**CRICKET MANAGEMENT SYSTEM**

Software Engineering

Project report

Submitted by: Supervisor:

**PRIYANSHU GUPTA (21547) Ms. Kavita Rastogi PIYUSH UPADHYAY (21545)**

**VINIT KUMAR MEENA (21565)**



Department of Computer Science

Shaheed Sukhdev College of Business Studies

University of Delhi

Acknowledgement

Table of Contents

Problem Statement

Process Model

1. Software Requirement Specifications
   1. .1 Introduction

1.1.2 Feature

1.1.3 Purpose

* + 1. Scope

1.1.5 Overall Descriptions

* + 1. Product Prospective

1.1.6 Constraints, Assumptions and dependences

* 1. Specific Requirement
     1. Interface Requirement

1.2.2 Functional Requirement

1.2.3 Non- functional Requirement

1.2.4 Design Constraints

* 1. Data flow Diagram
  2. Data Dictionary

2. Size estimation and scheduling

* 1. function point estimation
  2. size estimation
  3. cost estimation

1. Project Scheduling
2. Timeline Chart
3. Risk Management
4. Design
   1. Use case Diagrams
   2. Architectural Design
   3. Data design
5. Implementation of module
   1. screenshot of software
6. Testing
7. References

**Problem Statement**

CRICKET MANAGEMENT SYSTEM is a software application to manage all the activities concern to the cricket. This project will help the cricket organisation to store all cricket related information. In India cricket is considered as a religion, cricket is the most popular game in the world wide.

Information is available country wise and player wise information. There are various format of cricket such as IPL, TEST, ODI, T20, matches and other types of matches.

By entering the data of each match. We can get all type of report. Such information are number of win match and lost match. Everything will be on your dashboard. Such as team performance of match can be obtained.

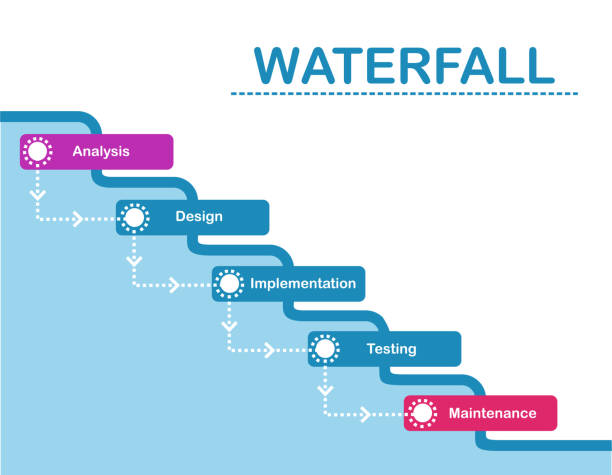
This software provide you score card of match with each player’s performance is very difficult task do manually so we digitalizing the whole process through software

1. **Process Model**

Waterfall model is an sequential model that divides software development into predefined phases. Each phase must be completed before the next phase can begin with no overlap between the phases.

The model is used for the following reason

* Project is to sort
* Requirement is clear
* Environment is stable
* Resources are available and trained
* Requirement are not changing frequently



**Software requirement specification**

**1.1.1 INTRODUCTION**

In cricket management software provide you complete information of country wise information of players details. By the entering the data of match you can get all type of report instantly that is related to the match such as history of match and match performance. Number of matches win by team and lost. And you get complete statistics of matches of national and international matches. And all types of news that are related to the cricket are also available.

**1.1.2 FEATURES OF CRICKET MANAGEMENT SOFTWARE**

* Team’s information
* Cricketer information
* Cricketer personal details
* ODI and test ranking
* Commentary of each ball
* Upcoming series information
* International news and travel

**1.1.3 PURPOSE**

Purpose of cricket management software is to provide a comprehensive platform to manage Automate various cricket related tasks and activities. This software can be used by cricket organisations, teams, and manage the matches. This software provides a centralized platform to manages different aspects of cricket such as scheduling of matches, managing players and teams, tracking scores and statistics, generating reports, and financial transactions.

**1.1.3 SCOPE**

Cricket management software is designed to be user friendly software allowing non - Technical users to operate the system with ease. Cricket management software has a Significant scope in the sport industry. The cricket management software aims to automate. The tasks related to cricket management such as scheduling cricket match managing Player statistics, managing team and player profile, managing scorecards, managing much Results and many more

**1.1.4 OVERALL DESCRIPTIONS**

The overall descriptions of cricket management software is created and managing the Database, developing user friendly software to manipulate the database, and provide An authentication mechanism to safely accomplish task mentioned above

**1.1.5 PRODUCT PROSPECTIVE**

The product prospective of cricket management system software is essential to ensure That the software meets their requirements of users and operates effectively in ecosystem It includes elements such as functionalities, interfaces, constraints, context and Quality.

**1.1.6 CONSTRAINTS, ASSUMPTIONS AND DEPENDENCES**

Cricket management software must ensure the security of data and the access, hacking. And cricket management software must comply with the regulations of the cricket Governing bodies and the laws of country where system is deployed. Cricket management System software must have a fast response time, even when handling large volume of data

Cricket management system software assumes that users will have access to the internet to Use the system. And software assumes that user will have some knowledge of cricket and Rules of the cricket. And software assumes that user will have access to a device that meets Minimum hardware requirement to run the system

Cricket management system software depends on various data sources such as live match Data, player statistics and match schedules. And software depends on the third party. Libraries for various functionality such as data visualization.

* 1. **SPECIFIC REQUIREMENTS** 
     1. **INTERFACE REQUIREMENTS**

All the user will see same pages when they enter in the cricket management system Software. There will be 2 options are available one for the user one for the admin. Admin required to login through an id and password so to make changes in the Database or to information to it on other hand user retrieve information of different Players, along with personal details as well as after getting logged into Respective accounts.

* + 1. **SOFTWARE REQUIREMENTS**

Requirements that are related to the functional aspects of software fall into this Category. According to the cricket management system software we have

* View option is given to the user to view details of the favourite players as well as Personal details
* Retrieve function lets the user to offline download the details the players and the Teams
  + 1. **Functional Requirement**

Functional Requirement of cricket management system software would typically include:

1. Player management: the system should be able to manage players information, including their names, and contact details. It should also keep track of their performance statistics such as batting average and bowling average, field performance.
2. Team management: the system should be able to manage the schedule matches, date time locations.
3. Score card management: the system should be able to manage scorecards for each match individual player scores, team scores and match results.
4. Report and Analytics: the system should be able to generate reports and analytics, such as player performance and statistics, team ranking and match results.
5. Communication Management: the system should be able to manage communication between team and umpires and administrators, including other functionality
   * 1. **Non-Functional Requirements**

Some non-functional requirements that could be applicable to a cricket management system software:

1. Performance: The software should be able to handle a large number of users and data without any significant lag or delay.
2. Availability: The system should be available 24/7, with minimum downtime for maintenance or upgrades.
3. Security: The system should be scalable, meaning it should be able to handle increasing amounts of data and users without any significant impact on performance.
4. Usability: the system should be user-Friendly, with a simple and intuitive user interface.

**1.2.5 DESIGN CONSTRAINTS**

1. HARDWARE REQUIREMENTS

The cricket management system software will be storing of 1000 players personal Data which will roughly require 1gb of data storage space

2. SOFTWARE REQUIREMNTS

1. since cricket management system is an android based application so the user will Require mobile for the deployment of the software.

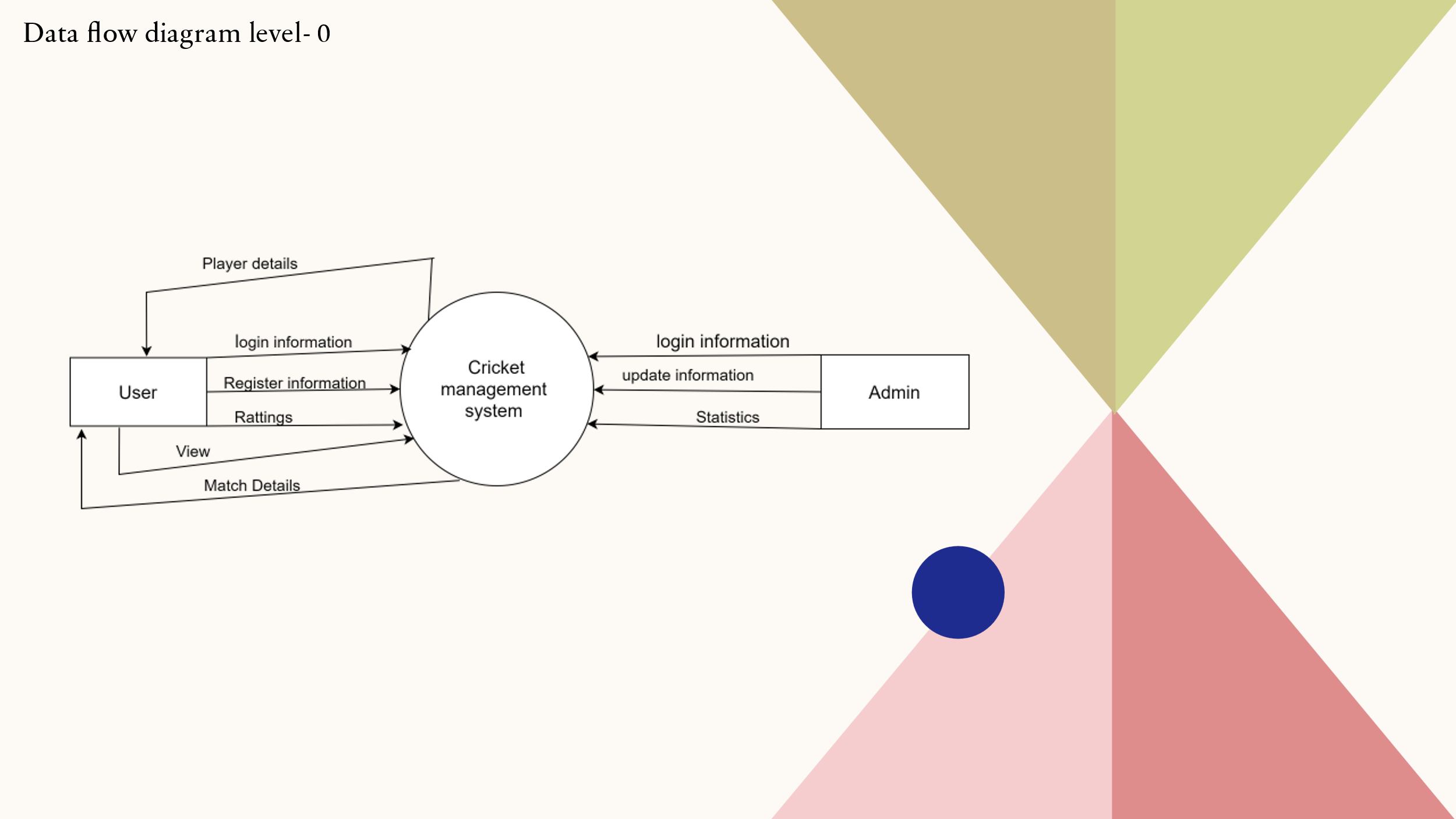
2. Application will run on 256mb or higher RAM.

