NETAJI SUBHAS UNIVERSITY OF TECHNOLOGY

COMPUTER HARDWARE SOFTWARE WORKSHOP

(COCSC19)

Project Report on Women Safety System

Submitted By :

2021UCS1692, 2021UCS1701, 2021UCS1698, 2021UCS1676 (CSE-III)

Submitted to :

Prof. Anand Gupta and Ekta Mam

Problem Statement

The reason we need a women's safety system is because many women face a lot of harassment, violence, and threats in different parts of their lives. Statistics show that women experience these dangers much more often than men do. Women often deal with things like catcalling, groping, sexual assault, domestic violence, and other types of violence. This makes them feel scared and unsafe in places like public areas, workplaces, and even their own homes.

Adding to the problem, society often blames the victim and discourages women from reporting these incidents. Many women are afraid they won't be believed or worry about facing retaliation if they speak up. This means a lot of incidents go unreported, and the people responsible don't face consequences for their actions. Also, women who belong to marginalized groups, like those who are poor, minorities or have disabilities, face even more discrimination and are at higher risk of violence.

Introduction

The primary aim of the Women's Safety System is to provide security and protection for women in various situations. This specialized system incorporates advanced technologies such as GPS to enable real-time tracking of the user's location, ensuring constant monitoring of movements and activities.

The Women's Safety System discreetly sends location data to a monitoring unit, allowing for immediate assistance in case of emergencies. In instances of danger, such as harassment or violence, the system can alert authorities or trusted contacts with precise GPS coordinates, enabling swift response and support.

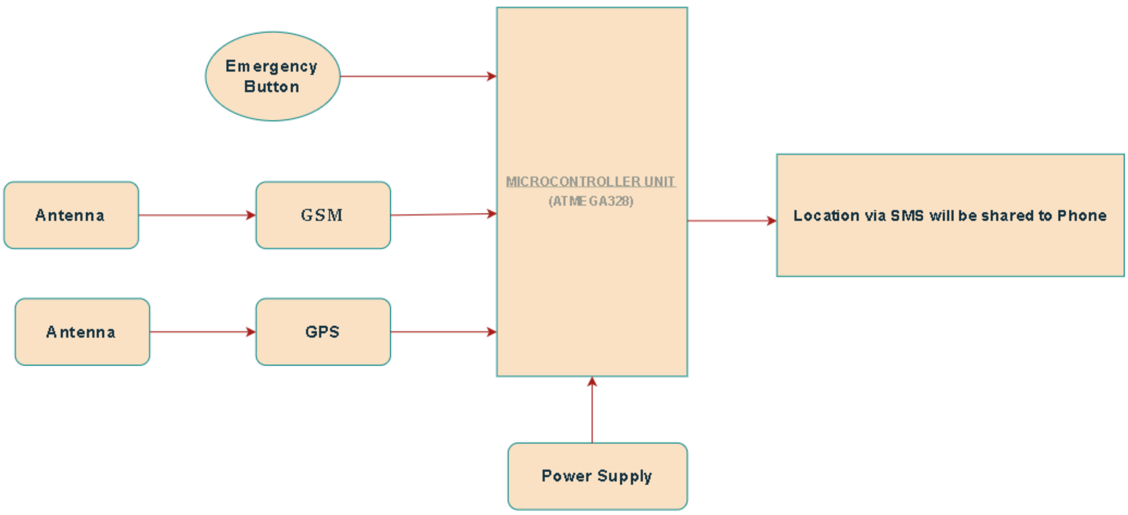
Moreover, the Women's Safety System includes features tailored to address the safety concerns of women, such as panic buttons or gesture-based activation mechanisms for discreet alerts. Additionally, sensors can detect potential threats, similar to how vehicle tracking systems can detect unauthorized movements.

The system also offers functionalities such as real-time communication with emergency services, location sharing with trusted contacts, and live tracking on mapping platforms, akin to the capabilities of vehicle tracking systems. Overall, the Women's Safety System serves as a crucial tool in empowering women to navigate their surroundings confidently and securely, offering proactive solutions to address the unique safety challenges they face.

Methodology

The Women's Safety System comprises a GPS receiver, GSM modem, and microcontroller, which are installed on the user's person or belongings. The system is designed to ensure the safety of women in various situations.
In the Women's Safety System, if the woman feels unsafe or encounters a threatening situation, she can activate the system. The GPS receiver tracks the user's location and sends the coordinates to the GSM module. Through the GSM module, the location will be shared via SMS to helping agency or police.
Overall, the Women's Safety System offers a comprehensive solution for ensuring the security and well-being of women, providing them with the means to seek assistance and alert authorities in times of need.

