



VNR Vignana Jyothi Institute of Engineering and Technology
(Affiliated to J.N.T.U, Hyderabad)
Bachupally(v), Hyderabad, Telangana, India.

EVENT SCHEDULING MANAGEMENT

A course project submitted in complete requirements for the award of the degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING

Submitted by

A.Harsha Vardhan (23071A05D9)

A.Yoshith (23071A05E0)

D.Pooja Sree (23071A05F0)

D.Harshitha (23071A05F4)

G.Vaishnavi (23071A05F6)

Under the guidance of

Ms. D N Vasundhara

Assistant Professor

CSE

VNRVJIET



VNR Vignana Jyothi Institute of Engineering and Technology
(Affiliated to J.N.T.U, Hyderabad)

Bachupally(v), Hyderabad, Telangana, India.

CERTIFICATE

This is to certify A.Harsha Vardhan (23071A05D9),A.Yoshith (23071A05E0),D.Pooja Sree (23071A05F0),D.Harshitha (23071A05F4),G.Vaishnavi (23071A05F6)have completed their course-based project at CSE Department of VNR VJIET, Hyderabad entitled" **EVENT SCHEDULING MANAGEMENT** " in complete fulfilment of the requirements for the award of B Tech degree during the academic year 2023- 2024. This work is carried out under my supervision and has not been submitted to any other University/Institute for award of any degree/diploma.

Ms. D N Vasundhara
Assistant Professor
CSE
VNRVJIET

Dr.V Baby
Professor and HOD
CSE
VNRVJIET

DECLARATION

This is to certify that our project report titled “**EVENT SCHEDULING MANAGEMENT** ” submitted to Vallurupalli Nageswara Rao Institute of Engineering and Technology in complete fulfilment of requirement for the award of Bachelor of Technology in Computer Science and Engineering is a bonafide report to the work carried out by us under the guidance and supervision of Mr. K. Sri Rama Murthy, Assistant Professor, Department of Computer Science and Engineering, Vallurupalli Nageswara Rao Institute of Engineering and Technology. To the best of our knowledge, this has not been submitted in any form to other university or institution for the award of any degree or diploma.

A.Harsha Vardhan (23071A05D9)

A.Yoshith (23071A05E0)

D.Pooja Sree (23071A05F0)

D.Harshitha (23071A05F4)

G.Vaishnavi (23071A05F6)

ACKNOWLEDGEMENT

Over a span of three and a half years, VNRVJIET has helped us transform ourselves from mere amateurs in the field of Computer Science into skilled engineers capable of handling any given situation in real time. We are highly indebted to the institute for everything that it has given us. We would like to express our gratitude towards the principal of our institute, **Dr. Challa Dhanunjaya Naidu** and the Head of the Computer Science & Engineering Department, **Dr. V Baby** for their kind co- operation and encouragement which helped us complete the project in the stipulated time. Although we have spent a lot of time and put in a lot effort into this project, it would not have been possible without the motivating support and help of our project guide **Ms. D N Vasundhara** . We thank them for their guidance, constant supervision and for providing necessary information to complete this project. Our thanks and appreciations also go to all the faculty members, staff members of VNRVJIET, and all our friends who have helped us put this project together.

ABSTRACT

The "Event Scheduling Management" project addresses the complexities and challenges of organizing, managing, and prioritizing events. It provides a comprehensive solution designed to facilitate effective event management by leveraging advanced scheduling algorithms. The system offers a user-friendly interface that allows users to create, edit, and manage events, ensuring optimal utilization of resources and time. By integrating scheduling algorithms such as First-Come, First-Served (FCFS), Shortest Job Next (SJN), and Priority Scheduling, the system provides a robust mechanism to handle events of varying importance and durations efficiently. This project not only aids in understanding event management processes but also serves as a practical tool for analyzing and improving scheduling strategies.

Introduction

Effective event scheduling is critical in various domains, from academic environments to corporate settings, as it enhances productivity by optimizing the use of time and resources while minimizing conflicts. The "Event Scheduling Management" system is designed to address these needs by providing a comprehensive platform for organizing and managing events.

This system is particularly valuable in educational settings, where understanding and applying scheduling algorithms can significantly impact the efficiency of event management. By implementing this system in C++, the project demonstrates a practical application of computer science principles, focusing on dynamic scheduling, resource allocation, and priority management. The primary objective is to provide an intuitive and efficient tool for managing events, thereby enhancing overall productivity and learning outcomes.

