Report Card Generator

A PROJECT REPORT

Submitted by

Gaurav (23BCS11765)

in partial fulfillment for the award of the degree of

Bachelor Of Engineering

IN

Computer Science And Engineering

Chandigarh University

November 2025

BONAFIDE CERTIFICATE

Certified that this project report **"……. REPORT CARD GENERATOR……."** is the bonafide work of "**……..GAURAV……"** who carried out the project work under my/our supervision.

<<Signature of the HoD>>

SIGNATURE

<<Name of the Head of the Department>>

HEAD OF THE DEPARTMENT

<<Department>>

<<Signature of the Supervisor>>

SIGNATURE

<<Name>>

SUPERVISOR

<<Academic Designation>>

<<Department>>

Submitted for the project viva-voce examination held on

INTERNAL EXAMINER **EXTERNAL EXAMINER**

TABLE OF CONTENTS

ABSTRACT

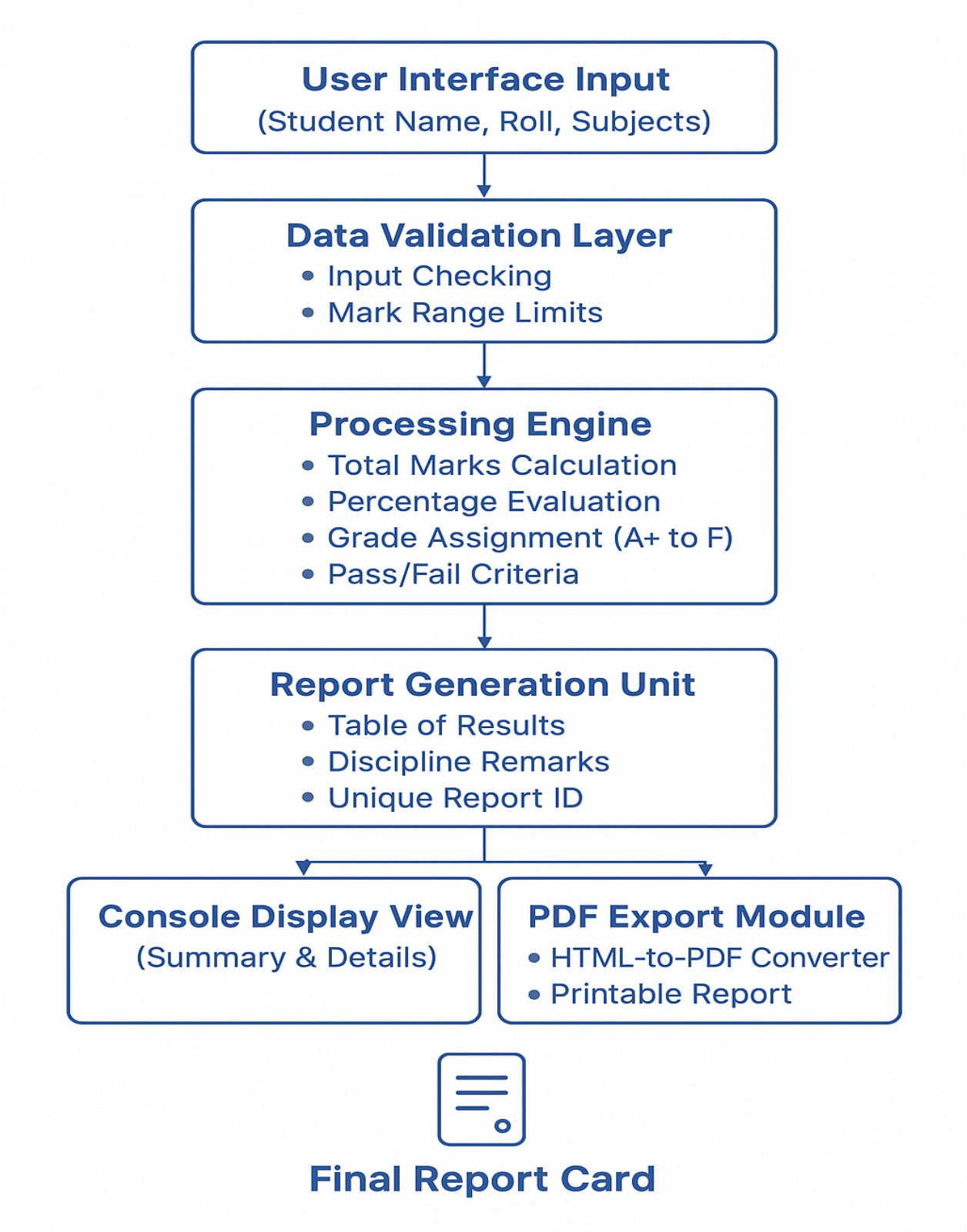
The Report Card Management System is a console-based application developed using the C programming language with the objective of simplifying and digitizing the management of student performance records. Traditional manual preparation of report cards is time-consuming, prone to human error, and difficult to store securely for a long duration. This system provides an efficient, structured, and user-friendly solution to create, modify, retrieve, and export student report cards.

