. This project was undertaken to explore how VPNs operate, their advantages, and how to set up a functional VPN system. By delving into the technical structure and practical uses of VPNs, the project enhances our understanding of cybersecurity and modern network communication.

## PROBLEM STATEMENT:

As internet use becomes increasingly widespread, so do risks related to data theft, surveillance, and unauthorized access. Many users are unaware of how exposed their online activities can be on public or unsecured networks. The main problem addressed by this project is the lack of a secure and private way to transmit information over the internet. This project aims to provide a working solution using a VPN to establish a protected channel for communication, even across potentially unsafe or public networks. It highlights the importance of encryption and security protocols in preserving user privacy and data integrity.

## METHODOLOGY:

To implement the VPN, we followed a structured approach that included both research and hands-on setup. First, we studied the theoretical concepts behind VPNs, including tunneling (PPTP, L2TP, OpenVPN), encryption (AES, RSA), and authentication methods. Next, we selected appropriate tools and software (such as OpenVPN or WireGuard) to build a working VPN prototype. The setup involved configuring server and client systems, defining secure keys, and testing connections across different networks. We also examined potential vulnerabilities and how to mitigate them. Our methodology combines theoretical learning with practical application to ensure a comprehensive understanding of how VPNs operate in real-world scenarios.

## PROJECT SCOPE:

This project focuses on building a basic, functional VPN and explaining its significance in secure communication. The scope includes researching different VPN technologies, implementing one of the protocols, and testing the system across devices and networks. While the project does not aim to create a commercial-grade VPN service, it lays the foundation for understanding how enterprise-level VPNs work. Key areas covered include encryption, tunneling, authentication, and real-world use cases. The project also discusses limitations, such as potential speed reductions and device compatibility, while suggesting areas for future improvement or expansion.

## CODE:

```
import sys import requests from bs4
import BeautifulSoup import re from
PyQt5.QtCore import QUrl, Qt from
PyQt5.QtGui import QFont, QIcon
from PyQt5.QtWidgets import QApplication, QMainWindow, QVBoxLayout, QWidget, QListWidget, QSplitter,
QHBoxLayout, QLineEdit, QPushButton, QLabel, QComboBox, QMessageBox from PyQt5.QtNetwork import
QNetworkProxy from PyQt5.QtWebEngineWidgets import QWebEngineView, QWebEngineProfile,
QWebEngineSettings import random


def fetch_fresh_proxies():
  """Fetch fresh proxy list from free-proxy-list.net"""
  try:
    response = requests.get('https://free-proxy-list.net/', timeout=10)
if response.status_code == 200:
      soup = BeautifulSoup(response.text, 'html.parser')
rows = soup.select('table.table tbody tr')


      fresh_proxies = []
for row in rows:
        cells = row.find_all('td')
if len(cells) >= 7:
          ip = cells[0].text.strip()
port = cells[1].text.strip()
https = cells[6].text.strip()


          if https.lower() == 'yes':
            proxy_url = f"https://{ip}:{port}"
          else:
            proxy_url = f"http://{ip}:{port}"
```

```
            fresh_proxies.append(proxy_url)

        return fresh_proxies
except Exception as e:
    print(f"Error fetching proxies: {e}")

  return []


proxies = [