Block Diagram:



1. **Arduino Nano Microcontroller (MCU):** The MCU acts as the central hub of the system, orchestrating the functions of all components seamlessly.

2. **GSM SIM800L Module:** This module facilitates communication via the GSM network, enabling the exchange of SMS messages. It empowers the system to transmit alerts and receive instructions from users or the monitoring station effectively.

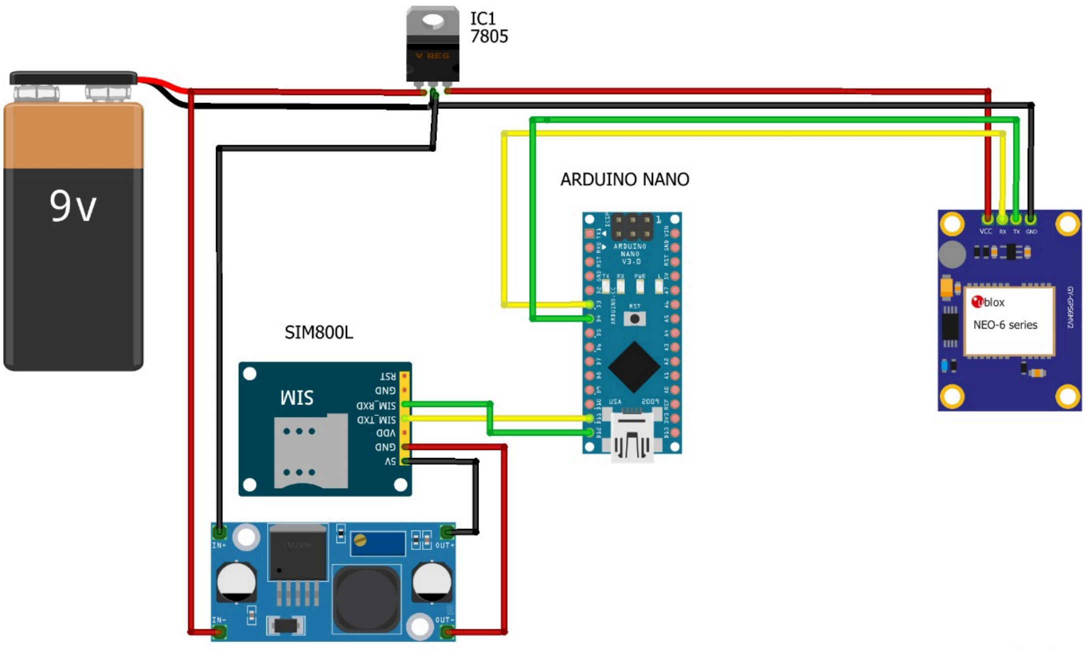
3. **GPS NEO-6M Module:** The GPS module intercepts signals from satellites, deciphering the user's precise location in terms of latitude and longitude

coordinates. This data is pivotal for furnishing accurate location information during emergencies.

4. **Mobile Phone (for SMS):** A mobile phone serves as the conduit for sending and receiving SMS messages. Linked to the GSM module, it facilitates the transmission of alerts to predetermined contacts or authorities.

5. **Help Button:** The help button acts as the primary interface for activating the system during emergencies. Depressing it triggers the transmission of an alert message along with the user's location coordinates.

Circuit:



Code:

Data Preprocessing:

- Data preprocessing involves preparing the dataset for clustering.
- Common preprocessing steps may include handling missing values, scaling features, or encoding categorical variables.
- In spatial clustering, specific preprocessing steps may not be required if the data is already in latitude and longitude format.

Model/Algorithm:

- The KMeans algorithm is chosen for spatial clustering due to its simplicity and effectiveness.
- KMeans partitions the data into 'k' clusters by iteratively assigning data points to the nearest centroid and updating centroids based on the mean of points in each cluster.

Training the Model:

- The KMeans model is trained on the dataset.
- During training, the algorithm iteratively optimizes cluster centroids to minimize the within-cluster sum of squares (WCSS) or inertia.