Algorithm

Scheduling Algorithms

The Event Scheduling Management system employs several key scheduling algorithms to optimize the organization and prioritization of events. These algorithms are based on classic operating system scheduling techniques, each with unique advantages and use cases.

1. First-Come, First-Served (FCFS):

- **Description:** This algorithm schedules events in the order they are added to the system. It is the simplest form of scheduling, ensuring that the first event to enter the queue is the first to be processed.
- **Advantages:** Easy to implement and understand. Maintains a straightforward and predictable scheduling order.
- **Disadvantages:** Does not consider event duration or priority, which may lead to inefficient use of resources if longer events block shorter, more critical tasks.

2. Shortest Job First (SJF):

- **Description:** Also known as Shortest Job First (SJF), this algorithm prioritizes events based on their duration, scheduling shorter events before longer ones. This approach aims to minimize the average waiting time for events in the queue.
- **Advantages:** Efficient for reducing overall wait time and maximizing throughput. Particularly effective when there are many short events that can be completed quickly.
- **Disadvantages:** Requires knowledge of event durations in advance, which may not always be possible. Can lead to the "starvation" of longer events if short events continue to arrive.

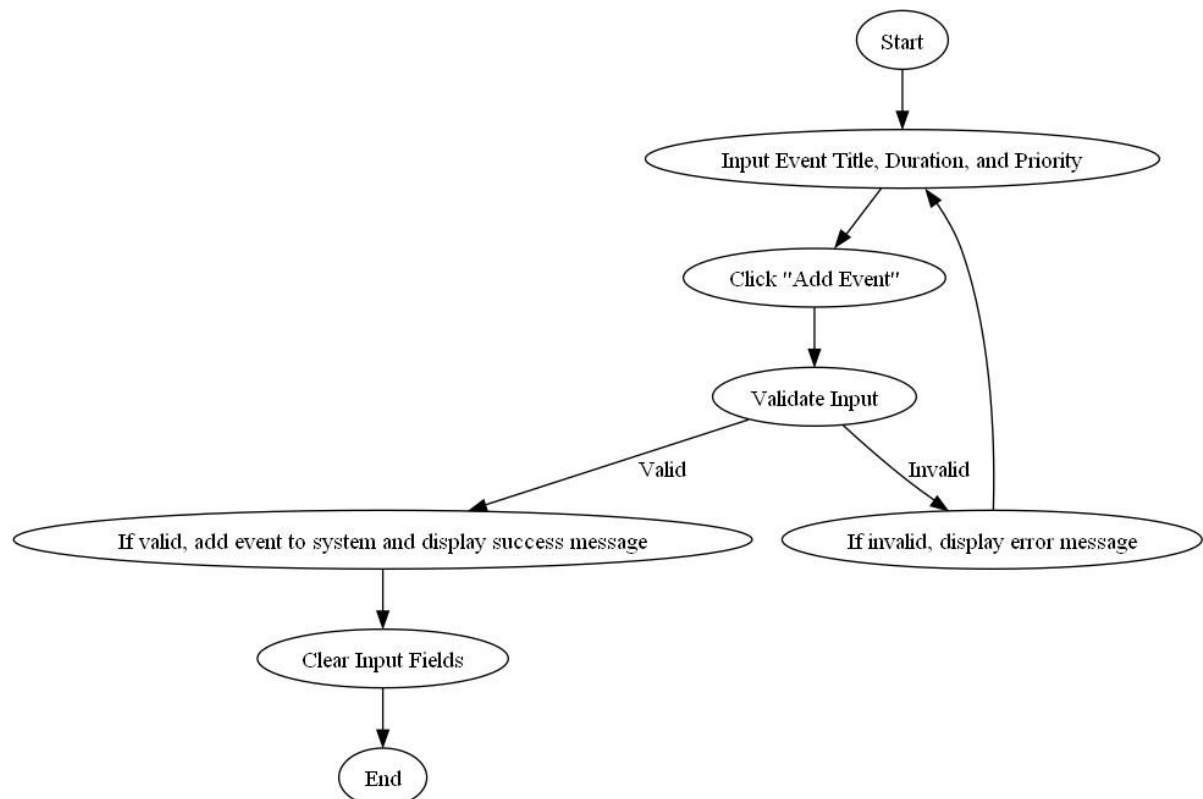
3. Priority Scheduling:

- **Description:** Events are assigned a priority level, and those with higher priority (lower numerical value) are scheduled first. This method ensures that more important events are handled promptly.