  "67.43.236.18",  # Default proxy
  # "http://51.158.123.35:8811",
  "http://167.99.232.31:3128",
  "http://134.209.29.120:3128"
]


fresh_proxies = fetch_fresh_proxies() if
fresh_proxies:
   proxies = fresh_proxies    print(f"Successfully fetched
{len(proxies)} fresh proxies") else:    print("Using default
proxy list")

class Browser(QMainWindow):
   def _init_(self):
super()._init_()
self.setWindowTitle("Python
VPN")
```

```
self.setGeometry(100, 100,
1400, 900)


    self.setStyleSheet("""
QMainWindow {           background-
color: #121212;
        }
        QPushButton {
background-color: #2C2C2C;
color: #E0E0E0;          border:
none;          padding: 8px 16px;
border-radius: 6px;          font-
weight: bold;          font-size:
12px;
        }
        QPushButton:hover {
 background-color: #3D3D3D;
        }
        QPushButton:pressed {
 background-color: #505050;
        }
        QLineEdit {          padding:
 8px 12px;          border: 2px solid
 #2C2C2C;          border-radius:
 6px;          font-size: 14px;
 background-color: #1E1E1E;
 color: #E0E0E0;
        }
        QLineEdit:focus {
 border-color: #4CAF50;
```

```
        }
        QListWidget {
background-color: #1E1E1E;
border: 1px solid #333;
border-radius: 6px;
padding: 5px;            font-size:
12px;            color: #E0E0E0;
        }
        QListWidget::item {
padding: 5px;            border-bottom:
1px solid #333;
        }
        QListWidget::item:selected {
background-color: #2C2C2C;            color:
#4CAF50;
        }
        QListWidget::item:hover {
background-color: #252525;
        }
        QLabel {
color: #E0E0E0;
font-weight: bold;
font-size: 14px;
        }
    """)


    nav_bar = QWidget() nav_bar.setStyleSheet("""
        QWidget {            background-
 color: #1E1E1E;            border-
```

```
bottom: 2px solid #333;
padding: 10px;
        }
""")
    nav_layout = QHBoxLayout()
nav_layout.setSpacing(10)
nav_layout.setContentsMargins(15, 10, 15, 10)


    nav_btn_group = QWidget()
nav_btn_layout = QHBoxLayout()
nav_btn_layout.setSpacing(5)
nav_btn_layout.setContentsMargins(0, 0, 0, 0)


    title_label = QLabel("🌐 VPN Browser")       title_label.setStyleSheet("color: #4CAF50; font-size:
16px; font-weight: bold; margin-right: 20px;")


    self.prev_btn = QPushButton("← Back")
self.prev_btn.setMaximumWidth(80)
self.prev_btn.clicked.connect(self.go_back)


    self.next_btn = QPushButton("Forward →")
    self.next_btn.setMaximumWidth(100)
    self.next_btn.clicked.connect(self.go_forward)
    self.refresh_btn = QPushButton("🔄 Refresh")
    self.refresh_btn.setMaximumWidth(90)
    self.refresh_btn.clicked.connect(self.refresh_page)
```

```
        nav_btn_layout.addWidget(self.prev_btn)
    nav_btn_layout.addWidget(self.next_btn)
    nav_btn_layout.addWidget(self.refresh_btn)
    nav_btn_group.setLayout(nav_btn_layout)


        url_group = QWidget()       url_layout =
    QHBoxLayout()       url_layout.setSpacing(5)
    url_layout.setContentsMargins(0, 0, 0, 0)


        self.url_entry = QLineEdit()       self.url_entry.setPlaceholderText("🔍 Enter
    URL here (e.g., google.com)...")
    self.url_entry.returnPressed.connect(self.navigate_to_url)
    self.url_entry.setMinimumHeight(35)


        self.go_btn = QPushButton("🚀 Go")
    self.go_btn.setMaximumWidth(60)
    self.go_btn.clicked.connect(self.navigate_to_url)

        url_layout.addWidget(self.url_entry)
    url_layout.addWidget(self.go_btn)
    url_group.setLayout(url_layout)


        status_group = QWidget()       status_layout
    = QHBoxLayout()
    status_layout.setContentsMargins(0, 0, 0, 0)
```

