



Event Management and Collaboration App

K. Asha Devi ¹, P. Deepika Siva Kamakshi ², V. Sai Kumar ³, S. Aravindh ⁴, T. Kalyani ⁵, V. Naveen ⁶

¹ Assistant Professor, Dept. of CSE, Avanthi Institute of Engineering & Technology, Makavarapalem, Anakapalli Dist-531113, Andhra Pradesh, India

^{2,3,4,5,6} Student, Dept. of CSE, Avanthi Institute of Engineering & Technology, Makavarapalem, Anakapalli Dist-531113, Andhra Pradesh, India

Abstract— Organizing an event involves coordinating scheduling, task assignment, vendor communication, and attendee logistics across multiple stakeholders, a process that is error-prone when managed through disconnected spreadsheets, emails, and messaging threads. This paper presents a detailed technical treatment of an event-management and collaboration application that combines scheduling, task-dependency tracking, real-time collaboration, and attendee-facing features into a single platform. Beyond a conventional system description, this paper grounds the scheduling and task-dependency design in classical project-scheduling theory, critically reviews collaboration-platform and real-time-communication literature, and formalizes the task-critical-path computation used to highlight scheduling risk ahead of an event date. A prototype was implemented using a React front end with WebSocket-based real-time updates, a Node.js/Express backend, and a MongoDB data store. Illustrative (simulated) evaluation indicates that critical-path-aware task scheduling reduces estimated pre-event scheduling risk relative to an unmonitored task list.

Index Terms— event management, collaboration software, critical path method, real-time sync, task dependencies, project scheduling.

I. INTRODUCTION

Event planning requires synchronized multi-stakeholder collaboration. Disconnected tools lead to deadline slippage and communication bottlenecks. This paper presents a critical-path-aware event management platform.

II. PROBLEM STATEMENT / OBJECTIVES

To formalize event task dependencies using Critical Path Method (CPM) graph analysis, enabling real-time multi-user WebSocket state synchronization and risk flagging.

III. LITERATURE REVIEW & IV. EXISTING SYSTEM

Grounded in classical project management theory (CPM/PERT), real-time operational transformation, and collaborative software architectures.

TABLE I: COMPARISON OF EXISTING APPROACHES

Approach	Key Technique	Limitation
Disconnected spreadsheets/email	Manual tracking	High error rate; lack of real-time visibility
Generic task boards	Unlinked task lists	Ignores task dependencies and critical path risk
Enterprise project software	Full PERT/CPM tools	Overly complex for event-specific workflows
Proposed platform	CPM-aware event collaboration + WebSockets	Requires accurate dependency input



V. PROPOSED SYSTEM & VII. METHODOLOGY

Task graphs are evaluated for critical paths, identifying zero-float tasks. Real-time updates propagate via WebSocket connections.

VIII. ARCHITECTURE & IX. IMPLEMENTATION

Built with React front end, Node.js/Express backend, Socket.io WebSocket server, and MongoDB graph store.

XI. RESULTS AND DISCUSSION

Simulated scheduling evaluations show significant reduction in pre-event bottleneck risks and milestone delays when using CPM-guided prioritization.

XII. ADVANTAGES / LIMITATIONS &

XIII. ETHICAL CONSIDERATIONS

Provides actionable risk visibility for organizers. Requires accurate task duration and dependency estimation.

XIV. CONCLUSION & XV. FUTURE SCOPE

Future work involves automated vendor SLA integration, AI duration forecasting, and offline sync capability.