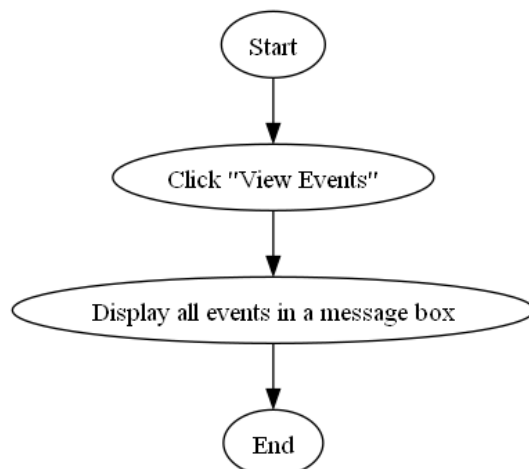
- **Advantages:** Ensures that critical events receive immediate attention. Flexible in accommodating different priority schemes.
- **Disadvantages:** May lead to "starvation" of lower-priority events if high-priority events are continually added. Requires careful management of priority levels to avoid unfair scheduling.

Analysis and Design Diagrams

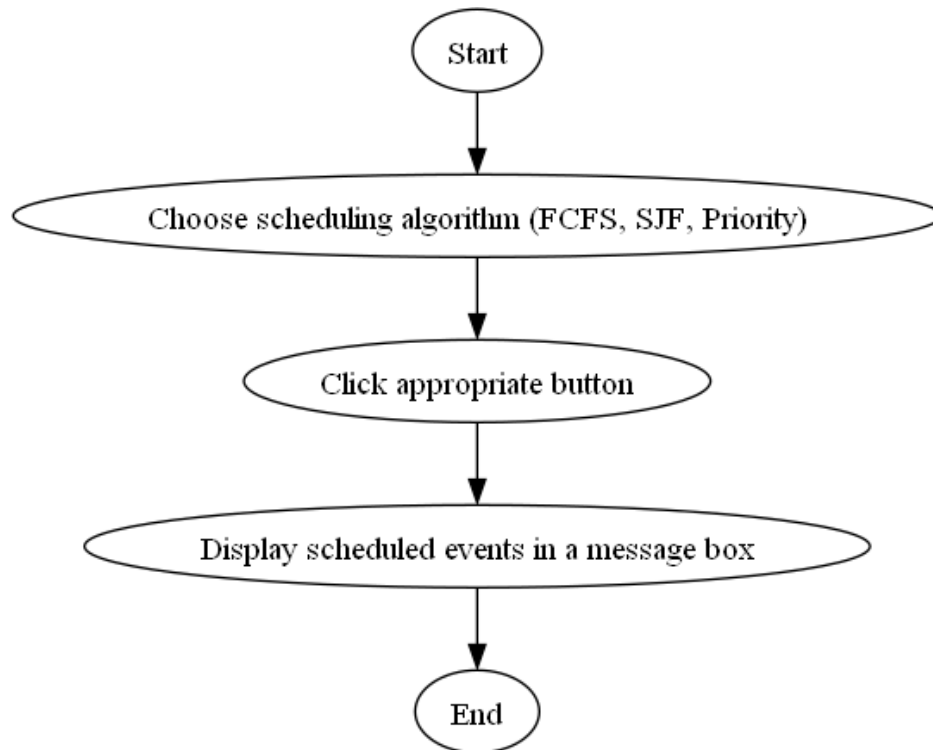
Flow chart to add an event



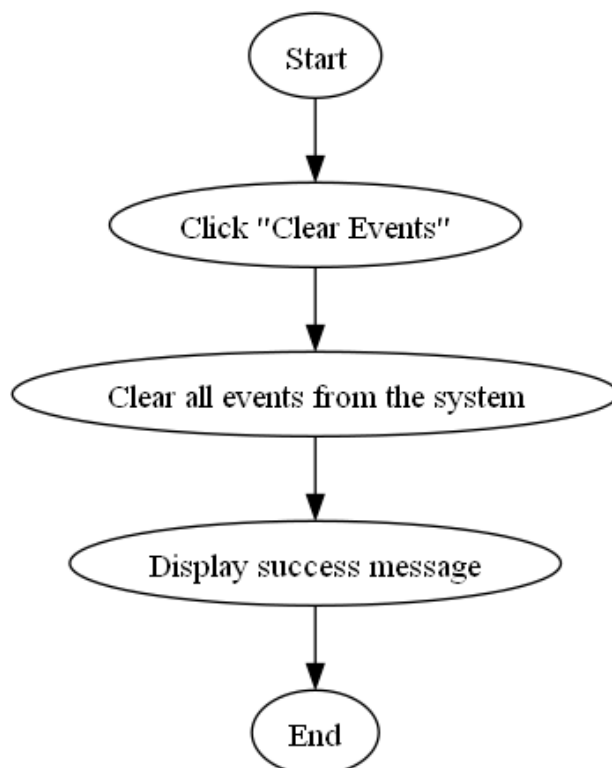
Flow chart to view an event



Flow chart to schedule the events



Flow chart to clear all events



Implementation

```
import tkinter as tk

from tkinter import messagebox

class Event:
    def __init__(self, event_id, title, duration, priority):
        self.event_id = event_id
        self.title = title
        self.duration = duration
        self.priority = priority

class EventManagementSystem:
    def __init__(self):
        self.events = []
        self.next_id = 1

    def add_event(self, title, duration, priority):
        event = Event(self.next_id, title, duration, priority)
        self.events.append(event)
        self.next_id += 1

    def get_events(self):
        return self.events

    def clear_events(self):
        self.events = []
        self.next_id = 1

class SchedulingAlgorithms:
    @staticmethod
    def fcfs(events):
        return sorted(events, key=lambda x: x.event_id)

    @staticmethod
    def sjn(events):
        return sorted(events, key=lambda x: x.duration)

    @staticmethod
    def priority_scheduling(events):
        return sorted(events, key=lambda x: x.priority, reverse=True)

class EventManagementGUI:
    def __init__(self, root):
        self.ems = EventManagementSystem()
        self.root = root
