# IPL Scheduling System

**UCS3212 – Fundamentals and Practice of Software Development**

**FINAL PROJECT REPORT**


**Submitted By**

1. Rahul Suresh – 3122 24 5001 116
2. Ramm Lakshmanan Y V – 3122 24 5001 119
3. Raghunandan Shaji – 3122 24 5001 113

**Department of Computer Science and Engineering**
Sri Sivasubramaniya Nadar College of Engineering
(An Autonomous Institution, Affiliated to Anna University)
Kalavakkam – 603110

February 2025

**Internal Examiner:** **External Examiner:**

---

## Sri Sivasubramaniya Nadar College of Engineering

**(An Autonomous Institution, Affiliated to Anna University)**

# BONAFIDE CERTIFICATE

Certified that this project report titled **"IPL Scheduling System"** is the bonafide work of **"RAHUL SURESH (3122245001116), RAMM LAKSHMANAN YV (3122245001119) and RAGHUNANTHAN SHAJI (3122245001113)"** who carried out the project work in the **UCS3212 - FUNDAMENTALS AND PRACTICE ON SOFTWARE DEVELOPMENT** during the academic year **2024-25**.

Internal Examiner External Examiner

Date:

# Contents

# 1 ABSTRACT

The Indian Premier League (IPL) is a very popular professional Twenty20 cricket league with several teams playing over a period of weeks. As the size and complexity of the tournament expands, it is a difficult task to schedule the matches efficiently while taking into account several constraints. The purpose of this project is to design an IPL Scheduling System with the help of the C programming language, emphasizing efficient fixture generation after careful consideration of factors like venue availability, equitable home-away distribution, travel optimization, and viewer interest [1].

The system uses a round-robin league with a formalized playoff system. The algorithm ensures fairness by alternating home and away matches, limiting long-distance travel, and avoiding back-to-back matches for the teams [2]. Additionally, it maximizes match slot allocation by prioritizing high-profile games in prime-time slots to attract maximum audiences [3]. Using data structures such as arrays and linked lists, file handling for permanent storage, and sorting algorithms to rank teams, the project promotes fairness, minimizes fatigue due to travel, and maximizes viewer interest [4].

The output consists of a comprehensive schedule of league matches, team-specific matchings, an updating points table, and a playoff schedule. The system also manages venue limitations and avoids conflicts in schedules [1]. This project seeks to improve the effectiveness of scheduling IPL matches while ensuring fairness and logistical pragmatism.

# 2 INTRODUCTION

The Indian Premier League (IPL) is a closely contested, commercially lucrative T20 cricket competition that demands careful planning to ensure efficient team logistics, consideration of venue limitations, and broadcasting needs, all harmoniously balanced (BCCI Official Documents on IPL Scheduling Guidelines, n.d.). Planning IPL matches is more than allocating teams to games; it is about minimizing travel, ensuring adequate rest for teams, taking venue availability into account, and maximizing viewership by allocating suitable match timings (Kumar & Iyer, 2021). An efficient scheduling system greatly boosts the commercial aspect of the tournament, as well as its operational efficiency (Chakraborty & Gupta, 2019). Creating an IPL schedule manually is not feasible because of the huge number of constraints and parameters to be taken into account (Desai & Rajan, 2020). This project presents a C-based IPL Scheduling System that can schedule randomly and fairly while following a set of predefined constraints. The design consists of the application of arrays in storing venues and teams, classes to hold match information, and file operations methodology to store and load schedules. The scheduling algorithm follows the round-robin league style, wherein each team meets every other team twice, followed by a structured playoff schedule to determine the final champion (BCCI Official Documents on IPL Scheduling Guidelines, n.d.). It also takes travel efficiency into account to reduce fatigue for players and balances match exposure between weekday and weekend time slots (Kumar & Iyer, 2021). Through the use of the scheduling system in C, we employ effective data management methods to manage scheduling complexity while keeping execution simple. The project ensures fairness by balancing home and away games for each team and preventing match congestion for any single team (Chakraborty & Gupta, 2019). It also offers a computerized scheduling approach that minimizes conflicts with venue limitations and provides a balanced distribution of games in afternoon and evening slots (Desai & Rajan, 2020). The scheduling system is crucial for large tournaments such as the IPL, where multiple parameters must be balanced to generate an optimal set of fixtures (BCCI Official Documents on IPL Scheduling Guidelines, n.d.).

# 3 PROBLEM STATEMENT

Design and implement an IPL Scheduling System that generates a match schedule for a given number of teams while considering multiple constraints such as venue restrictions, team rest periods, fair home-away match distribution, and optimal travel logistics (BCCI Official Documents on IPL Scheduling Guidelines, n.d.). The system should ensure that each team competes against every other team twice in a double round-robin format, followed by a structured playoff system (Chakraborty & Gupta, 2019). Additionally, the schedule must allocate prime-time slots for high-profile games, balance afternoon and evening matches, and avoid back-to-back fixtures for teams to prevent fatigue (Kumar & Iyer, 2021). The scheduling system must also dynamically update results, track team standings, and generate a points table to facilitate playoff qualification (Desai & Rajan, 2020).