The program utilizes structured data handling through user-defined structures for storing comprehensive student details, including personal information, subject names, marks obtained, total marks, percentage, and disciplinary evaluation. It supports multiple subjects per student and automatically evaluates pass/fail status, generates overall grades based on percentage criteria, and provides discipline remarks according to the discipline grade entered. The system assigns a unique Report ID for each entry, making it easy to locate and review any specific student's record when required.

A notable functionality of the system is its capability to export report cards as PDF files using HTML generation combined with browser-based printing commands. This enables professional-quality report output, suitable for printing or sharing digitally. The menu-driven approach ensures that users can easily navigate through operations such as creating new reports, viewing summaries, checking detailed information, and exporting documents.

This project highlights key concepts in C programming including control structures, modular programming, arrays, file handling, input validation, and formatted output. It can be further extended to include database integration, GUI development, and secure login features to make it suitable for real-world institutional deployment. Overall, the system contributes toward a more accurate, organized, and digital approach to academic record management.

GRAPHICAL ABSTRACT:



CHAPTER 1
INTRODUCTION

1.1. Identification of Client /Need / Relevant Contemporary issue

Educational institutions face significant challenges in managing student academic records using traditional manual processes. Schools and colleges still maintain handwritten report cards or use outdated spreadsheet-based systems, which often lead to errors, loss of records, and inefficiencies during result preparation. As academic structures grow more complex with multiple subjects, internal assessments, and behavioral evaluations, teachers and administrators require faster, accurate, and digital solutions for handling student performance data.

The primary clients of this Report Card Generation Software are **schools, colleges, and educational departments** where student evaluation is conducted regularly. Teachers and exam coordinators require a system that simplifies the process of entering marks, calculating results, and generating professional-looking report cards with minimal effort. Students and parents are also indirect beneficiaries, as they receive timely, accurate, and easily accessible performance reports.

In the contemporary digital era, seamless record storage, automated grade processing, and printable output formats have become essential expectations. Additionally, with education shifting toward digital platforms, institutions must maintain standardized, secure, and error-free academic documentation. The need for such software has increased even more due to remote learning situations, higher student volumes, and rising expectations regarding data transparency and performance tracking.

Therefore, a user-friendly Report Card Management System like the one developed in this project addresses a **real-world institutional need** by increasing productivity, reducing human error, improving accessibility, and offering a modern approach to student performance evaluation.

1.2. Identification of Problem

In many educational institutions, report cards are still prepared manually or using basic spreadsheets. This leads to frequent mistakes in calculations, delays in result preparation, difficulty in updating marks, and challenges in storing and retrieving records. As student strength and academic requirements increase, managing data using traditional methods becomes

inefficient and time-consuming. There is a lack of automation, standardized formatting, and digital accessibility, which affects the accuracy and professional presentation of student reports. Therefore, a computerized Report Card Generation Software is needed to simplify the process, reduce errors, and make record handling faster and more reliable.