3. it require a space of 2.5 mb to deploy onto the machine

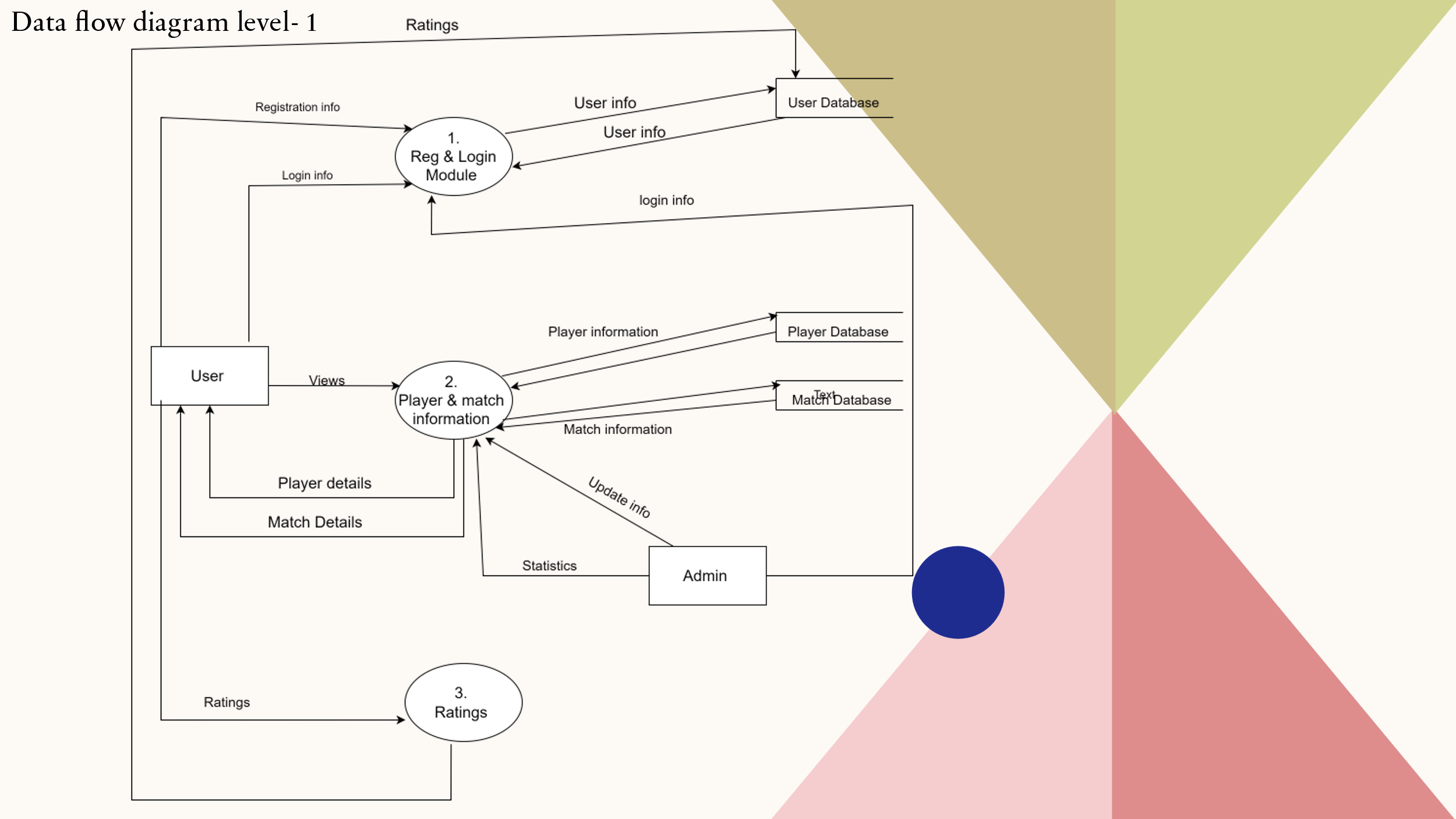
4. the cricket management system software will work properly on any mobile set Android version or higher