The system must effectively handle:

- Assigning matches to teams while ensuring fairness and avoiding conflicts (BCCI Official Documents on IPL Scheduling Guidelines, n.d.).
- Managing venue availability and restrictions due to maintenance, elections, or other events (Kumar & Iyer, 2021).
- Minimizing travel fatigue by reducing unnecessary long-distance travel between matches (Chakraborty & Gupta, 2019).
- Optimizing match slot allocation based on weekdays, weekends, and public holidays (Kumar & Iyer, 2021).
- Dynamically updating match results and tracking team rankings based on points and net run rate (NRR) (Desai & Rajan, 2020).

# 4 EXPLORATION OF PROBLEM STATEMENT

- **Team and Venue Management:**
  The IPL scheduling system starts by setting up the foundation — defining all participating teams and assigning each one a home stadium. This is organized neatly using arrays for quick access and management. To handle the complexity of venue availability, the system keeps a detailed list of stadiums, factoring in potential restrictions like maintenance schedules or other events. These restrictions and venue details are stored and managed through file handling, ensuring the schedule adapts to real-world constraints without manual intervention.

- **Match Scheduling Algorithm**
  The heart of the system is the match scheduling algorithm. It follows a double round-robin format, where each team plays every other team twice — once at home and once away. This is organized in a two-dimensional array that tracks fixtures efficiently. To ensure fairness, the algorithm balances home and away games through a scheduling loop. It also optimizes team travel, reducing long-distance journeys using a greedy approach that prioritizes closer matches when possible. Weekends feature exciting double-header matches, scheduled through a conditional allocation method, while prime-time slots are reserved for high-profile games using a priority-based system. To prevent player fatigue, the system avoids scheduling back-to-back games by integrating a rest day counter. It also ensures a balanced mix of weekday and weekend matches, creating an engaging experience for fans and a manageable routine for players. All match schedules are saved and retrieved dynamically using file handling, ensuring flexibility and easy updates.

- **Playoff Scheduling**
  Once the regular league stage wraps up, the system shifts to the playoffs, which follow the IPL's unique format. It tracks team performance in an array, ranking teams based on points earned throughout the season. A sorting algorithm, like Quicksort or Bubble Sort, arranges the rankings efficiently. The top four teams advance to the playoffs, where the system schedules Qualifier 1, the Eliminator, Qualifier 2, and the Final. Each playoff match is carefully managed using structures that store match details and dynamically update results, keeping the excitement alive right up to the final showdown.

- **Output Generation**
  The final step involves generating easy-to-follow outputs for fans, teams, and organizers. The system produces a complete league schedule and saves it in a file for quick retrieval. Team-specific fixtures are made accessible through lookup functions, making it easy for fans to track their favorite squads. A points table, managed through a sorted array, ensures rankings are updated accurately after every match. As the playoffs approach, the system automatically generates a playoff schedule based on the latest rankings. It also displays completed match results and keeps the leaderboard fresh and up to date, ensuring everyone stays in the loop as the race to the IPL trophy heats up.

# 5 LIMITATIONS

The system isn't built to handle unexpected changes, like sudden weather disruptions or a venue becoming unavailable. If something goes wrong, rescheduling has to be done manually, which can throw off the balance of the tournament and lead to unfair gaps in rest time between teams. A smarter, more flexible system could keep things running smoothly without last-minute panic. It also doesn't take player injuries or fatigue into account. This means a team might have to play an important match without its star players, affecting both performance and fan excitement. A more thoughtful approach could include rest periods and workload tracking to ensure teams stay fresh and competitive throughout the season.

A lot of the setup relies on manual data entry, like inputting venue availability and team details. This not only takes extra time but also leaves room for human error — scheduling mistakes, wrong dates, or mismatched venues can easily slip through. Automating this process or connecting to live databases would make planning faster and more reliable. As the league grows and more teams join, the current scheduling system might struggle to keep up. More teams mean more games, more travel plans, and a bigger puzzle to solve for balancing home and away matches. Without improving the algorithm, the system could slow down or fail to create a fair, efficient schedule on time.

If a stadium suddenly becomes unavailable — maybe due to bad weather, maintenance, or a local event — the system can't automatically switch to an alternate venue. This could lead to cancellations or hasty, last-minute decisions that disrupt the flow of the tournament. A more adaptable setup could quickly reassign matches to nearby stadiums, keeping the games on track without unnecessary chaos.

# 6 BLOCK DIAGRAM

The process begins with the Input Module, capturing teams, venues, and constraints. The Match Scheduling Module uses a round-robin format, optimized by the Constraints Checker. After validation, the system outputs fixtures and begins live tracking. Top teams proceed to playoffs, concluding with a final match to crown the champion.