Parameter Tuning:

- The number of clusters 'k' is a crucial parameter in KMeans.
- Parameter tuning involves selecting the optimal value of 'k' that best captures the underlying structure of the data.
- Common techniques for determining 'k' include the Elbow Method, silhouette score, or domain knowledge.

Prediction:

- After training, the trained model is used to predict cluster labels for each data point.
- Cluster labels represent the cluster to which each data point belongs based on its proximity to the cluster centroids.

Python code:

\+ Code + Text

```
[1] import pandas as pd
    from sklearn.cluster import KMeans
    import matplotlib.pyplot as plt
    from mpl_toolkits.mplot3d import Axes3D
```

```
[4] # Load your dataset (replace 'data.csv' with your file path)
    data = pd.read_csv('data.csv')
```

```
# Select features (latitude and longitude)
X = data[['Latitude', 'Longitude']]
```

\+ Code + Text

```
# Model/Algorithm
# Initialize KMeans clustering with a default number of clusters
model = KMeans(random_state=42)
```

```
[7] # Training the Model
    model.fit(X)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default v
  warnings.warn(
```

```
KMeans
KMeans(random_state=42)
```

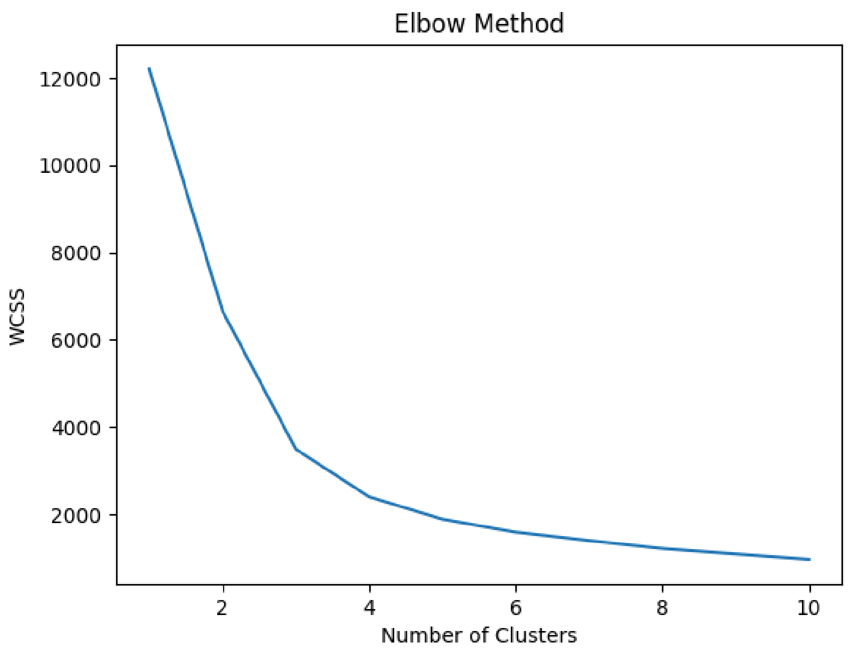
\+ Code + Text

```
# Parameter Tuning
# Determine the optimal number of clusters using the Elbow Method
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default \
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default \
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default \
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default \
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default \
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default \
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default \
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default \
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default \
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default \
  warnings.warn(
```

+ Code + Text

```
# Plot the Elbow Method graph
plt.plot(range(1, 11), wcss)
plt.title('Elbow Method')
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.show()
```




```
# From the Elbow Method, determine the optimal number of clusters and retrain the model
optimal_clusters = 3  # Adjust based on the Elbow Method graph
model = KMeans(n_clusters=optimal_clusters, random_state=42)
model.fit(X)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default v
  warnings.warn(
KMeans
KMeans(n_clusters=3, random_state=42)
```

```
[11] # Prediction
     # Predict cluster labels for each data point
     cluster_labels = model.predict(X)
```