* 1. **Data flow Diagram**

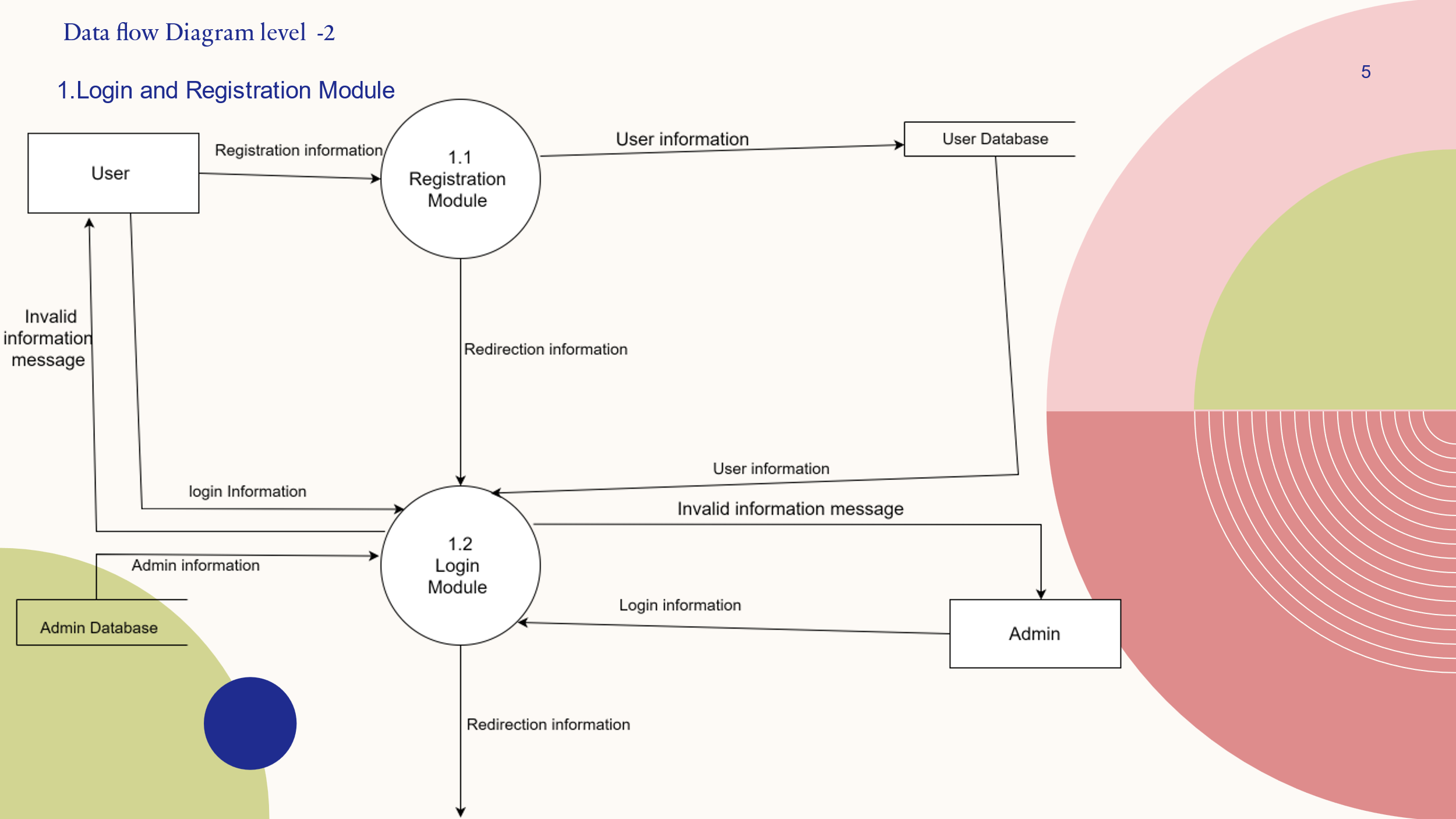
**Context level diagram**



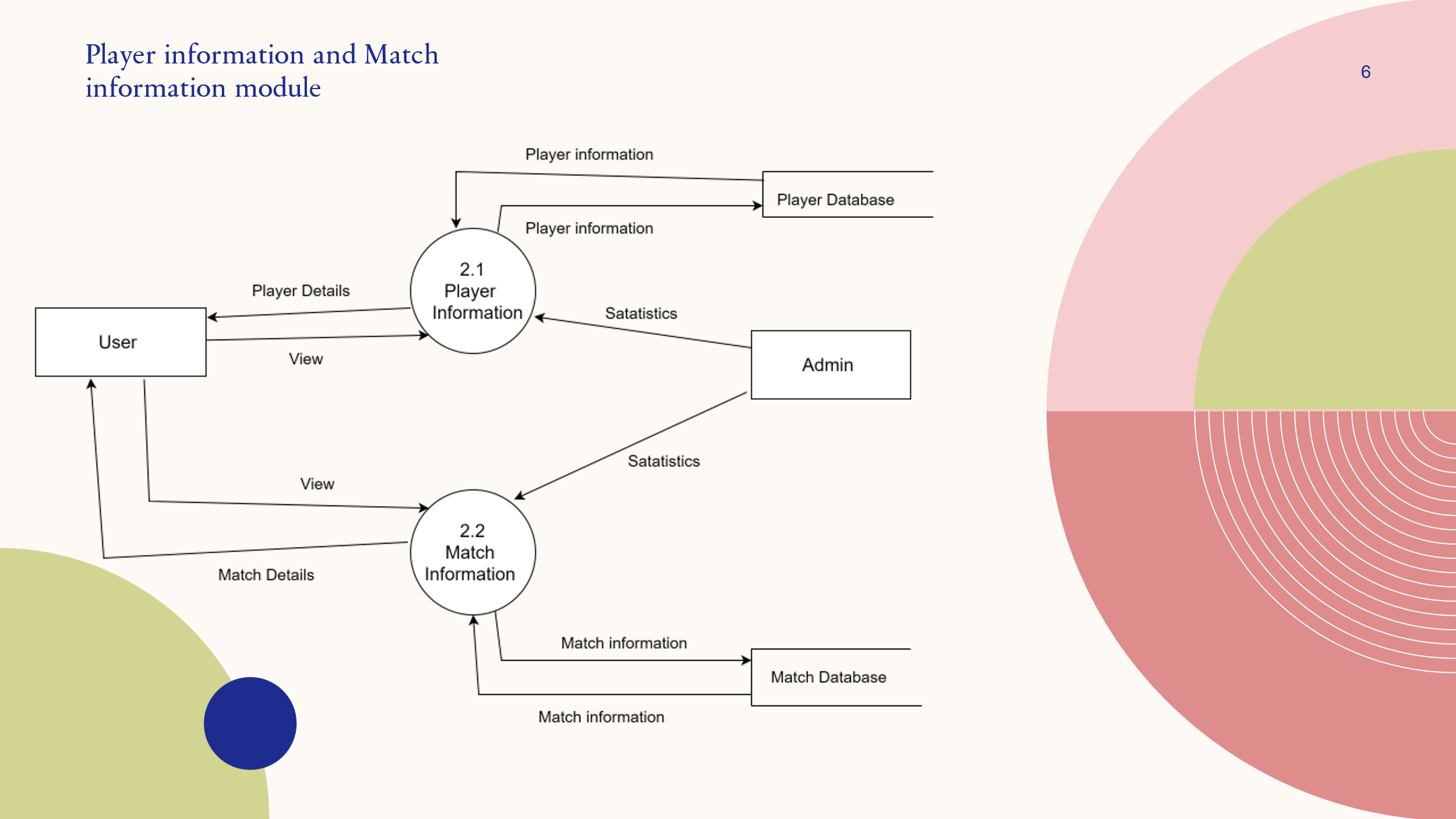
**DFD level – 1**



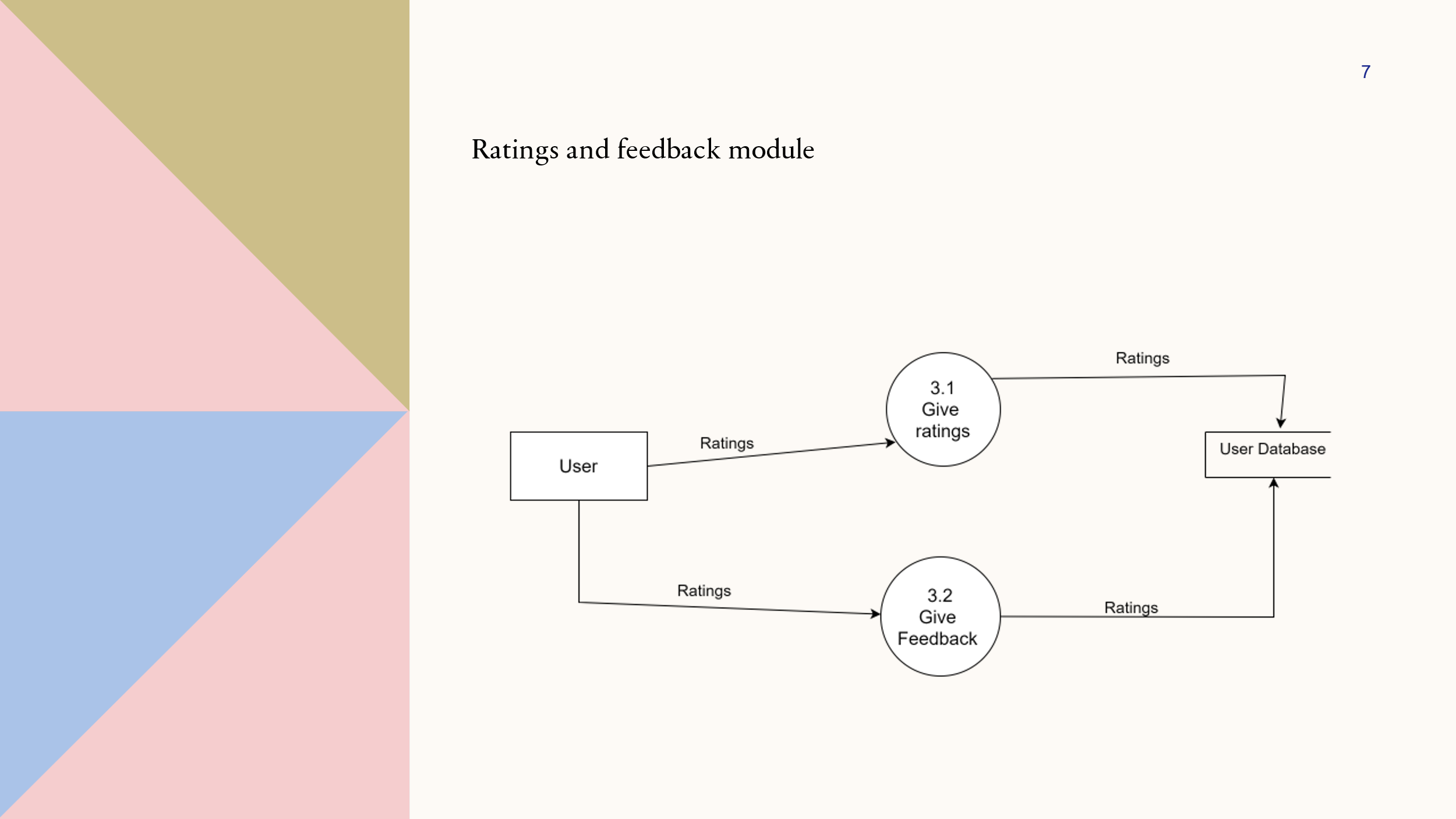
**DFD level 2**



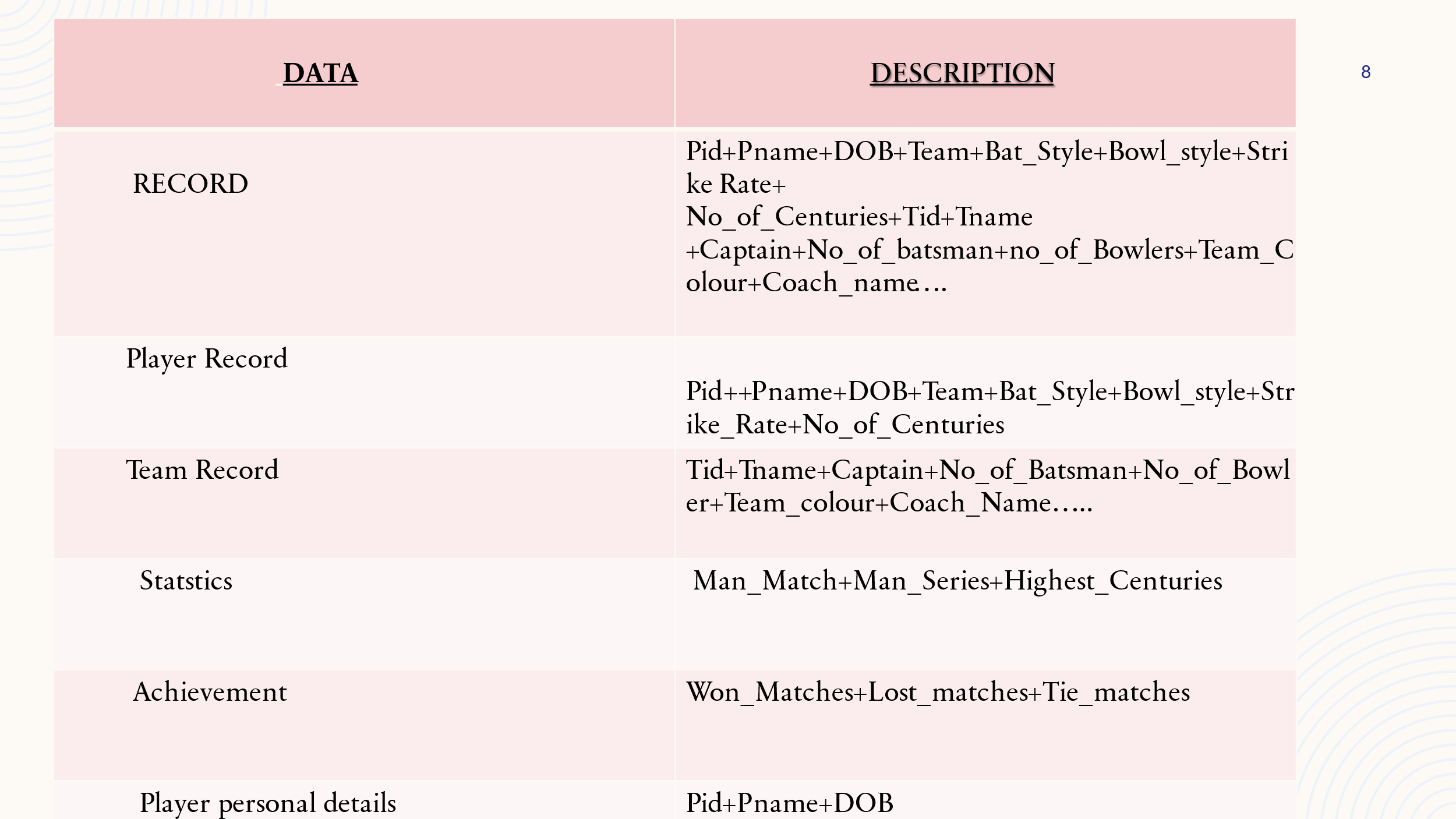
**b)**



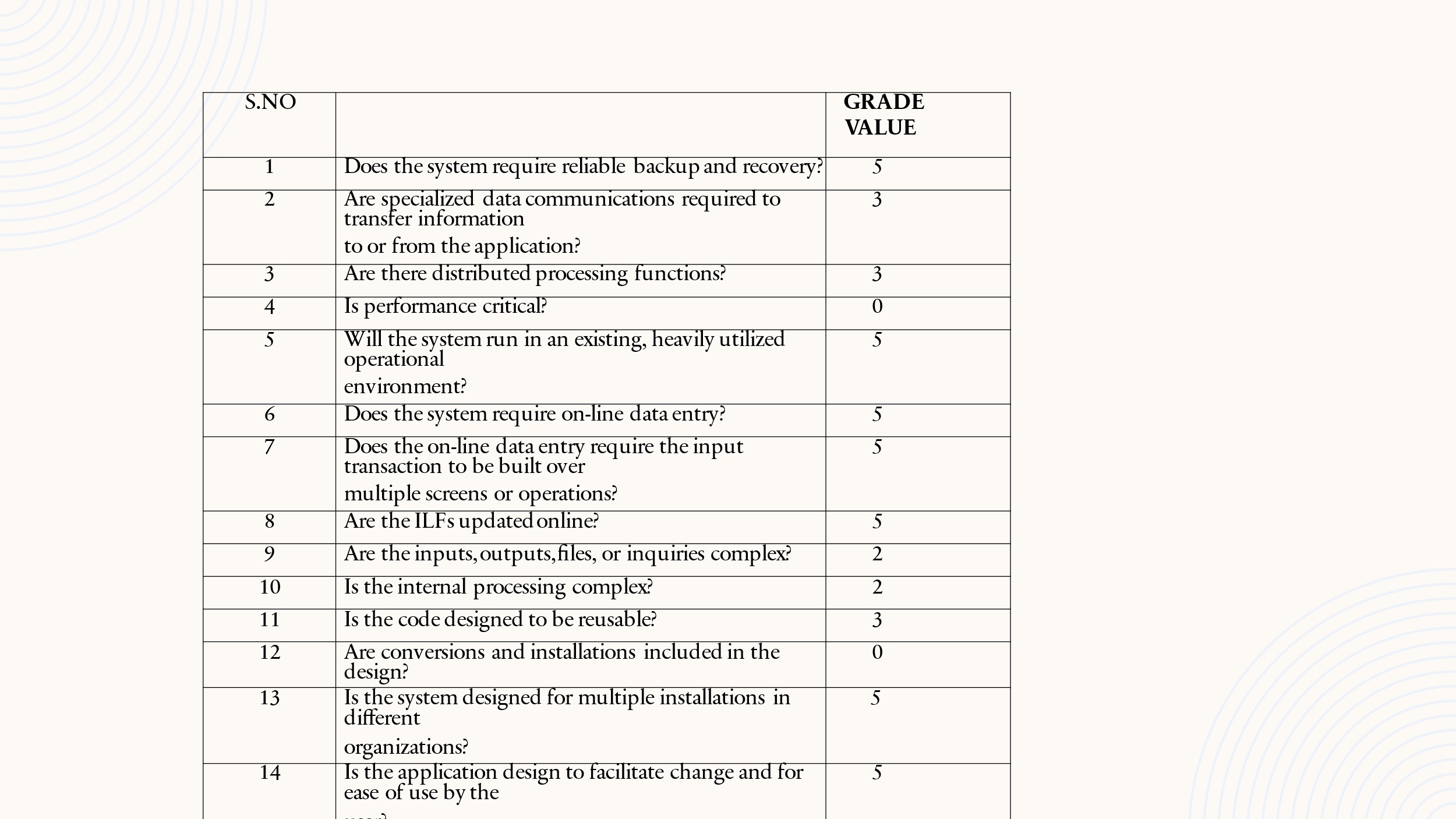
**C)**



* 1. **Data Dictionary**

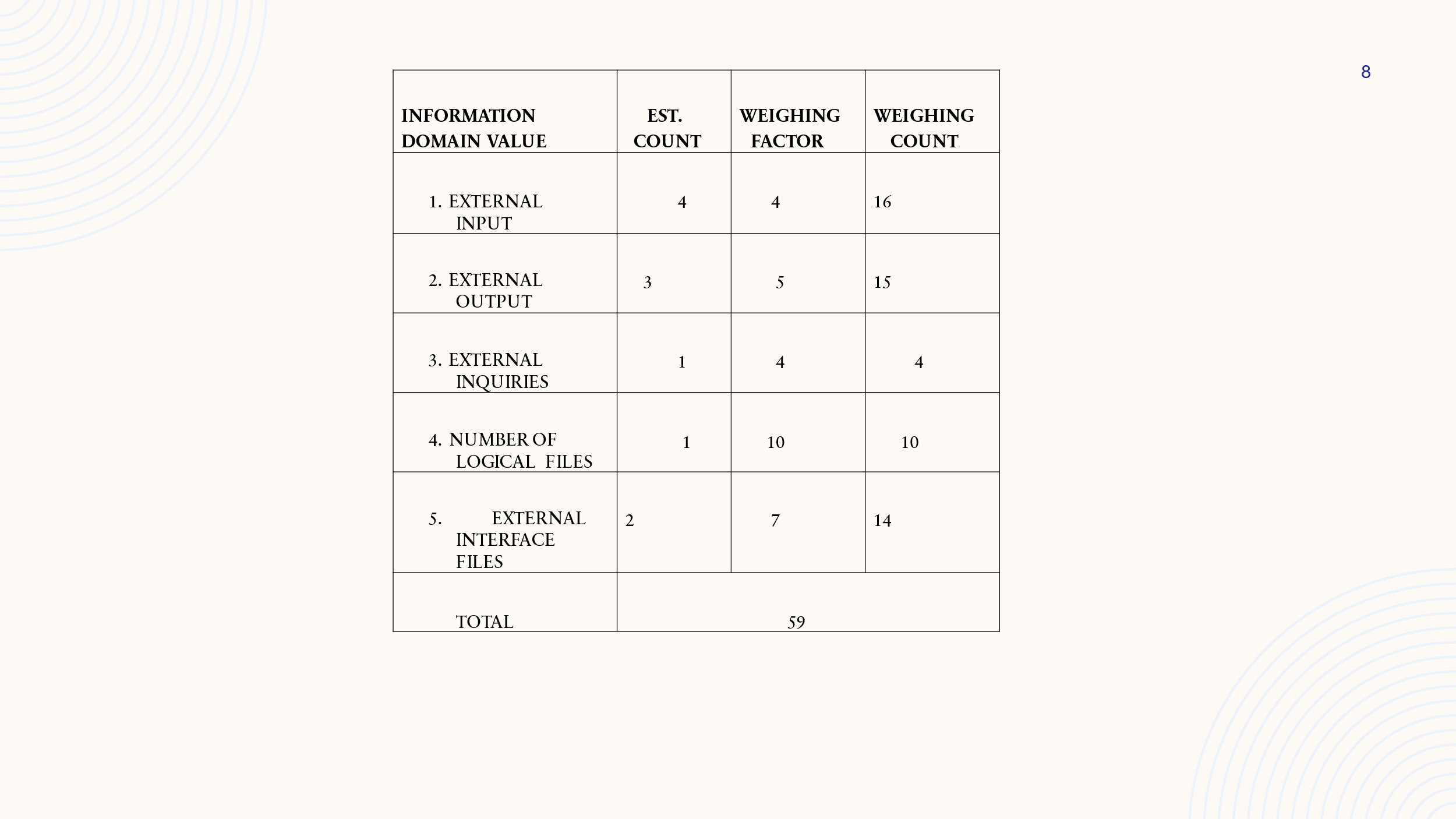


1. **Size estimation and scheduling**



**Value Adjustment Factors = ∑fi = 48**

1. **Function point Estimation**



**Computing function points**

Fp = Count Total \*(0.65+001\*∑fi)

= 59\*(0.65+0.01\*48)

= 66.67

1. **Size Estimation**

Assuming that Average Productivity for such a system is 6.5Fp/pm i.e functional point per person month, then estimators are

Size = totalFP’s / Average Productivity

= 66.67/6.5

= 10.25pm i.e 11 person month

**C) Cost Estimation**

Assuming that the cost is Rs 7000/pm i.e 7000 per person month.

Cost = Cost per person month \* effort

= Rs(7000\*11)