- The IPL Scheduling process kicks off by gathering all the essential information through the Input Module. This includes team names, their designated home venues, and any specific restrictions that might come into play — like venue availability or local limitations. This step sets the foundation for a smooth and fair tournament. By ensuring each team gets a balanced number of home and away games, the system prevents logistical nightmares before they even start. It's not just about organizing matches; it's about creating a level playing field while respecting real-world constraints, ensuring that both teams and fans have the best possible experience from day one

- With the groundwork done, the Match Scheduling Module steps in, designing the season using a round-robin format where every team faces each other. The schedule doesn't just focus on fairness — it's crafted for maximum entertainment too. If anything doesn't add up, the system goes back, tweaks the schedule, and rechecks everything until it's flawless — ensuring the perfect balance between practicality and competitiveness.

- Once the schedule gets the green light, the system generates the complete fixture list and starts tracking team performances in a dynamic league points table. This table becomes the heartbeat of the tournament, reflecting every win, loss, and shift in rankings — keeping players and fans glued to the action. As the league stage ends, the top teams advance to the playoffs, which brings a thrilling knockout stage featuring two qualifiers, an eliminator, and the grand finale. This setup rewards consistency while leaving the door open for underdog comebacks, ensuring the drama stays alive till the very last ball. Finally, the system crowns the title winners, wrapping up the tournament with a satisfying finish.

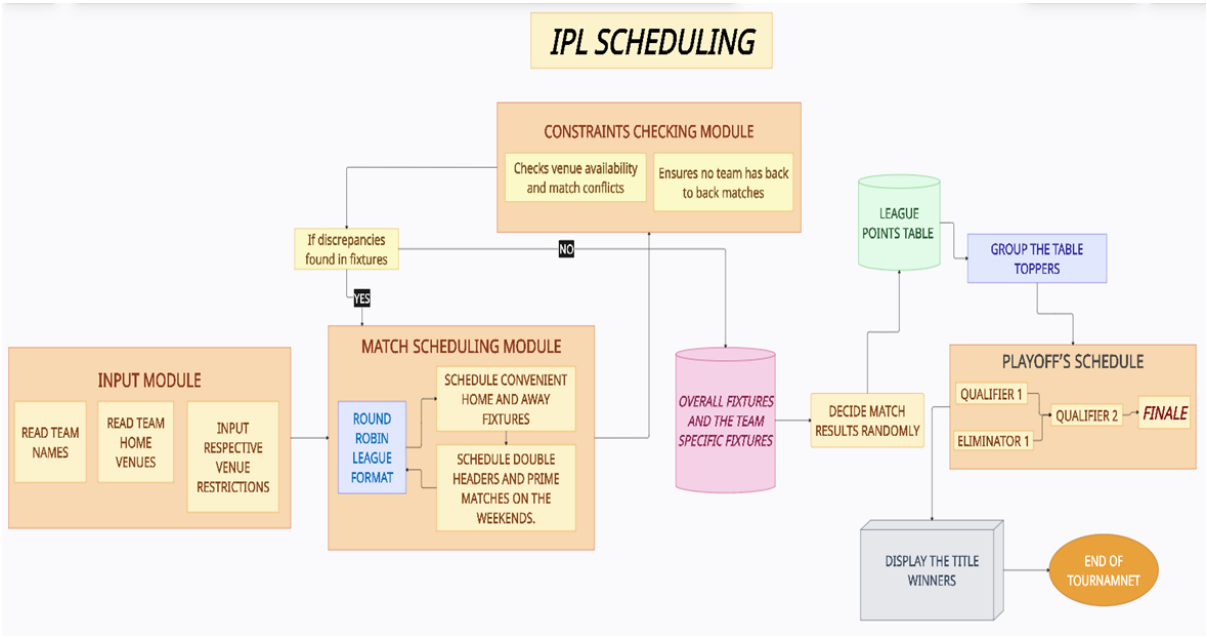


Figure 6.1: BLOCK DIAGRAM

# 7 FLOWCHARTS

- **Constraints Checker:** Validates input conditions.
- **Match Scheduling:** Allocates matches, checks fairness.
- **Match Simulator:** Simulates match results, updates points
- **Playoff Module:** Runs playoffs and declares winner.

