### **Task: Log Request Information to a File Using fs Module in Express.js**

**Objective:** Create an Express.js application that logs essential request information (timestamp, IP, URL, protocol, HTTP method, and hostname) to a file using the fs (File System) module. This task will help you understand middleware in Express.js and how to interact with the file system in Node.js.

### **Problem 1: Set Up the Express.js Application**

* Create a basic Express.js server.
* Set up the necessary dependencies (e.g., Express) and create the main server file.
* Ensure the server is running and listening on a port.

### **Problem 2: Implement Middleware to Capture Request Details**

* Implement middleware in Express.js to capture the following details for every incoming request:
  + **Timestamp**: The current date and time when the request was made.
  + **IP Address**: The IP address of the client making the request.
  + **URL**: The requested URL path.
  + **Protocol**: The protocol used for the request (e.g., HTTP, HTTPS).
  + **HTTP Method**: The HTTP method used for the request (e.g., GET, POST).
  + **Hostname**: The hostname from which the request is made.

### **Problem 3: Use the fs Module to Write Request Details to a File**

* Using the built-in **fs (File System)** module in Node.js, write the captured request details to a log file named requests.log.
  + Make sure that each log entry is written as a **JSON object**.
  + Use fs.appendFile() to ensure new log entries are appended to the file without overwriting the existing data.
  + Each entry in the log file should be written on a new line to make it easily readable.

### **Problem 4: Test the Logging Functionality**

* Start the Express.js server and test it by making several requests (e.g., by visiting the server in a browser or using tools like Postman or cURL).
* After making requests, verify the contents of the requests.log file. It should contain the logged details in JSON format, including:
  + The timestamp of the request.
  + The IP address of the client.
  + The requested URL path.
  + The HTTP method (GET, POST, etc.).
  + The protocol used (http or https).
  + The hostname from which the request originated.

### **Problem 5: Optional Advanced Features**

* **Log Rotation**: Implement a mechanism that rotates the log file when it reaches a certain size (e.g., 1MB). You can archive the current log and create a new log file with a timestamp in the filename.
* **Enhanced Logging**: Add more details to the log, such as:
  + Query parameters (e.g., req.query).
  + Request headers (e.g., req.headers).
  + User-agent (e.g., req.get('User-Agent')).

These problems guide students through setting up an Express.js application, implementing middleware, using the **fs module** to log requests to a file, and ensuring that the logs are written and appended correctly.