= 77,000 Rs

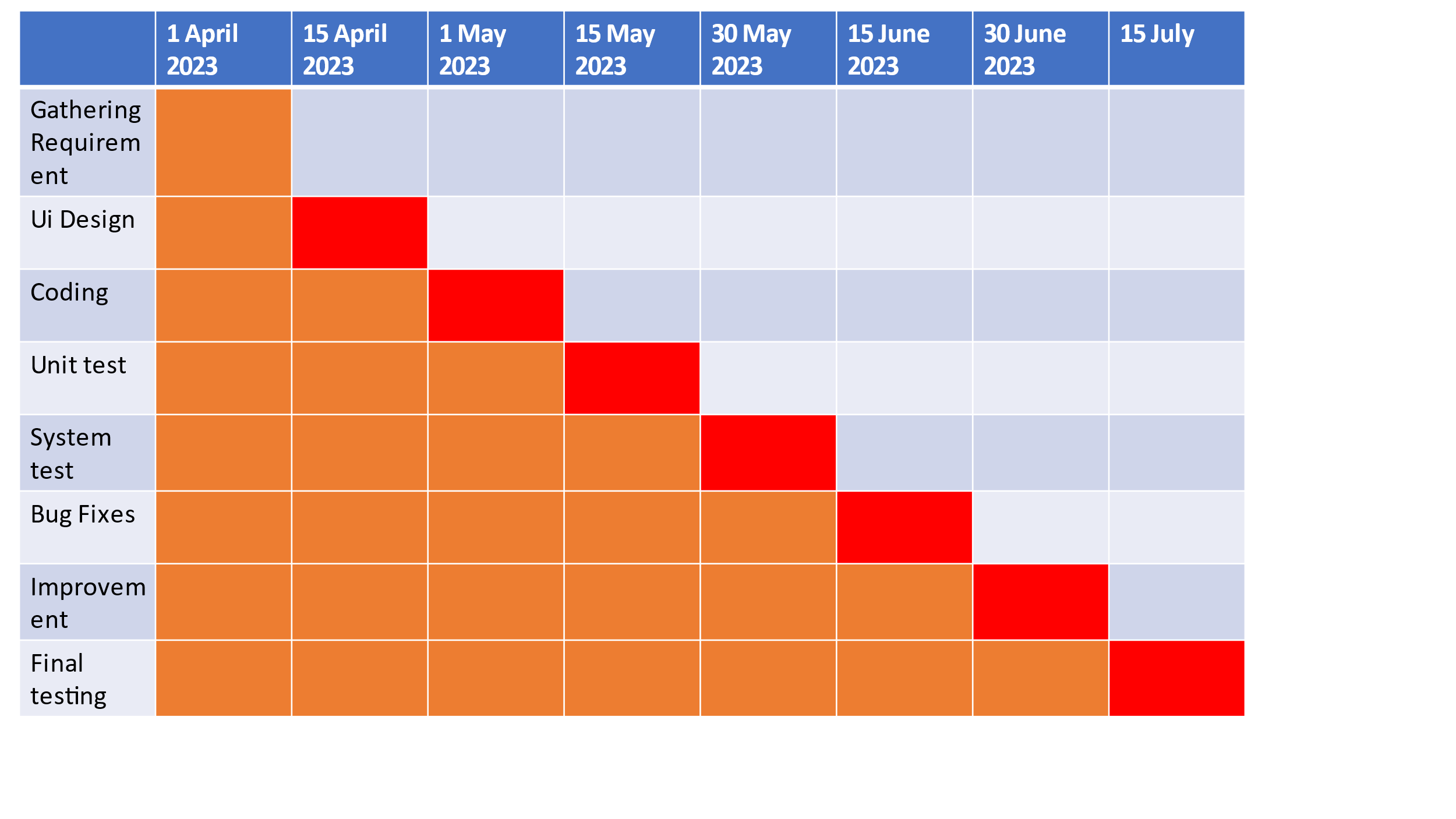
1. **Project Scheduling**

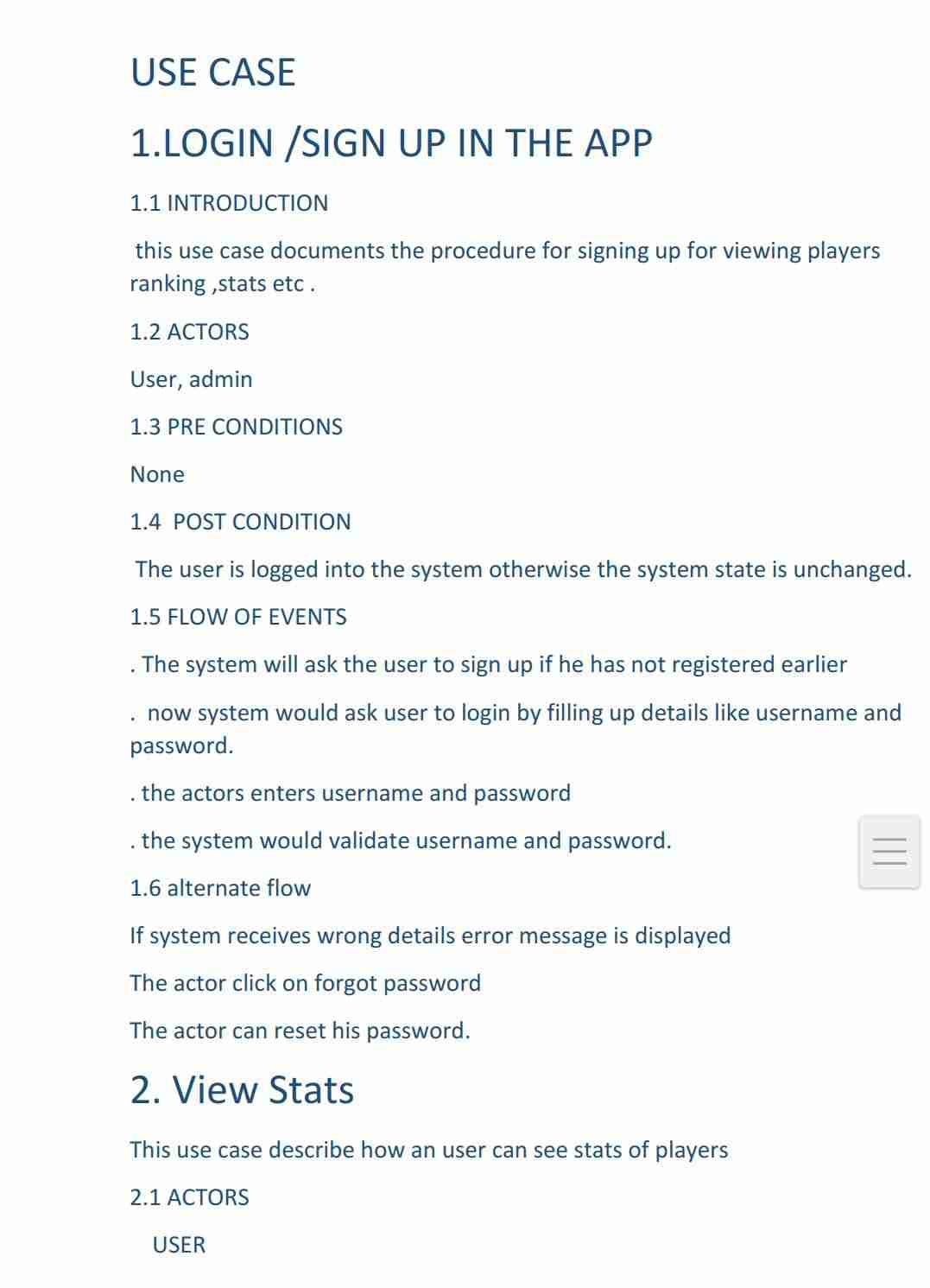
Scheduling is a planning activity that a primary component of software project management.

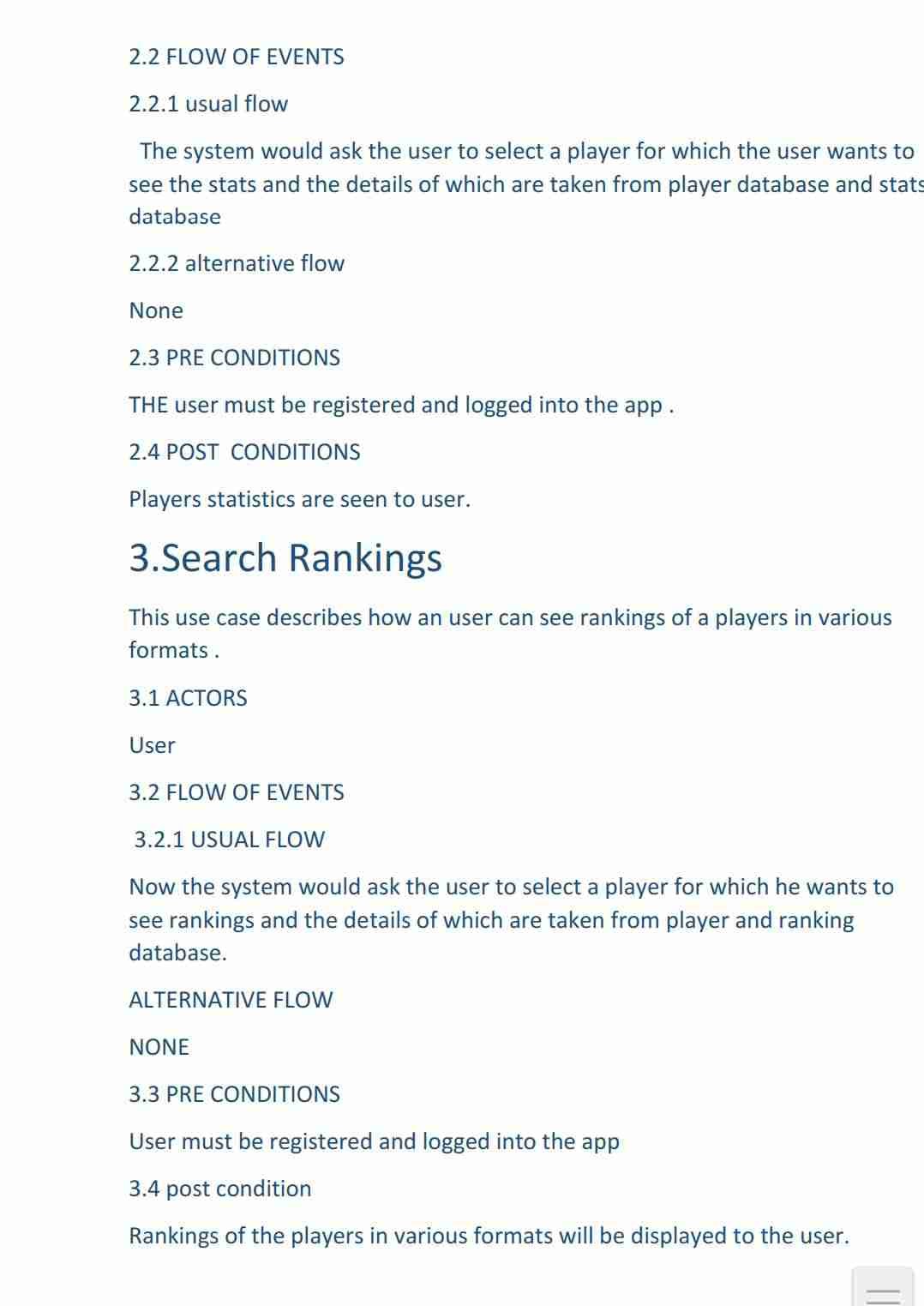
When Combined with estimation methods and risk analysis, scheduling establishes a road map for the project manager.

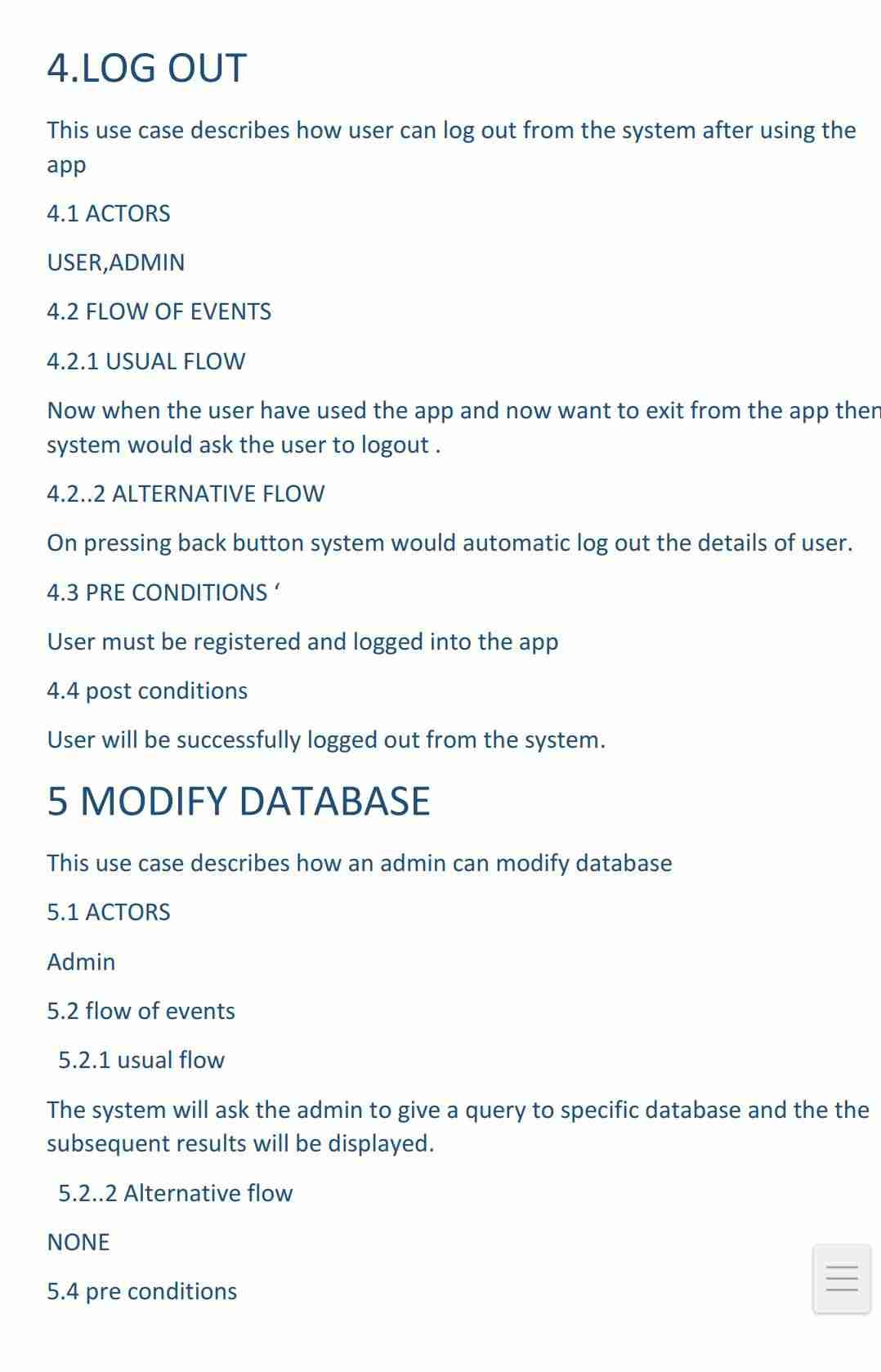
The characteristics of the project are used to adapt an appropriate task set for the work to be done. A task network depicts each engineering task.