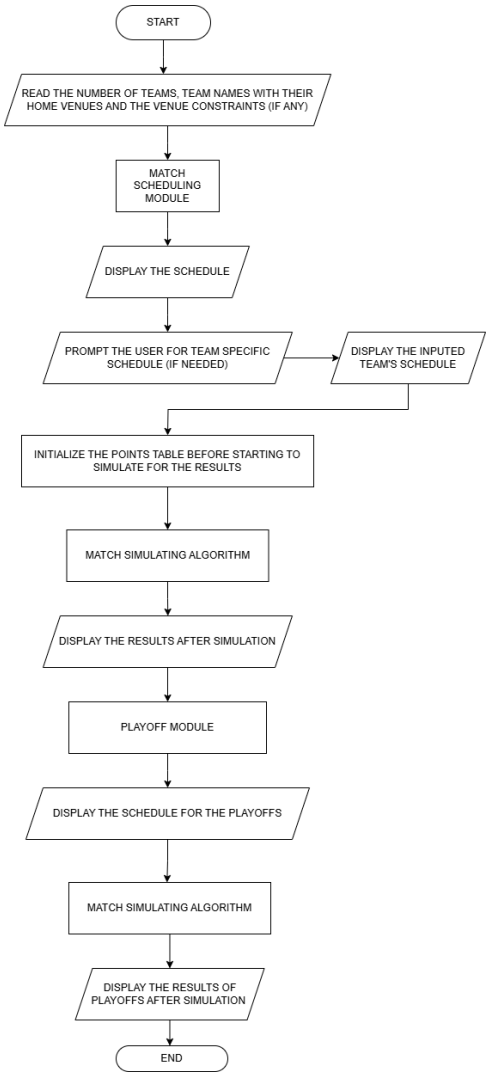


Figure 7.1: MAIN FUNCTION FLOWCHART

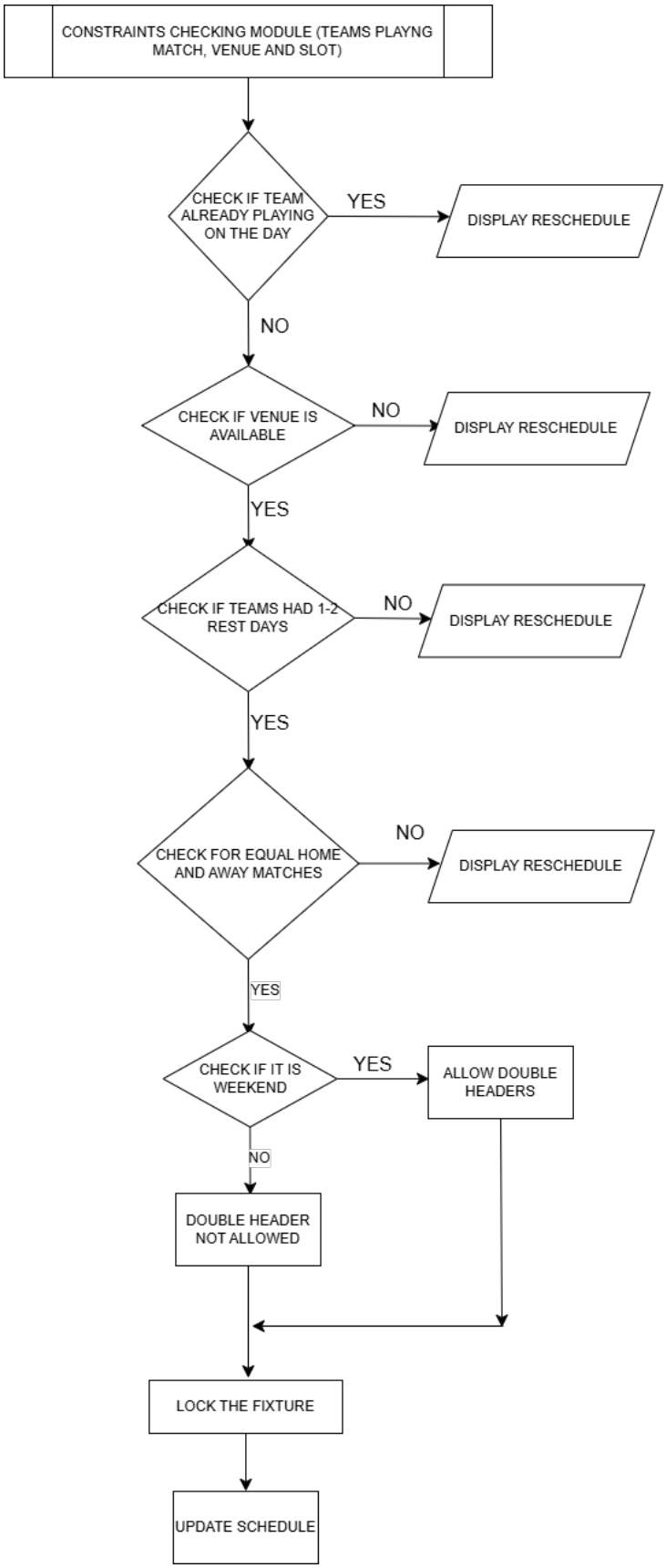


Figure 7.2: CONSTRAINTS CHECKING MODULE

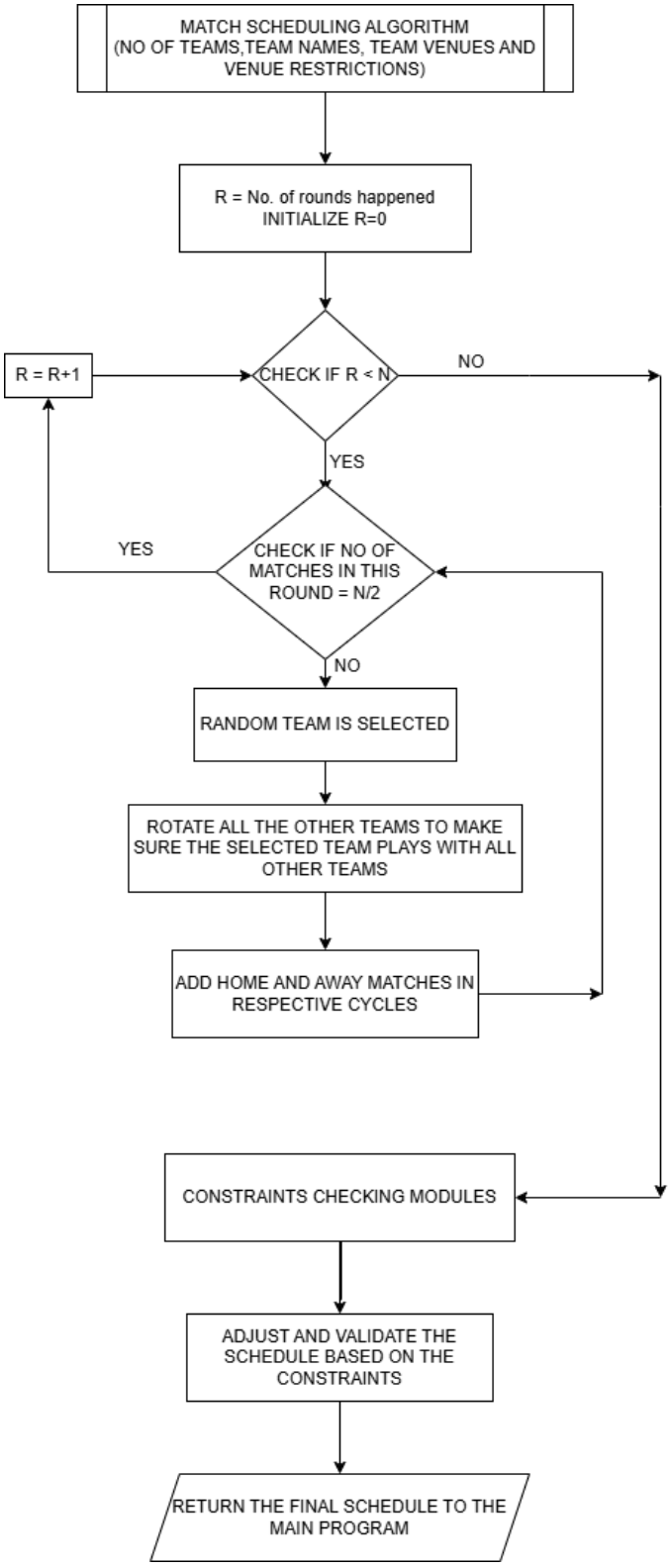


Figure 7.3: MATCH SCHEDULING ALGORITHM FLOWCHART

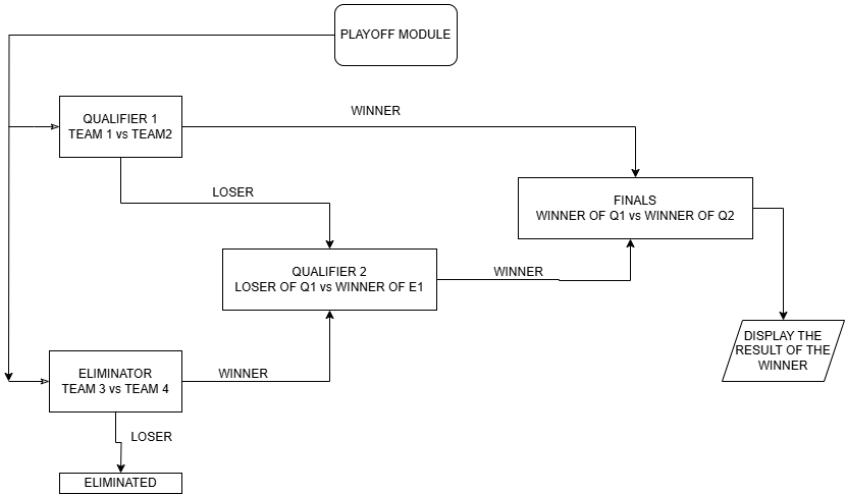


Figure 7.4: PLAYOFF MODULE FLOWCHART

The system for scheduling and simulating a sports tournament follows a structured flow across multiple stages. It begins with the Constraints Checking Module, ensuring matches adhere to predefined rules like team availability, venue constraints, rest periods, and home-away balance before locking fixtures.

The Main Scheduling Process then initializes by reading the number of teams, home venues, and constraints before generating a preliminary match schedule, which can be customized before finalization. Once confirmed, the system initializes the points table and proceeds to the Match Simulation Module, which loops through all fixtures, predicting winners, generating scores, calculating Net Run Rate (NRR), and updating the points table, awarding three points for a win and one for a draw.