```

```

# Add Event Frame
self.add_event_frame = tk.Frame(root, padx=10, pady=10)
self.add_event_frame.pack(pady=10)

tk.Label(self.add_event_frame, text="Event Title:").grid(row=0, column=0,
sticky='e', padx=5, pady=5)
self.title_entry = tk.Entry(self.add_event_frame, width=30)
self.title_entry.grid(row=0, column=1, padx=5, pady=5)

tk.Label(self.add_event_frame, text="Duration (hours):").grid(row=1, column=0,
sticky='e', padx=5, pady=5)
self.duration_entry = tk.Entry(self.add_event_frame, width=30)
self.duration_entry.grid(row=1, column=1, padx=5, pady=5)

tk.Label(self.add_event_frame, text="Priority:").grid(row=2, column=0, sticky='e',
padx=5, pady=5)
self.priority_entry = tk.Entry(self.add_event_frame, width=30)
self.priority_entry.grid(row=2, column=1, padx=5, pady=5)

tk.Button(self.add_event_frame, text="Add Event", command=self.add_event,
width=15).grid(row=3, columnspan=2, pady=10)

# View Events Button
tk.Button(root, text="View Events", command=self.view_events,
width=20).pack(pady=5)

# Schedule Events Frame
self.schedule_frame = tk.Frame(root, padx=10, pady=10)
self.schedule_frame.pack(pady=10)

tk.Label(self.schedule_frame, text="Schedule using:").pack(pady=5)

tk.Button(self.schedule_frame, text="FCFS", command=self.schedule_fcfs,
width=15).pack(side=tk.LEFT, padx=5)
tk.Button(self.schedule_frame, text="SJN", command=self.schedule_sjn,
width=15).pack(side=tk.LEFT, padx=5)
tk.Button(self.schedule_frame, text="Priority", command=self.schedule_priority,
width=15).pack(side=tk.LEFT, padx=5)

# Clear Events Button
tk.Button(root, text="Clear Events", command=self.clear_events,
width=20).pack(pady=5)

def add_event(self):
    title = self.title_entry.get()
    try:
        duration = int(self.duration_entry.get())