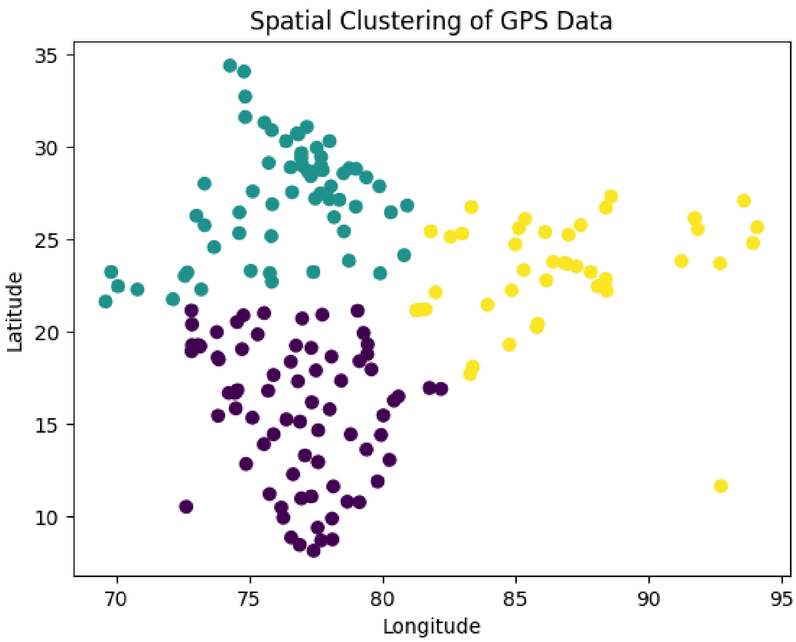
[19]
```
from sklearn.metrics import silhouette_score
```

[20]
```
# Calculate silhouette score
silhouette_avg = silhouette_score(X, cluster_labels)

# Calculate Within-Cluster Sum of Squares (WCSS)
wcss = model.inertia_
```

[21]
```
print("Silhouette Score:", silhouette_avg)
print("Within-Cluster Sum of Squares (WCSS):", wcss)
```

```
Silhouette Score: 0.5017158871041542
Within-Cluster Sum of Squares (WCSS): 3488.3951762689303
```

```
# Result Visualization
# Visualize the clusters on a 2D plot
plt.scatter(X['Longitude'], X['Latitude'], c=cluster_labels, cmap='viridis')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.title('Spatial Clustering of GPS Data')
plt.show()
```



Arduino Code:

```
#include <SoftwareSerial.h>
#include <TinyGPS++.h>

// GSM module setup
SoftwareSerial gsmSerial(12, 11);  // RX, TX pins

String phoneNumber = "+919310369517";  // Replace with the desired phone number

// GPS module setup
SoftwareSerial gpsSerial(3, 4);  // RX, TX pins

unsigned long lastTime = 0;
unsigned long timerDelay = 2000;
TinyGPSPlus gps;

// SOS button setup
const int sosButtonPin = A5;

void setup()
{
  Serial.begin(9600);
  gsmSerial.begin(9600);
  gpsSerial.begin(9600);
  pinMode(sosButtonPin, INPUT_PULLUP);
}
```

```
void loop()
{
  while (gpsSerial.available() > 0)
  {
    if (gps.encode(gpsSerial.read()))
    {
      handleGPS();
      Serial.print("Inside loop...");
    }
  }

  delay(200);
}
```

```
void handleGPS()
{
  if (gps.location.isValid())
  {  Serial.print("Valid Location---");

      float latitude = gps.location.lat();
      float longitude = gps.location.lng();
      Serial.print("Latitude= ");
      Serial.println(latitude, 6);
      Serial.print(", Longitude=");
      Serial.println(longitude, 6);

    if (digitalRead(sosButtonPin) == LOW)
    {
      String message = "Emergency Alert!\r\n";
      message += "Latitude: " + String(latitude, 6) + "\r\n";
      message += "Longitude: " + String(longitude, 6) + "\r\n";
      message += "Location URL: http://maps.google.com/maps?q=loc:" + String(latitude, 6) + "," + String(l

      sendSMS(phoneNumber, message);
    }
  }
```