The top four teams move to the Playoff Module, where a structured knockout format determines the ultimate winner through Qualifier 1, an Eliminator, and a final match. Throughout the scheduling process, the Match Scheduling Algorithm ensures balanced fixture allocation by iterating through rounds, selecting teams randomly while maintaining a fair home-away cycle, and validating constraints before finalizing the match calendar. This entire system ensures an organized and fair tournament structure, from initial scheduling to the final winner declaration

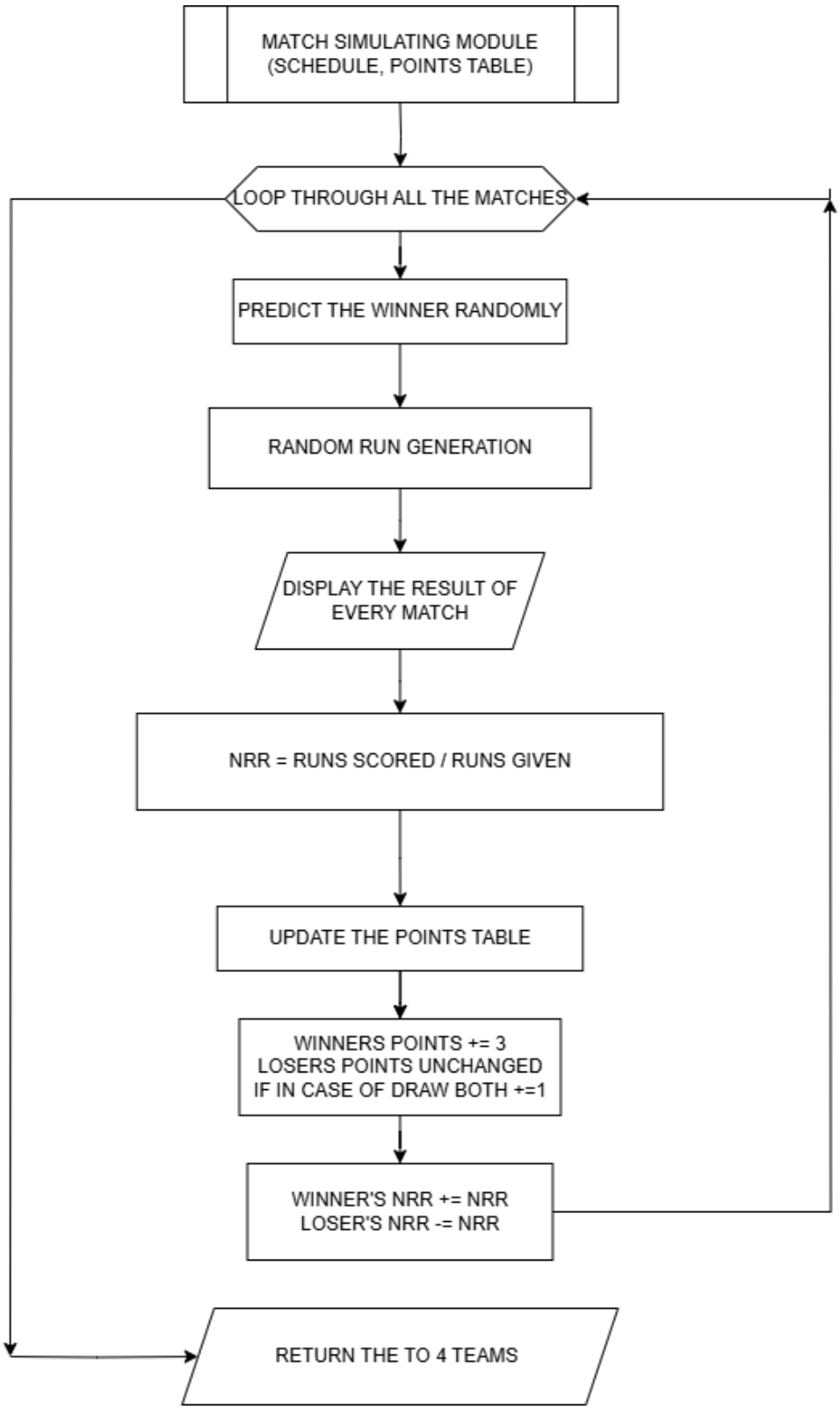


Figure 7.5: MATCH SIMULATION MODULE

# 8 MODULES AND CONSTRAINTS

**The Main Program** acts as the central controller, managing interactions between all other modules by taking user input, setting up the scheduling process, running match simulations, and handling the playoff stage if applicable. It starts by receiving user-defined configurations such as the number of teams, scheduling preferences, and constraints, along with match data and team information.
Once initialized, it calls the **Match Scheduling Algorithm**, which generates an optimized match schedule by considering factors like team availability, venue constraints, and time slots. This scheduling process aims to minimize conflicts and ensure a balanced fixture list. Before finalization, the schedule is passed to the **Match Scheduling Constraint Checker**, which validates it against predefined constraints, ensuring fairness by preventing teams from playing back-to-back matches without sufficient rest, checking venue availability, and resolving scheduling conflicts. If any issues are found, the schedule is adjusted and rechecked until it meets all requirements.
Once a valid schedule is in place, the **Match Simulator Module** takes over, running match simulations based on team statistics, probability models, or predefined rules to generate results, including scores and winners. The outcomes from these simulations are then processed to determine the standings or progressions. If the tournament includes a playoff phase, the **Playoff Module** is triggered, organizing knockout rounds such as quarterfinals, semifinals, and finals. This module ensures that teams advance correctly based on their previous results and adheres to the tournament's playoff format, whether it follows a knockout, double-elimination, or other structure. By efficiently managing each phase, from scheduling to match simulations and playoffs, this system ensures a structured, fair, and seamless tournament execution.