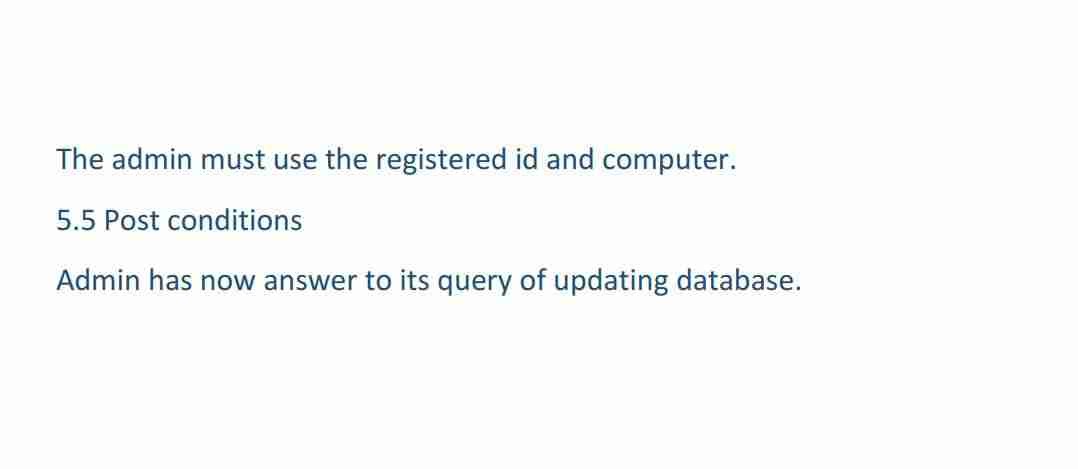
1. **Timeline chart**







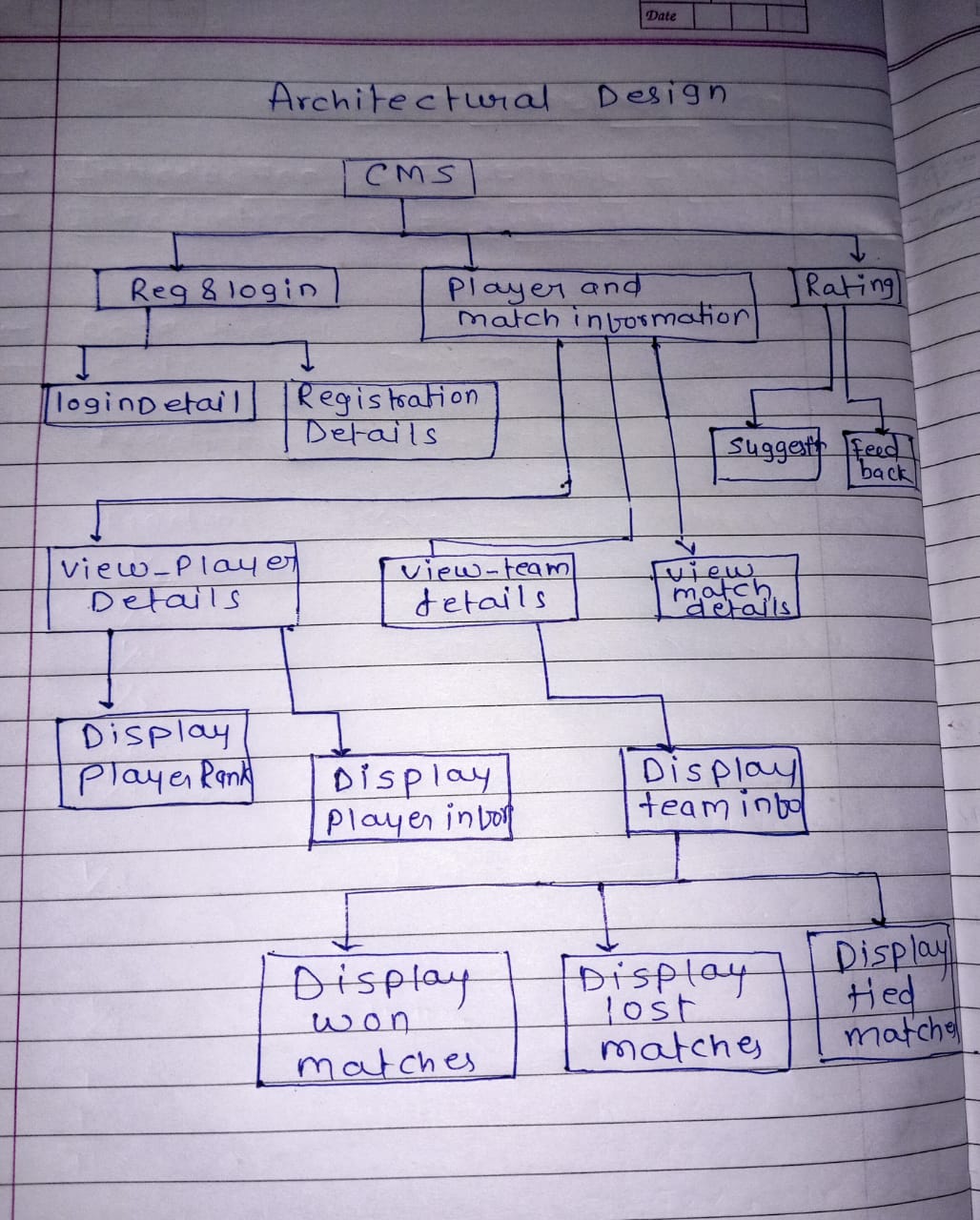


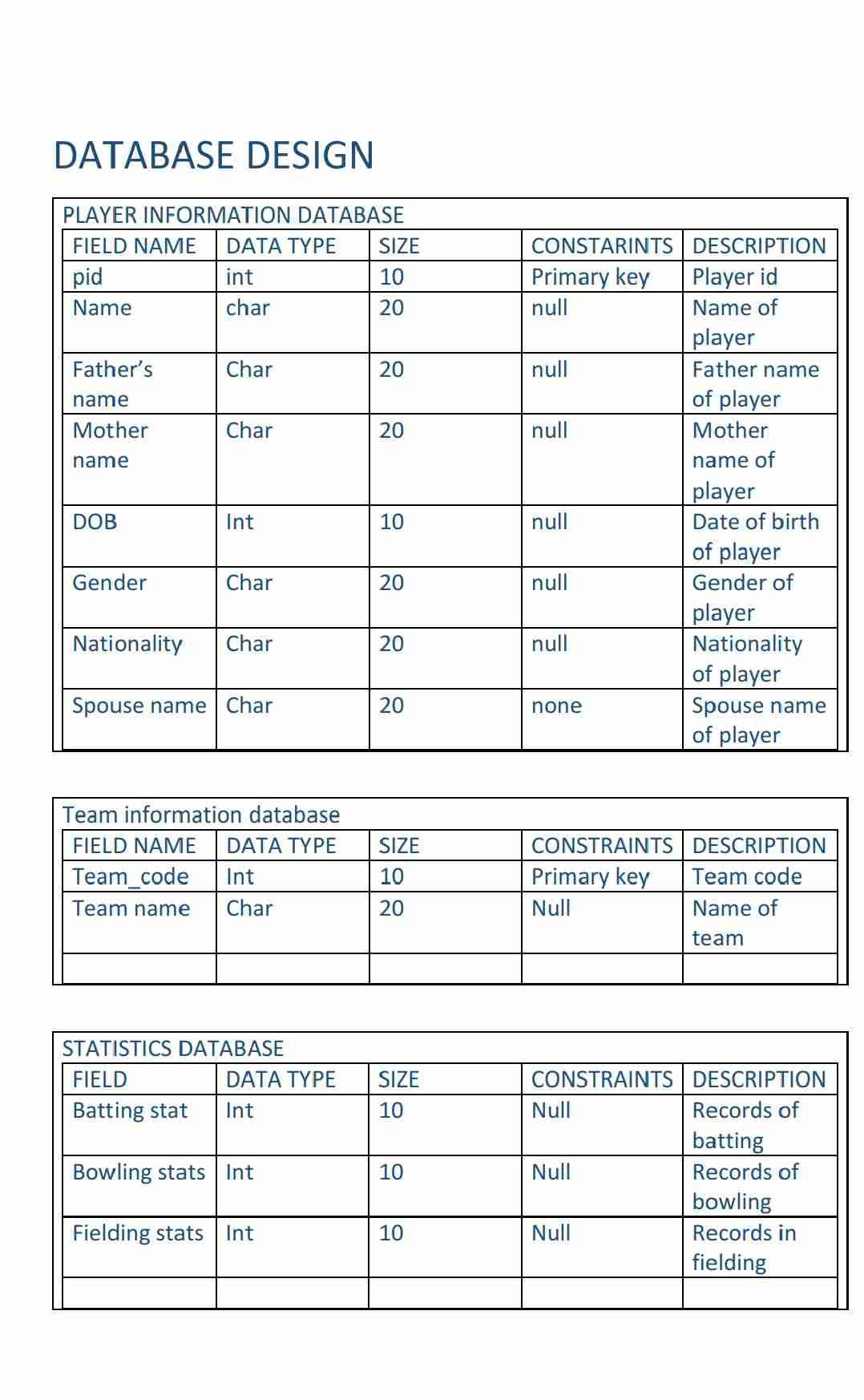


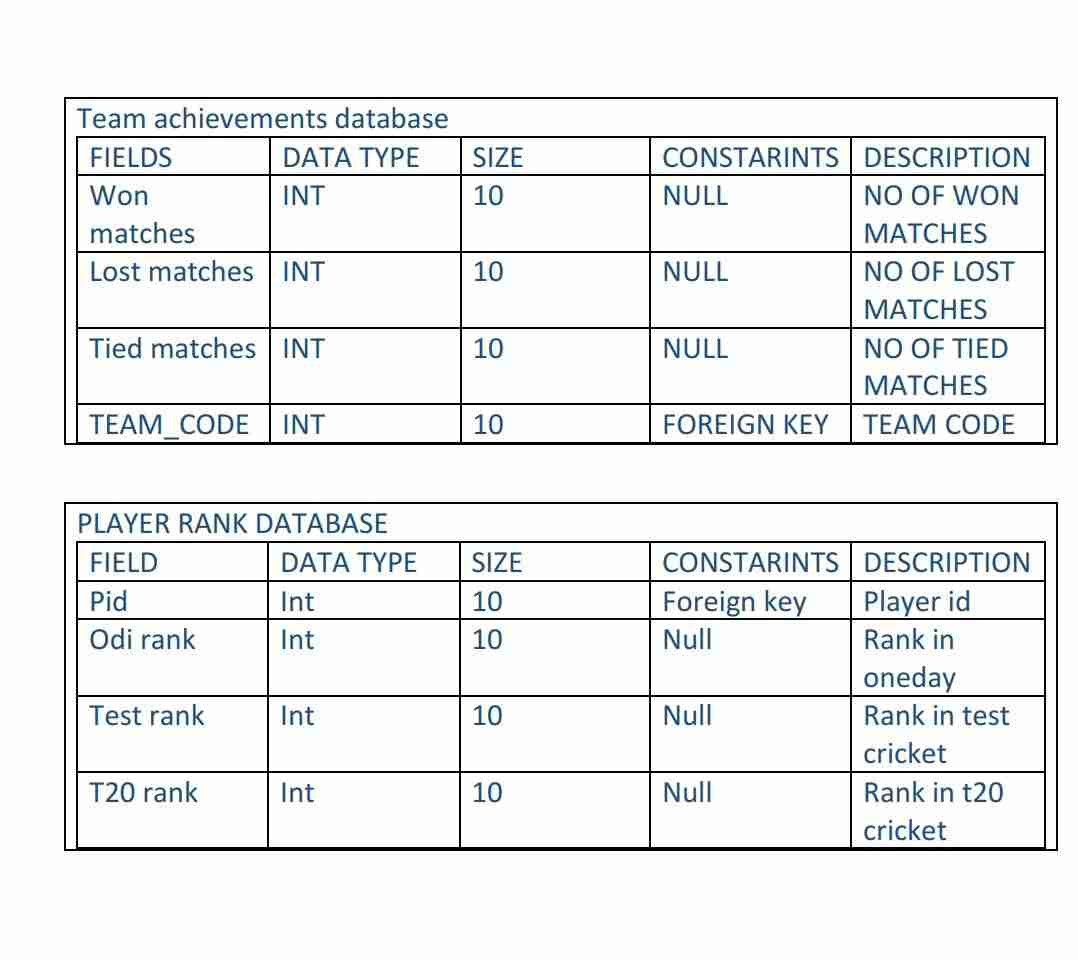
USER

ADMIN

**12. Architectural design**







**7. Implementing of Module**

The architectural design of the software defines every module of the application in details in a hierarchical way. Cricket management system software divided into three modules.

1. **Registration and login module**

In a registration module user first login and fill the personal information and the information stores in the User database.

