**Project Report: Student Helpdesk System using Queue in Python**

**1. Title** Student Helpdesk System using Queue in Python

**2. Introduction** The Student Helpdesk System is a Python-based application designed to manage and organize student queries efficiently. It replicates a real-world helpdesk where queries or issues raised by students are handled in a fair and systematic manner using the First-In-First-Out (FIFO) principle. This method ensures that all requests are addressed in the order they are received, maintaining fairness and avoiding favoritism or neglect.

This project utilizes Python's built-in data structure, deque from the collections module, to simulate a queue for storing and processing help requests. The application is built as a console-based menu-driven interface that allows users to perform basic helpdesk operations like adding, resolving, viewing, and counting requests.

**3. Problem Definition** In educational institutions, managing student grievances and technical support requests is a continuous challenge. Often, in the absence of a structured system, some requests get ignored or are unfairly delayed. A lack of proper tracking leads to dissatisfaction among students and overburdened staff.

To overcome these issues, this project proposes a simple helpdesk system that implements a queue to store and manage help requests. By enforcing FIFO order, it ensures that each query is attended to in the exact sequence it was made, improving efficiency and student satisfaction.

**4. Objectives**

* To build a console-based application using Python that simulates a student helpdesk.
* To use the queue data structure (deque) to maintain the FIFO order of request processing.
* To provide basic functionalities such as add, resolve, view, and count help requests.
* To demonstrate object-oriented programming principles using Python classes and methods.
* To enhance practical understanding of data structures and algorithm concepts.

**5. Tools and Technologies Used**

* **Programming Language:** Python 3.6+
* **Data Structure Used:** Queue (implemented using collections.deque)
* **IDE/Text Editors:** Visual Studio Code, PyCharm, or IDLE
* **Platform:** Command-line interface (CLI)
* **Operating System:** Windows/Linux/macOS

**6. Experimental Setup**

* Install Python (version 3.6 or higher) on the system.
* Create a file named helpdesk.py and copy the Python code.
* Run the file using terminal or command prompt with python helpdesk.py.
* Interact with the application using the menu displayed on the console.

**7. System Features**

* **Add Request:** Allows the user to submit a new help request.
* **Resolve Request:** Removes and displays the oldest request from the queue.
* **View Requests:** Lists all current pending help requests.
* **Count Requests:** Displays the total number of pending requests.
* **Exit:** Closes the application.

**8. System Design** The system is designed using object-oriented programming (OOP). It consists of the following components:

* **Class:** Helpdesk
  + **Attributes:** queue (stores requests)
  + **Methods:**
    - add\_request(request)
    - resolve\_request()
    - view\_requests()
    - count\_requests()
* **Function:** menu() to provide user interaction.

**9. Example Use Case**

User selects option 1:

Input: "Wi-Fi not working"

Output: Request added: "Wi-Fi not working"

User selects option 1 again:

Input: "Laptop not starting"

Output: Request added: "Laptop not starting"

User selects option 3:

Output:

1. Wi-Fi not working

2. Laptop not starting

User selects option 2:

Output: Resolved request: "Wi-Fi not working"

User selects option 4:

Output: Total pending requests: 1

**10. Case Study** Imagine a college IT department where multiple students raise complaints regarding Wi-Fi, login issues, or system access. Without a centralized tracking system, some of these issues might be forgotten or addressed unfairly. By using the Student Helpdesk System, these requests are queued and processed in the order they arrive. This ensures no student is overlooked and technical staff can address issues systematically.

**11. Discussion** This project exemplifies how simple data structures like queues can be effectively used to solve real-world problems. By employing the deque structure from Python's standard library, the system efficiently handles queue operations. The modular and class-based design ensures code reusability and better maintenance. It also reinforces key OOP concepts while teaching the importance of task order in systems handling multiple requests.

**12. Results** The application successfully manages help requests using FIFO logic. It performs as expected when:

* Multiple help requests are added.
* Requests are resolved in the correct order.
* Pending requests are correctly listed and counted.

The menu interface is intuitive and easy to use even for users with minimal programming knowledge.

**13. Conclusion** The Student Helpdesk System provides a practical demonstration of how Python can be used to build real-world systems using basic data structures. It successfully simulates a helpdesk scenario, ensuring fair and timely handling of queries. This foundational project can be enhanced further with features like GUI integration, database connectivity, and priority-based request management.

**14. Future Scope**

* Add user authentication and login system.
* Store help requests in a database.
* Implement priority queues for urgent issues.
* Create a web or mobile application with the same backend logic.