Smart Traffic Signal Optimization

Data Collection and Modeling

Task : Design the data structure for collecting live traffic sensor (num_vehicles, speeds) readings from different intersections in city.

Algorithm Design

```
public class TrafficData {
    private String intersectionId;
    private LocalDateTime timestamp;
    private int vehicleCount;
    private double averageSpeed;
    private int pedestrianCount;
    private double occupancyRate;


    // Constructor, getters, and setters
}
```

Goal: These algorithms are used to process the acquired data and hence changing traffic signals function accordingly, in realtime; Incorporate things like level of traffic, lines at a signalised intersection or other interference points from vehicles arriving on cross streets during each cycle, peak hour movements and pedestrian strategies in your solution.

Algorithm Pseudocode:

```
function optimizeSignalTimings(intersectionData):
    for each intersection in intersectionData:
        vehicleCount = intersection.vehicleCount
        averageSpeed = intersection.averageSpeed
        pedestrianCount = intersection.pedestrianCount
        occupancyRate = intersection.occupancyRate
        if vehicleCount > threshold or occupancyRate > threshold:
            extend green light duration
        else if pedestrianCount > threshold:
```

```
        activate pedestrian crossing signal
    else:
        maintain default signal timings
    adjustSignalTimings(intersection.id, newTimings)
```

Implementation

Challenge:

Development of a Java application integrates with the sensors installed around the city and implements adaptive traffic light control system only for limited junctions. Make sure the application can modify signal timings on-the-fly to adapt with fluctuating traffic patterns in allowing optimal flow.

Visualization and Reporting

Build visualizations to monitor real-time traffic status and timings of the signal; Produce reports of traffic flow improvements, wait reduction based on averages and congestion reductions.

Visualization Tools:

Use a library like D3. is or Chart. Mongoose and JS to display real time visualizations.

Generate detailed reports using a reporting tool such as Jasper Reports.

Example Visualization:

```
<!-- HTML code for real-time traffic monitoring dashboard -->
<div id="traffic-dashboard">
  <!-- Real-time traffic condition visualization -->
  <canvas id="trafficChart"></canvas>
</div>


<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
<script>
  var ctx = document.getElementById('trafficChart').getContext('2d');
  var trafficChart = new Chart(ctx, {
    type: 'line',
    data: {
      labels: [], // Time labels
      datasets: [{
```

```
                label: 'Vehicle Count',
                data: [], // Real-time vehicle count data
                borderColor: 'rgba(75, 192, 192, 1)',
                borderWidth: 1
            }]
        },
        options: {
            scales: {
                x: { type: 'time' },
                y: { beginAtZero: true }
            }
        }
    });

    function updateTrafficData(newData) {
        // Code to update traffic data and refresh the chart
    }
</script>
```

User Interaction

Task:

Interface meant for traffic managers who can view the updated signal timings and able to control them manually if needed. To display performance metrics and historic data for city officials (dashboard) UI Design: Traffic Manager Interface Real-time dashboard to monitor traffic conditions and control signals. Meto Dashboard Official: Data River details, metrics & traffic improvements in form of chart.

Example User Interface Layout:

```
<!DOCTYPE html>
<html>
<head>
    <title>Smart Traffic Signal Management</title>
    <link rel="stylesheet" type="text/css" href="styles.css">
</head>
<body>
    <div id="traffic-manager-interface">
        <h1>Traffic Manager Dashboard</h1>
        <div id="real-time-monitoring">
            <canvas id="trafficChart"></canvas>
        </div>
        <div id="manual-control">
            <button onclick="adjustSignal('intersection1', 'extendGreen')">Extend Green</button>
            <button onclick="adjustSignal('intersection1', 'activatePedestrian')">Activate Pedestrian</button>
        </div>
    </div>
    <div id="city-official-dashboard">
        <h1>City Official Dashboard</h1>
        <div id="performance-metrics">
            <!-- Metrics visualization -->
        </div>
    </div>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
    <script src="app.js"></script>
</body>
</html>
```

User Interaction

Task: Develop a user interface that is used by traffic managers to track and adjust signal timings manually if necessary. Dashboard for City Official to view performance Metrics and Historical Data. UI Design: Traffic Manager Interface - Real-time dashboard for observing traffic situations and controlling signals. Historical data, performance metrics and visualizations of traffic improvements City Official Dashboard Deliverables

1**. Data Flow Diagram** - A representation of how real-time data flows to control traffic lights so that they may be timed using this particular information.

2. **Theorem4:** Write pseudocode and Java implementations for the algorithms to optimize traffic signals, manage intersections.

3. **Documentation:** This includes the design decisions for algorithms, data structures you used to make sure it is processed efficiently; assumptions (such as sensor reliability); and what more can be done for improvement=end optimal performance.

4. **User Interface**: Develop intuitive and informative interfaces for traffic managers and city officials to interact with the system, monitor traffic conditions, and manage signal timing.

5. **Testing**: Include comprehensive test cases to validate the functionality and effectiveness of the traffic signal optimization system under various traffic scenarios and conditions.