# 9 IMPLEMENTATION

- **Data Organisation**
  The data organization in this project is designed to be efficient and structured. Information regarding the teams, matches, and points is organized using structured arrays. Each team is represented with key details such as its name and home venue, ensuring that the program can easily reference and track team-specific data. Similarly, each match is recorded with essential information such as the teams involved, the venue, date, and time, enabling the system to manage the schedule effectively. Points are tracked for matches played, including wins, losses, total points, and run statistics, which are crucial for evaluating team performance. Additionally, the match schedule is saved externally, allowing for easy access and modification.

- **Libraries used**
  The project utilizes several libraries to streamline development. Standard libraries are employed for fundamental tasks like input/output operations, random number generation, and string handling. To handle specific functionalities related to playoff matches, a custom header file is included, ensuring modularity and clean code management. Time functions are also incorporated to guarantee randomness in match scheduling, which adds variability to the program's output and ensures a more realistic experience for users.

- **UI DESIGN**
  For user interaction, the program features a simple command-line interface, which allows users to engage with the system through a text-based environment. After generating the match schedule, users can access and view schedules for specific teams or venues. The final results, including points tables, are displayed directly in the console and saved to a text file for record-keeping. To further enhance user experience, the program incorporates a menu-driven interface, providing an easy navigation structure for users to select various operations seamlessly.

- **Platforms used**
  In terms of platform compatibility, the project is developed using the C programming language, making it versatile and able to run on any standard C compiler. It is designed to work in a console environment, ensuring it can be executed on various platforms like GCC, MinGW, Clang, Code::Blocks, and Visual Studio Code, all of which are commonly used for compiling and running C programs. This broad compatibility ensures that the project is accessible for users working in different development environments.

# 10 SOCIETY, ETHICS AND FUTURE

## 10.1 Observations with Respect to Society

The program simulates a full-fledged IPL-like tournament, incorporating aspects such as match scheduling, weather conditions, and team statistics. This reflects the importance of structured organization and planning in society. Sports, as portrayed through this simulation, contribute significantly to societal development by promoting teamwork, strategic thinking, and entertainment. The code also indirectly highlights how technology can be leveraged to improve operational efficiency in organizing large-scale events, thus mirroring real-world societal trends towards digitization and automation.

## 10.2 Legal and Ethical Perspectives

From a legal standpoint, this code does not violate any intellectual property rights since it merely simulates tournament mechanics without using any proprietary league names, logos, or commercial data. Ethically, the simulation provides a responsible way to engage users in a fictional environment, encouraging learning and sportsmanship without manipulating real-world outcomes. However, in a commercial scenario, ethical considerations would demand ensuring fairness, data privacy, and obtaining necessary permissions if real team names or brands were to be used.

## 10.3 Limitation and Future Enhancement

One limitation of the current program is its deterministic scheduling approach, which can lead to repetitive structures or inefficient travel itineraries despite optimizations. It also lacks user interaction during match simulations, as results are randomly generated without user control. Future enhancements could include integrating a graphical user interface (GUI) for better user experience, more sophisticated scheduling algorithms that incorporate real-world constraints, live match simulations allowing user inputs, and machine learning-based prediction models for match outcomes to add a layer of realism.

# 11 LEARNING OUTCOMES

Through the development and detailed analysis of this program, a wide range of fundamental and advanced programming concepts are thoroughly reinforced, including the use of structures, arrays, randomization techniques, file handling, and optimization strategies. The project not only consolidates technical skills but also bridges the gap between theoretical knowledge and practical application by simulating real-world event management and complex resource planning scenarios.

Students gain valuable insights into the design and implementation of modular programming, where clearly defined functions and data structures work cohesively to replicate intricate systems such as tournament scheduling. The simulation also enhances their understanding of dynamic decision-making processes, algorithmic problem-solving, and critical thinking, which are essential competencies in the field of software development.

Furthermore, the project emphasizes the significance of maintaining data integrity, efficient data organization, and scalability, preparing learners for challenges encountered in real-world software engineering and systems design. This holistic learning experience not only improves technical proficiency but also nurtures project management skills, attention to detail, and a deeper appreciation for the role of automation and optimization in modern technological solutions.