After that user can login from the username and password and see information that are related to the cricket management system.

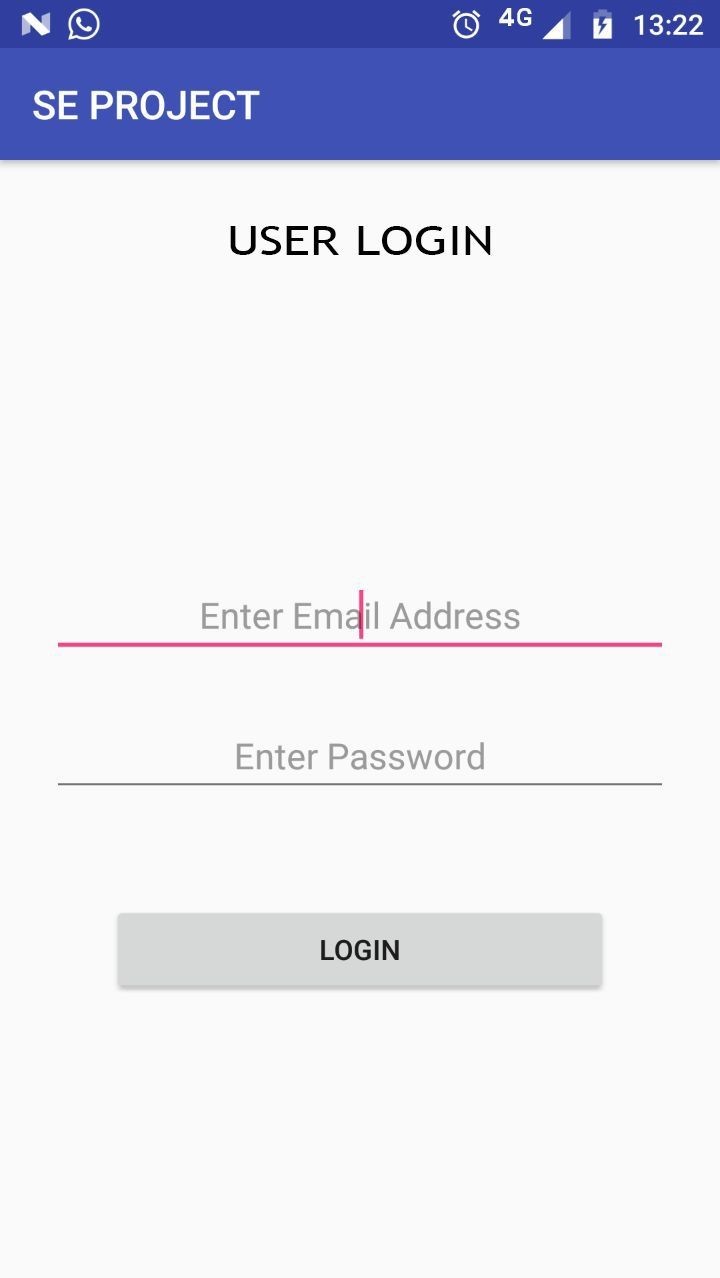
1. **Player and Match Information Module**

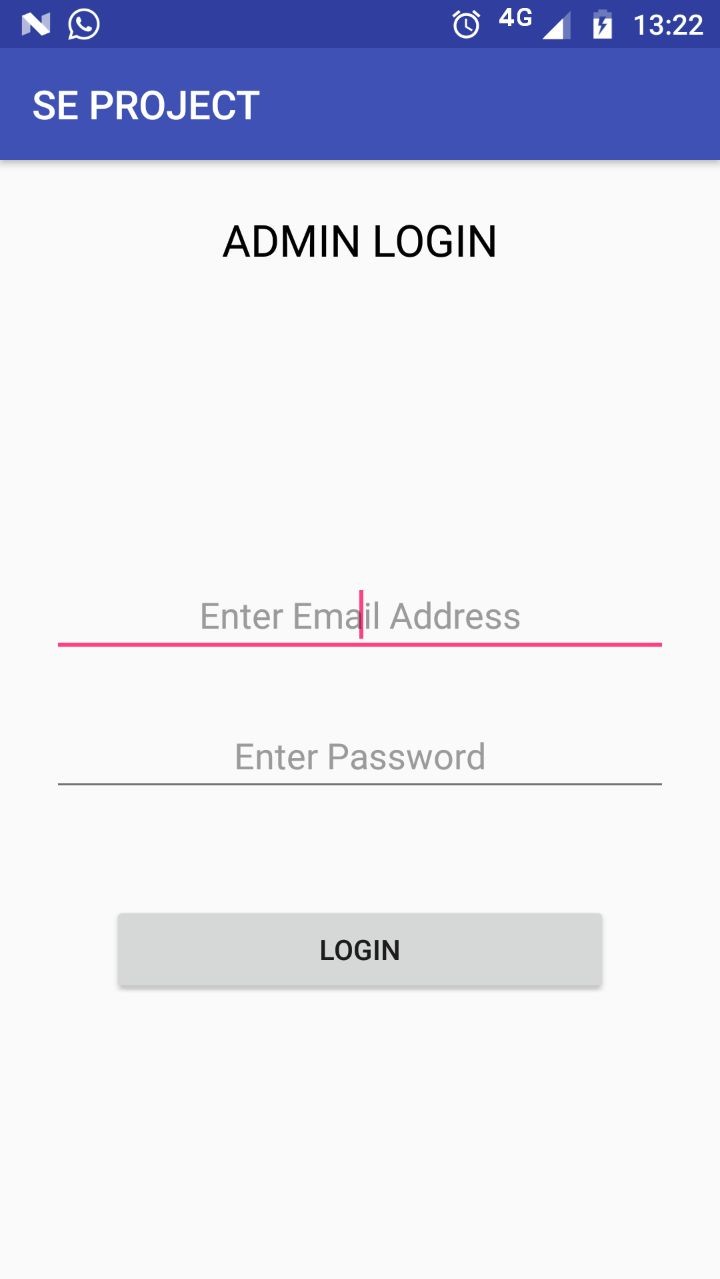
In this Module User can see details of player and match details and information of match and player are stored in the database At the admin end Admin can update statistics of player details and match details.

1. **Ratings Module**

In this module User give their feedback and also give Rating to software also.