```

```

        priority = int(self.priority_entry.get())
        self.ems.add_event(title, duration, priority)
        messagebox.showinfo("Success", "Event added successfully!")
        # Clear the input fields
        self.title_entry.delete(0, tk.END)
        self.duration_entry.delete(0, tk.END)
        self.priority_entry.delete(0, tk.END)
    except ValueError:
        messagebox.showerror("Error", "Please enter valid numerical values for duration
and priority.")

    def view_events(self):
        events = self.ems.get_events()
        events_str = "\n".join([f"ID: {event.event_id}, Title: {event.title}, Duration:
{event.duration}, Priority: {event.priority}" for event in events])
        messagebox.showinfo("Events", events_str if events_str else "No events available.")

    def schedule_fcfs(self):
        self.show_scheduled_events(SchedulingAlgorithms.fcfs(self.ems.get_events()))

    def schedule_sjn(self):
        self.show_scheduled_events(SchedulingAlgorithms.sjn(self.ems.get_events()))

    def schedule_priority(self):
        self.show_scheduled_events(SchedulingAlgorithms.priority_scheduling(self.ems.get_events()))

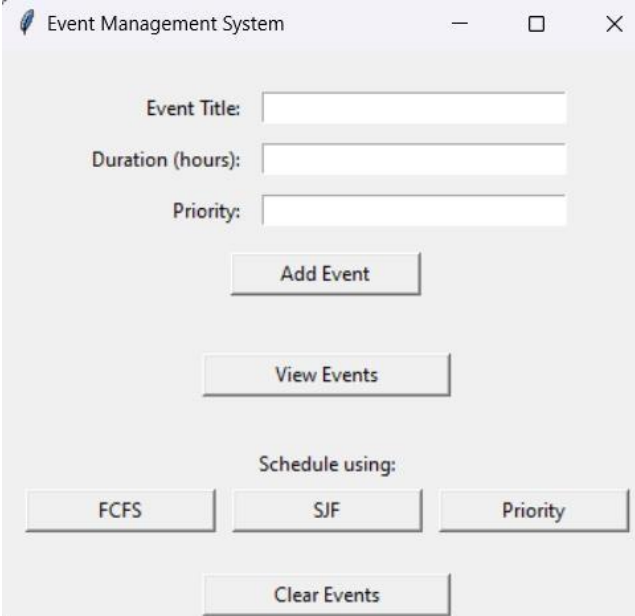
    def show_scheduled_events(self, events):
        events_str = "\n".join([f"ID: {event.event_id}, Title: {event.title}, Duration:
{event.duration}, Priority: {event.priority}" for event in events])
        messagebox.showinfo("Scheduled Events", events_str if events_str else "No events
to schedule.")

    def clear_events(self):
        self.ems.clear_events()
        messagebox.showinfo("Success", "All events cleared.")

if __name__ == "__main__":
    root = tk.Tk()
    app = EventManagementGUI(root)
    root.mainloop()

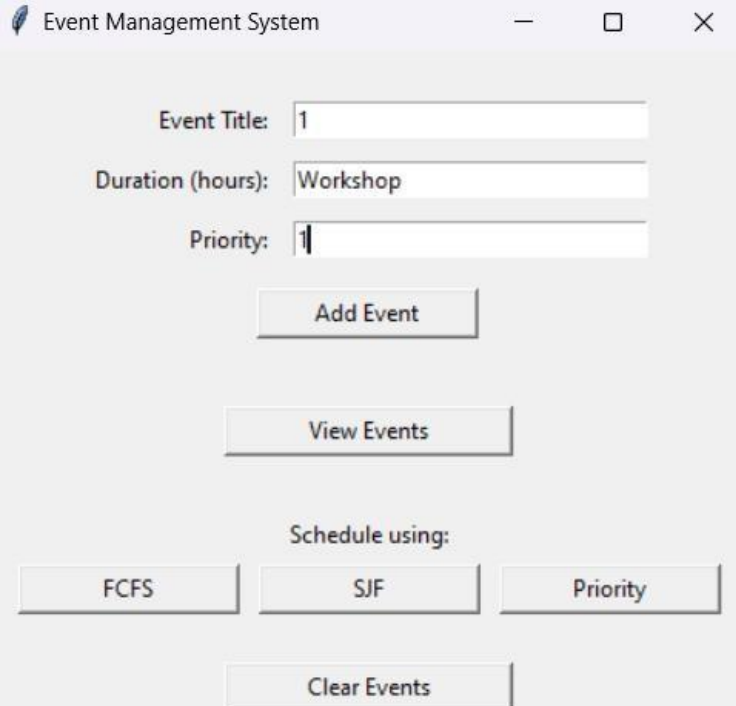
```