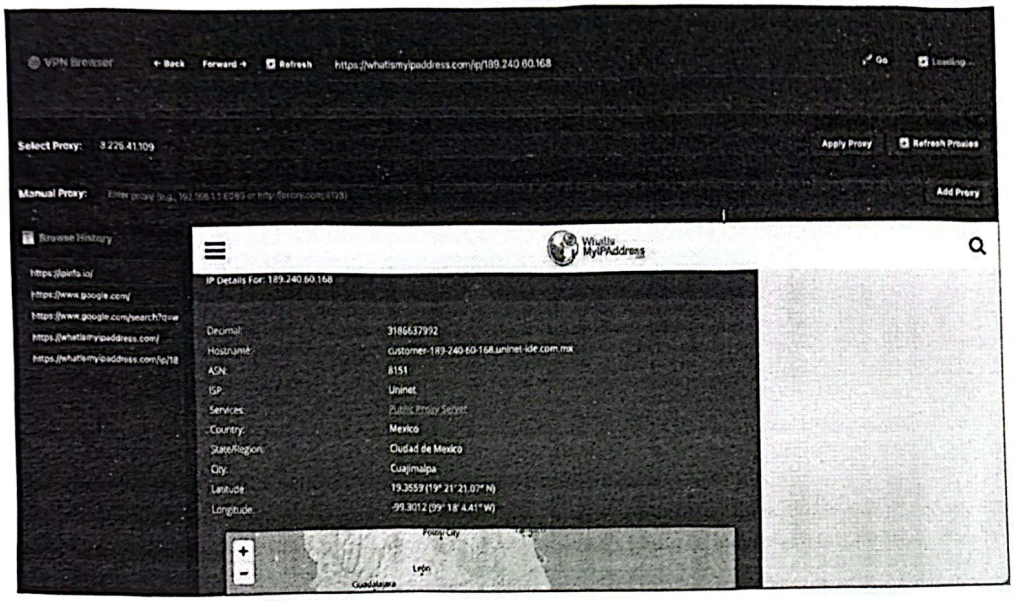
```
        self.proxy_label = QLabel("🔒 Proxy: Active")
    self.proxy_label.setStyleSheet("color: #FF6B35; font-size: 12px;")

        status_layout.addWidget(self.proxy_label)
    status_group.setLayout(status_layout)


        nav_layout.addWidget(title_label)
    nav_layout.addWidget(nav_btn_group)
    nav_layout.addWidget(url_group, 1)
    nav_layout.addWidget(status_group)
    nav_bar.setLayout(nav_layout)

        splitter = QSplitter()
    splitter.setOrientation(Qt.Horizontal)

        history_widget = QWidget()
    history_widget.setMaximumWidth(250)
    history_layout = QVBoxLayout()
    history_layout.setContentsMargins(10, 10, 10, 10)

        history_title = QLabel("📜 Browse History")
        history_title.setStyleSheet("color: #4CAF50; font-size:
        14px; font-weight: bold; margin-bottom: 10px;")

        self.history_list = QListWidget()
    self.history_list.itemClicked.connect(self.load_from_history)
```

```
        history_layout.addWidget(history_title)
    history_layout.addWidget(self.history_list)
    history_widget.setLayout(history_layout)

        self.web_view = QWebEngineView()
    self.web_view.setStyleSheet("border: 1px solid #333; border-radius: 6px;")
    self.web_view.urlChanged.connect(self.add_to_history)
    self.web_view.urlChanged.connect(self.update_url_entry)
    self.web_view.loadStarted.connect(self.on_load_started)
    self.web_view.loadFinished.connect(self.on_load_finished)

        splitter.addWidget(history_widget)
    splitter.addWidget(self.web_view)
    splitter.setSizes([250, 1150])

        proxy_selector_group = QWidget()
    proxy_selector_layout = QHBoxLayout()
    proxy_selector_layout.setContentsMargins(10, 5, 10, 5)

        proxy_label = QLabel("Select Proxy:")
    proxy_label.setStyleSheet("margin-right: 10px;")

        self.proxy_selector = QComboBox()
    self.proxy_selector.setStyleSheet("""
    QComboBox {             background-
    color: #1E1E1E;             color: #E0E0E0;
    padding: 5px;           border: 1px solid
    #333;           border-radius: 4px;
    min-width: 200px;
            }
```

```
        QComboBox::drop-down {
          border: 0px;
        }
        QComboBox::down-arrow {
image: url(down-arrow.png);
width: 12px;              height: 12px;
        }
        QComboBox QAbstractItemView {
background-color: #1E1E1E;              color:
#E0E0E0;              selection-background-color:
#4CAF50;
        }
      """)

      self.proxy_selector.addItem("No Proxy")
 for proxy in proxies:
          self.proxy_selector.addItem(proxy)
 apply_proxy_btn = QPushButton("Apply Proxy")
 apply_proxy_btn.clicked.connect(self.apply_proxy)

                    refresh_proxies_btn = QPushButton("🔄 Refresh Proxies")
                    refresh_proxies_btn.clicked.connect(self.refresh_proxies)
                        proxy_selector_layout.addWidget(proxy_label)
                    proxy_selector_layout.addWidget(self.proxy_selector, 1)
                      proxy_selector_layout.addWidget(apply_proxy_btn)
                     proxy_selector_layout.addWidget(refresh_proxies_btn)

      # Manual proxy entry section
  manual_proxy_layout = QHBoxLayout()
  manual_proxy_layout.setContentsMargins(10, 5, 10, 5)
```

```
    manual_proxy_label = QLabel("Manual Proxy:")       manual_proxy_label.setStyleSheet("margin-
right: 10px;")

    self.manual_proxy_entry = QLineEdit()       self.manual_proxy_entry.setPlaceholderText("Enter proxy (e.g.,
192.168.1.1:8080 or http://proxy.com:3128)")       self.manual_proxy_entry.setStyleSheet("""
        QLineEdit {             padding:
5px 8px;             border: 1px solid
#333;             border-radius: 4px;
font-size: 12px;
background-color: #1E1E1E;
color: #E0E0E0;             min-width:
250px;
        }
```

**OUTPUT:**



## FUTURE DEVELOPMENT:

While this project successfully demonstrates the core functionality of a basic VPN system, there are several areas for future enhancement and development. One direction is the integration of advanced VPN protocols like WireGuard, which offer improved speed and modern cryptography. Additionally, implementing multi-platform compatibility (Windows, Linux, Android, iOS) would expand usability. Future work could also involve integrating firewall and intrusion detection features to further enhance security. A user-friendly graphical interface for VPN setup and management could make the system more accessible to non-technical users. Exploring mesh VPN networks or combining VPN with other technologies like Tor could also broaden the privacy and security benefits.

## CONCLUSION:

The successful implementation and testing of a basic VPN system through this project has provided valuable insights into the world of secure networking. We have explored both the theoretical and practical aspects of VPN technology, including encryption, tunneling protocols, and user authentication. The project highlighted the importance of privacy in the digital age and demonstrated how VPNs serve as a reliable tool to protect data from cyber threats, especially on unsecured networks. While our setup was foundational, it has opened the door to more advanced applications and configurations in the future.