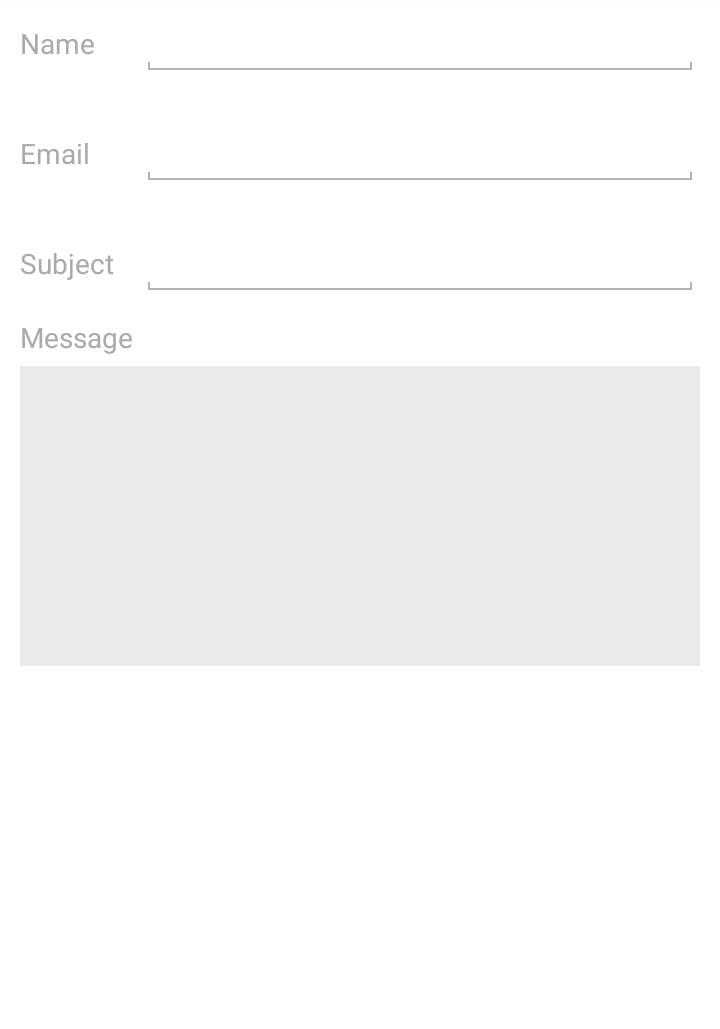
B) Screen Shot of the software







**SE PROJECT Feedback**

****

**Risk Analysis**

Risk analysis for a cricket management system involves the process of identifying, assessing, and prioritizing potential risks that could affect the success of the system. This includes identifying potential technical, data, operational, legal, financial, reputation, and regulatory risks that could impact the system's performance, security, and compliance. Once the risks have been identified, the next step is to assess their likelihood and potential impact, and then prioritize them based on their severity and likelihood of occurrence. Finally, the risk analysis process involves developing and implementing effective strategies for mitigating the identified risks, which could include contingency plans, risk mitigation strategies, and other measures to minimize the impact of the risks on the system's success.

Here we are proposing such risks for our project-

* Software Malfunction – Technical risks are a potential threat to the success of a cricket management system, including hardware failures, software bugs, and compatibility issues with different operating systems. Hardware failures can occur due to various reasons such as power outage or hardware malfunction, which can result in the system being unavailable, causing delays in processing data, or making the system unusable. Software bugs are defects in the software code that can cause errors, data corruption, or system crashes. Compatibility issues can arise if the system is not designed to work with specific operating systems or software applications. To mitigate these technical risks, the system needs to be thoroughly designed, tested, and maintained before deployment. Proper documentation, training for users, and regular backups of the system data can also help prevent errors and data loss.
* Data loss- A cricket management system may face risks related to the accuracy, completeness, and security of the stored data, such as data loss due to hardware or software failures, unauthorized access, or cyber-attacks. This could result in the loss or corruption of critical data, affecting the system's performance and availability. Mitigation strategies include implementing appropriate security measures like encryption, access controls, and backups, to prevent unauthorized access or data loss. Regular maintenance and updates to the system can help to minimize the risk of hardware or software failures, while employee training and awareness can prevent cyber-attacks.
* Inadequate Resource allocation - Day-to-day risks in a cricket management system include user errors, lack of training, and inadequate maintenance. These risks could lead to data inaccuracies, system downtime, or user frustration. Mitigation strategies include proper user training and education, regular system maintenance and updates, and having contingency plans in place to address system failures.
* Budgetary Uncertainty- Non-compliance refers to the failure to follow or adhere to laws, regulations, policies, or standards that are applicable to an individual or organization. It could result in penalties, fines, legal liabilities, and reputational damage. Non-compliance can occur due to ignorance, negligence, or deliberate actions, and it is important to ensure that individuals and organizations comply with applicable regulations to avoid adverse consequences.
* Public Scandal or controversy- The system may pose a risk to the reputation of the organization, such as negative publicity due to security breaches, data leaks, or system downtime.
* Rules and Regulations - The system may pose a risk to the reputation of the organization, such as negative publicity due to security breaches, data leaks, or system downtime.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Risk Number | Risk | Category | Probability | Impact |
| 1 | Software Malfunction | Technical risk | 30% | 3 |
| 2 | Data loss | project risk | 20% | 3 |
| 3 | Inadequate resource allocation | Project risk | 15% | 2 |
| 4 | Budgetary uncertainty | Project risk | 20% | 3 |
| 5 | Public scandal or controversy | Business Risk | 20% | 4 |
| 6 | Rules and regulation | business Risk | 15% | 2 |

Impact table

|  |  |
| --- | --- |
| 1 | Negligible |
| 2 | marginal |
| 3 | Critical |
| 4 | Catastrophic |

RMMM PLAN

Financial Risk: -  
 Mitigation

* Conduct thorough testing and maintenance.
* Implement security measures and take regular backups.

Monitoring

* Monitor system health and performance.
* Monitor system logs and user activities in real-time.

Management

* Assign dedicated IT staff for system maintenance.
* Develop and implement a disaster recovery plan.

Data Risk: -

Mitigation: -

* Implement appropriate security measures like access controls and encryption.
* Conduct regular data backups and store them in a secure location.

Monitoring

* Monitor user access and activity logs to identify any unauthorized access attempts.
* Regularly audit and test the security measures in place to ensure they are effective.

Management

* Assign a data security officer to oversee the protection of data
* Develop and implement a data breach response plan to address any incidents promptly.

Operations Risk

Mitigation: -

* + - Provide proper user training and education to reduce the risk of user errors.
    - Conduct regular system maintenance and updates to minimize the risk of downtime.

Monitoring: -

* Monitor system performance and user feedback to identify potential issues.
* Monitor system logs and error reports to identify system failures and take corrective actions promptly.

Management: -

* Assign dedicated staff to oversee system operations.
* Develop and implement a contingency plan to address system failures and minimize the impact on users.

Financial Risk: -

Mitigation

* implement appropriate financial controls and procedures to prevent fraud.
* Regularly review financial reports and budgets to identify any discrepancies or irregularities.

Monitoring

* Monitor financial transactions and accounts regularly to detect any unauthorized or fraudulent activities.
* Conduct regular internal and external audits to ensure compliance with financial regulations and policies.

Management

* Assign a dedicated finance officer to oversee financial operations.
* Develop and implement a financial risk management plan to address financial risks and uncertainties that could impact the organization.

Reputation Risk: -

Mitigation

* Establish and follow ethical and professional codes of conduct for all stakeholders involved.
* Develop and implement a public relations plan to manage the reputation of the organization in case of any negative incidents.

Monitoring

* Monitor media and public feedback regularly to detect any negative news or comments regarding the organization.
* Conduct regular stakeholder surveys to identify any areas of concern or improvement.

Management

* Assign a dedicated public relations officer to manage the organization's reputation.
* Develop and implement a crisis management plan to address any negative incidents and minimize the impact on the organization's reputation.

|  |  |  |
| --- | --- | --- |
| S no. | Questions | Value |
| 1 | How frequently will the system need to be updated with new match schedules and team/player data? | High importance (4/5) |
| 2 | What level of access control and user permissions will be required for the system? | High importance (4/5) |
| 3 | What are the potential risks of cyber-attacks and data breaches for the system? | High importance (4/5) |
| 4 | How will the system handle data backup and disaster recovery in case of hardware or software failure? | Medium importance (3/5) |
| 5 | What are the hardware and software requirements for the system, and are they compatible with the existing IT infrastructure? | Medium importance (3/5) |
| 6 | What level of user training and support will be required for the system? | Medium importance (3/5) |
| 7 | How will the system integrate with other third-party applications or services, such as payment gateways or social media platforms? | Low importance (2/5) |
| 8 | What are the system's scalability and performance requirements to support future growth? | Low importance (2/5) |
|  | VALUE ADJUSTMENT FACTOR(VAF) ∑(fi)=25 |  |

Value Adjustment Factor:

Rate on each factor on 0-5:

1. No Influential
2. Incidental
3. Moderate
4. Average
5. Significant
6. Essential

Testing

Testing is an essential phase in the software development life cycle that ensures the quality and functionality of the system. It involves the evaluation of a software application against the specified requirements and intended behaviour. In the context of a cricket management system, testing is crucial to ensure that the system meets the needs of users and stakeholders.

There are two main types of testing: black box testing and white box testing. Black box testing involves evaluating the system without considering its internal workings, while white box testing involves examining the internal structure of the system. Both types of testing are important to ensure the effectiveness and efficiency of the system. By combining both black box and white box testing, developers can identify and resolve issues in the system, ensuring that it meets the requirements and expectations of users.

Basic path testing is a white box testing technique because it requires knowledge of the program's internal structure and control flow. It involves analysing the control flow graph of the program to identify and test all possible paths or combinations of paths within the program.

Designing Program graph: -

1. int main(intballs\_played, inttot\_runs)

2. {

3. float strike rate;

4. char rank;

5. strike rate (tot runs/balls played)\* 100;

6. cout<<"the strike rate of the player is:"<<strike\_rate;

7. if((strike\_rate=300 && strike\_rate=200)&&tot\_runs>3000)

8. {

9 . rank='H';

10. cout<<"the player has the highest rank"<<rank;

11. }

12. else if((strike\_rate=200 &&strike\_rate>-100)&&tot\_runs>2000)

13. {

14. rank='A';

15. cout<<"the player has an average rank"<<<rank;

16. }

17. else if((strike\_rate<=100 &&strike\_rate=50)&&tot\_runs>1000)

18. {

19 rank='B';

20. cout<<"the player has a below average rank”<<rank;

21. }

22. Else

23. {

24. rank='L';

25. cout<<”the player has least rank"<<rank;

26. }

2

Cyclomatic complexity is a software metric used to measure the complexity of a program by counting the number of independent paths through its source code. The metric provides developers with a quantitative measure of the program's complexity and helps them identify code that may be difficult to test, maintain, or modify. It is commonly used in software engineering to assess the quality and maintainability of code and to identify potential errors and bugs.

P= 3

E= 11

N= 9

R= 4

Where P= no. of predicate nodes

E = no. of edges

N = no. of nodes

N = no. of regions

Cyclomatic complexity = E-N+2 = 11-9+2 = 4

Or

Cyclomatic complexity = P+1 = 3+1 = 4

Or

Cyclomatic complexity = R = 4

* **To find number of independent paths**

No. of independent paths = cyclomatic complexity = 4

* Independent paths

1-6,7,8-11,27

1-6,7,12,13-16,27

1-6,7,12,17,18-21,27

1-6,7,12,17,22-26,27