PyQt5 Automotive Exam: Automotive Fault Detection System

Exercise: Create an Automotive Fault Detection System

Create a PyQt5 application that simulates a fault detection system for a car. The application should monitor **engine temperature**, **oil pressure**, and **battery voltage** in real-time. If any of these metrics exceed their safe limits, the application should display a fault message and log the fault to a file.

Requirements and Questions

Question 1: Create the Main Window (3 Marks)

- Create a ‘QMainWindow‘ with a fixed size of 800x600 pixels.
- Add three ‘QLabel‘ widgets to display the current values of engine temperature, oil pressure, and battery voltage.
- Add a ‘QStatusBar‘ to display the current status (e.g., ”Monitoring”, ”Fault Detected”).

Question 2: Implement the Fault Detection Thread (5 Marks)

- Create a ‘QThread‘ subclass named ‘FaultDetectionThread‘ to simulate real-time monitoring of engine temperature, oil pressure, and battery voltage.
- Use a ‘QTimer‘ inside the thread to generate data at regular intervals (e.g., every 500 milliseconds).
- Emit custom signals to send the generated data to the main UI.

Question 3: Update the UI Dynamically (4 Marks)

- Connect the custom signals from ‘FaultDetectionThread‘ to slots in the main window to update the ‘QLabel‘ widgets.
- Ensure the UI updates in real-time as new data is received.
- Use appropriate ranges for the metrics (e.g., 0-150°C for engine temperature, 0-100 psi for oil pressure, 0-15 V for battery voltage).

Question 4: Handle Fault Detection (4 Marks)

- If the engine temperature exceeds 120°C, display a fault message in the status bar and log the fault to a file.
- If the oil pressure drops below 20 psi, display a fault message in the status bar and log the fault to a file.
- If the battery voltage drops below 11 V, display a fault message in the status bar and log the fault to a file.

Question 5: Add Start/Stop Functionality (3 Marks)

- Add two buttons: ”Start Monitoring” and ”Stop Monitoring”.
- When ”Start Monitoring” is clicked, start the ‘FaultDetectionThread‘ and update the status bar to ”Monitoring”.
- When ”Stop Monitoring” is clicked, stop the thread and update the status bar to ”Stopped”.

Question 6: Log Faults to a File (2 Marks)

- Add a "Log Faults" button to save all detected faults to a text file.
- The file should include a timestamp and details of each fault.

Total Marks: 20

- Question 1: 3 Marks
- Question 2: 5 Marks
- Question 3: 4 Marks
- Question 4: 4 Marks
- Question 5: 3 Marks
- Question 6: 2 Marks

Estimated Duration: 3 Hours

This exam tests your ability to work with 'QThread', 'QTimer', custom signals, 'QLabel', and file handling in PyQt5. It also evaluates your understanding of dynamic UI updates and fault detection logic. Good luck!