OUTPUT

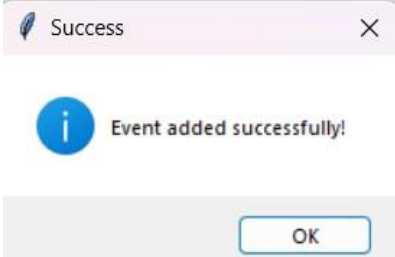


The image shows a window titled "Event Management System" with a standard Windows title bar (minimize, maximize, close buttons). Inside the window, there are three input fields: "Event Title:", "Duration (hours):", and "Priority:". Below these fields are four buttons: "Add Event", "View Events", "Schedule using:" (which is a label above three buttons: "FCFS", "SJF", and "Priority"), and "Clear Events".

Adding Events :



The image shows the same "Event Management System" window, but now with data entered into the input fields. The "Event Title:" field contains "1", the "Duration (hours):" field contains "Workshop", and the "Priority:" field contains "1". The "Add Event" button is highlighted, indicating it has been clicked. The other buttons and labels remain the same.



The image shows a small dialog box titled "Success" with a standard Windows title bar. It contains a blue information icon (i) and the text "Event added successfully!". Below the text is an "OK" button.

Success

Event added successfully!

OK

Event Management System

Event Title:

Duration (hours):

Priority:

Add Event

View Events

Schedule using:

Clear Events

View Events :

Events

ID: 1, Title: Workshop, Duration: 2, Priority: 1
 ID: 2, Title: TedX, Duration: 3, Priority: 4
 ID: 3, Title: Corporate Interaction, Duration: 4, Priority: 3
 ID: 4, Title: Seminar, Duration: 3, Priority: 2

OK

Scheduling Events :



FCFS :


Scheduled Events

ID: 1, Title: Workshop, Duration: 2, Priority: 1
 ID: 2, Title: TedX, Duration: 3, Priority: 4
 ID: 3, Title: Corporate Interaction, Duration: 4, Priority: 3
 ID: 4, Title: Seminar, Duration: 3, Priority: 2

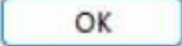
OK

SJF :



 Scheduled Events 




ID: 1, Title: Workshop, Duration: 2, Priority: 1
ID: 2, Title: TedX, Duration: 3, Priority: 4
ID: 4, Title: Seminar, Duration: 3, Priority: 2
ID: 3, Title: Corporate Interaction, Duration: 4, Priority: 3

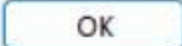


Priority :



 Scheduled Events 




ID: 2, Title: TedX, Duration: 3, Priority: 4
ID: 3, Title: Corporate Interaction, Duration: 4, Priority: 3
ID: 4, Title: Seminar, Duration: 3, Priority: 2
ID: 1, Title: Workshop, Duration: 2, Priority: 1

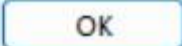




Clear Events :


 Success 




All events cleared.



 Events 



No events available.



Future Scope

- **AI and Machine Learning Integration:**
 - Enhancing algorithms with AI for dynamic adaptation and ML for predictive scheduling based on historical data.
 - Optimizing decisions in real-time with AI-driven insights and adaptive adjustments.
- **IoT and Sensor Networks Utilization:**
 - Leveraging IoT for real-time venue data and sensor networks for environmental conditions.
 - Enhancing scheduling accuracy and efficiency with data-driven insights from IoT and sensor inputs.
- **Blockchain for Transparency and Security:**
 - Implementing blockchain for immutable scheduling records and smart contracts automation.
 - Ensuring secure and transparent scheduling processes with blockchain's decentralized ledger.
- **Personalized and Context-Aware Scheduling:**
 - Tailoring schedules based on participant preferences and contextual factors.
 - Enhancing participant engagement and satisfaction through personalized scheduling approaches.
- **Scalability and Large-Scale Event Management:**
 - Scaling algorithms to manage complex events with efficiency and performance.
 - Optimizing scheduling for large-scale events with distributed computing and scalable solutions.
- **Environmental and Sustainability Focus:**

- Integrating sustainability considerations to minimize environmental impact.
- Promoting eco-friendly practices in event logistics and resource management.
- **Cross-Domain Applications and Industry Adaptation:**
- Applying scheduling algorithms beyond traditional domains to diverse industries.
- Enhancing efficiency and service delivery across healthcare, transportation, and urban planning.