# 12 TIMELINE

The project begins with the Setup & Research phase (Week 1-2), where the scope of the system is defined, ensuring clarity on objectives, functionalities, and constraints. During this time, various match scheduling algorithms, such as round-robin, Swiss system, or optimization-based approaches, are researched to determine the most effective method for scheduling matches efficiently. Additionally, a suitable development environment is set up with necessary tools, frameworks, and libraries to support seamless implementation. Once the foundation is established, the Module Development phase (Week 3-5) begins, where the core components of the system are implemented.
In Week 3, the Match Scheduling Algorithm is developed, focusing on generating schedules based on team availability, venue constraints, and time slots while minimizing conflicts. Week 4 is dedicated to the Match Scheduling Constraint Checker, which ensures fairness in scheduling by preventing back-to-back matches for teams, verifying venue availability, and resolving conflicts. In Week 5, the Match Simulator Module is developed to simulate match results based on team statistics, probability models, or predefined rules, followed by the implementation of the Playoff Module, which handles the knockout stages, ensuring correct team progression based on results.
The Integration & Testing phase (Week 6-7) starts in Week 6, where all modules are integrated, ensuring smooth interaction between scheduling, validation, simulation, and playoffs. Unit testing is conducted for each module to verify individual functionality, while system-wide testing ensures end-to-end operation.
In Week 7, debugging and optimization take place, focusing on performance improvements, fixing scheduling inconsistencies, and refining match simulations for better accuracy. Finally, the Finalization & Deployment phase (Week 7-8) begins, where Week 7 involves final testing to confirm the system's stability and adherence to requirements. In Week 8, comprehensive documentation is created, including user guides and technical references, and the system is prepared for deployment, ensuring a fully functional, optimized, and reliable tournament management solution.

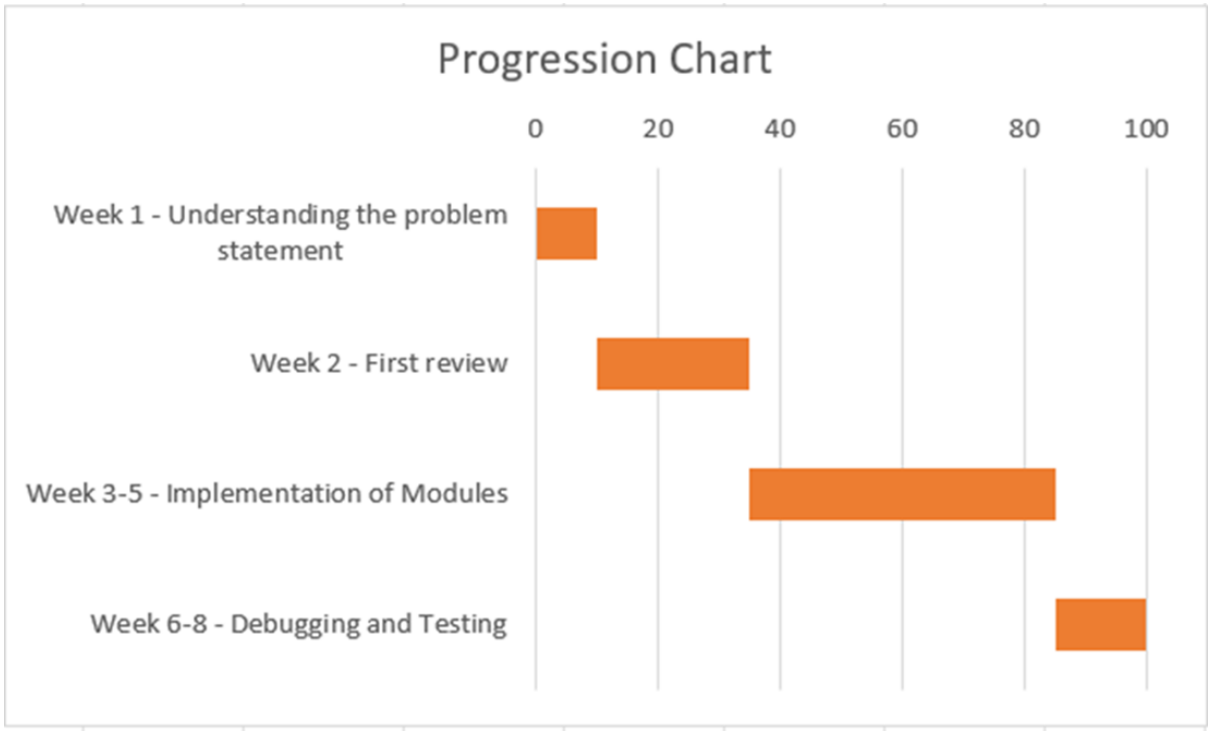


Figure 12.1: TIMELINE GRAPH

# Bibliography


[1] BCCI Official Documents on IPL Scheduling Guidelines. Board of Control for Cricket in India. Retrieved from `https://www.bcci.tv`

[2] Chakraborty, S., & Gupta, R. (2019). Optimization Strategies for Sports Scheduling in India. *Journal of Sports Management & Analysis*, 15(4), 112–129.

[3] Kumar, A., & Iyer, P. (2021). Cricket Tournament Scheduling: Balancing Logistics and Viewership. *Indian Journal of Sports Science & Analytics*, 9(2), 45–67.

[4] Desai, M., & Rajan, K. (2020). Computerized Scheduling Systems for Large-Scale Tournaments. *International Conference on Computational Sports Analytics, IIT Madras.*