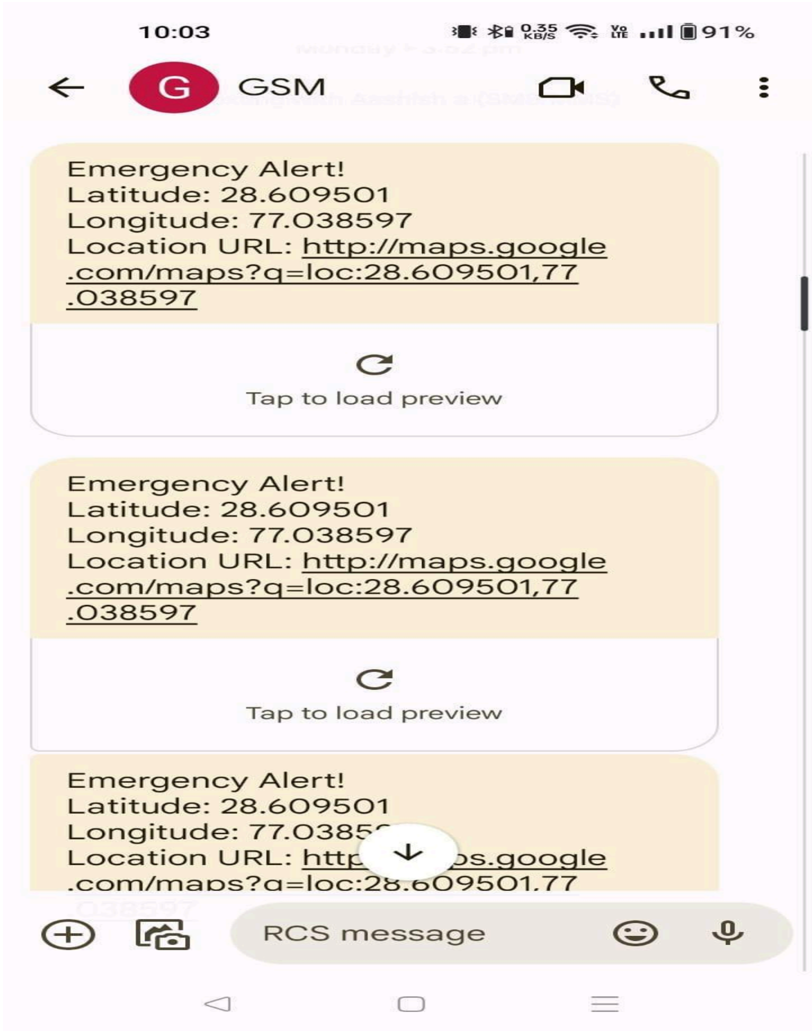
```
    else
    {
        Serial.println("Invalid GPS Data");

        Serial.println("Location: Not Available");

      if (digitalRead(sosButtonPin) == LOW)
      {
        float latitude=28.6095;
        float longitude=77.0386;
        Serial.print("SOS PRESSED!!!");
        Serial.print("Latitude= ");
        Serial.println(latitude, 6);
        Serial.print(", Longitude=");
        Serial.println(longitude, 6);

        String message = "Emergency Alert!\r\n";
        message += "Latitude: " + String(latitude, 6) + "\r\n";
        message += "Longitude: " + String(longitude, 6) + "\r\n";
        message += "Location URL: http://maps.google.com/maps?q=loc:" + String(latitude, 6) + "," + String(

        sendSMS(phoneNumber, message);
      }
    }
  }
```

```
void sendSMS(String number, String message)
{
  gsmSerial.print("AT+CMGF=1\r");
  delay(1000);
  gsmSerial.print("AT+CMGS=\"" + number + "\"\r");
  delay(1000);
  gsmSerial.print(message);
}
```

Result:

Upon pressing the emergency button integrated into the system, the Arduino Nano microcontroller detects the signal and initiates the execution of predefined algorithms. These algorithms activate the GPS module to acquire accurate location coordinates of the user's current position.

Once the GPS data is obtained, the system formulates an emergency alert message containing the GPS latitude and longitude coordinates. This message is then transmitted via the SIM800L GSM module to predefined contacts, such as emergency contacts or designated authorities.



Conclusion:

In summary, the integration of Arduino, GPS, GSM, and K-means clustering within the women's safety system offers a robust solution to address the urgent need for enhancing women's security and well-being across diverse settings. By harnessing these technologies, the system provides essential functionalities such as real-time tracking and discreet communication, enabling swift assistance during emergencies. Moreover, the incorporation of K-means clustering introduces a novel aspect to the system, allowing for the analysis of location data to pinpoint potential danger zones and hazardous areas. Through this spatial clustering approach, the system proactively alerts users to avoid risky locations and navigate safer paths. Overall, the women's safety system with Arduino, GPS, GSM, and K-means clustering embodies a holistic approach to bolstering women's safety. By amalgamating real-time tracking, seamless communication, and data-driven analysis, the system empowers women to traverse their surroundings with confidence while effectively mitigating potential risks. This comprehensive solution underscores a commitment to fostering a safer and more secure environment for women, thereby promoting their autonomy and well-being.