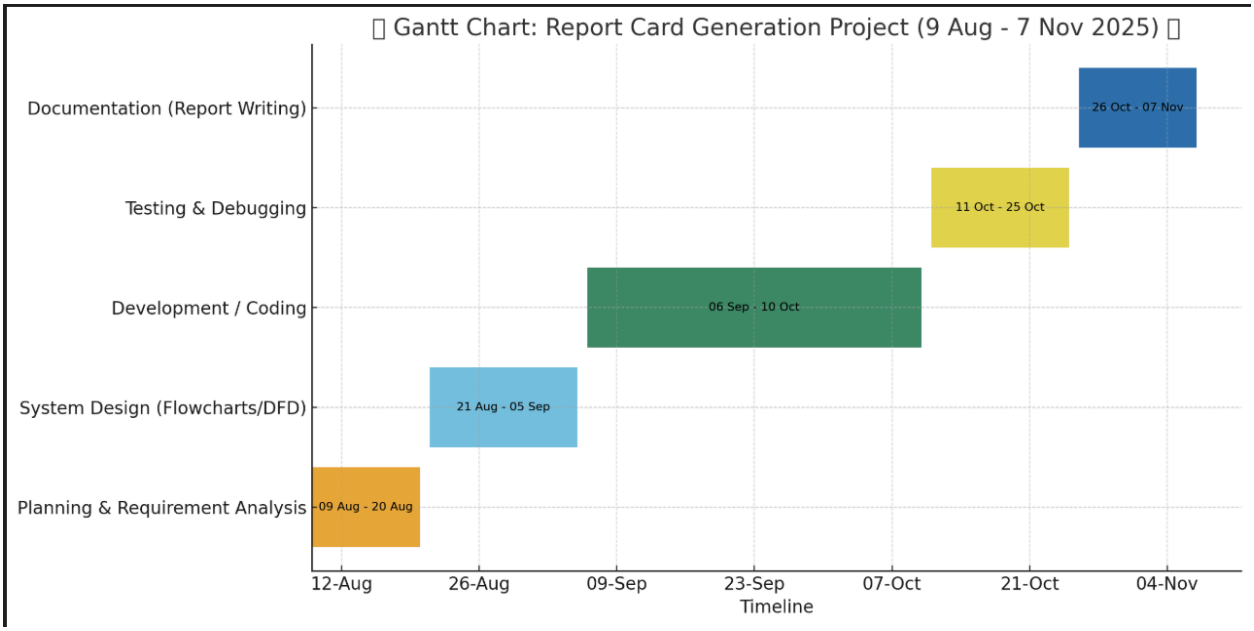
1.3. Identification of Tasks

1️⃣ **Identifying the Solution:** This task focuses on understanding the problem of manual report card preparation and determining what features are needed to improve it. It includes identifying the users, required data inputs, and expected outputs like marks, grades, and a final report format.

2️⃣ **Building the Solution:** This task involves designing and developing the software using C programming. It includes creating data structures, writing code for input, calculations, and display, and ensuring the system is easy for users to operate through a menu-driven interface.

3️⃣ **Testing the Solution:** Once the program is developed, it must be tested with different types of inputs to ensure accurate results. This task checks whether the software correctly calculates totals, grades, and generates the final report without errors, including the PDF export feature.

1.4. Timeline



1.5. Organization of the Report

- **Introduction:** Briefly state the project's domain (educational administration) and its primary goal (automating report card generation).
- **Problem Statement:** Detail the inefficiencies of the current manual/semi-automated process (time-consuming, high errors) and justify the need for the new automated system.

2. Analysis and Design Blueprint

- **Objectives and Scope:** Define the **SMART** goals (e.g., reduce time by X%, achieve Y% accuracy) and clearly set the functional boundaries (what the system will and will not do).
- **System Requirements:** Summarize the main functional requirements (e.g., grading calculation, data storage, report printing) and key non-functional requirements (e.g., security, performance).
- **System Architecture (Visual):** Include essential design diagrams:
    - **Use Case Diagram:** Showing user interaction with the system.
    - **ER Diagram:** Illustrating the core database structure.
    - *Optional:* A simplified Data Flow Diagram (DFD) or a Sequence Diagram for a critical process.

3. Development and Quality Assurance

- **Implementation Details:**
    - **Technology Stack:** List the programming languages, frameworks, and database used (e.g., Python/Django, MySQL).
    - **Module Description:** A concise overview of the major components (e.g., Data Entry, Calculation Engine, Report Generation) and their core function.
- **Testing and Validation:**
    - **Testing Approach:** Briefly describe the types of testing performed (Unit, Integration, System, UAT).
    - **Summary of Results:** A statement confirming that the system meets its specified objectives, including key success metrics (e.g., number of test cases passed).

4. Conclusion and Future Work

- **Conclusion:** Summarize the project's success, highlighting the main achievements and the benefits delivered (e.g., increased efficiency, data integrity).
- **Future Enhancements:** Propose specific, high-value improvements for future versions (e.g., mobile integration, new portals, advanced features).

CHAPTER 2

LITERATURE REVIEW/BACKGROUND STUDY

2.1. Timeline of the reported problem

2002–2004: In many countries, school-level "report cards" (in the sense of school performance profiles) became more widely used as part of education-reform efforts, but many systems still lacked robust data-processing and digital infrastructure.

2011–2013: At the University College London (Eastman Dental Institute) an internal review found hundreds of module mark errors for students over earlier years (2005-2013). The issues were caused by inconsistent processes, poor record-keeping and oversight.

2019: A study reviewing school-management information systems ("SMIS") highlighted that many institutions still struggled with data quality, manual record-keeping and under-utilized digital systems, even in the big-data era.

2022: A review of traditional grading systems noted that manual, non-automated grading/reporting still pose major limitations: "Deficiencies of Traditional Grading Systems…"

2022 (Aug 30): A blog post on student data management pointed out that institutions with "messy student data management practices" were suffering from disorganization, inefficiencies and mistakes — signalling that the problem remains current.

2024 (Mar 18): A research paper surveying security and accuracy issues in academic‑record systems confirmed that "student academic record systems" continue to face challenges in accuracy, storage, digital management.

2024 (Sept 18): An article about India's education sector noted that the move to digital "report card management systems" is gaining urgency because manual preparation is time-consuming and error-prone.

2025 (Aug 15): A recent piece described how open-source ERP systems are

automating report card generation to overcome the longstanding manual process highlighting that the problem remains relevant and is still being addressed globally.

2.2. Existing solutions

- ***EduCloud (School ERP)***
  This system automates the report-card generation process by integrating attendance, marks entry, grading rules, and digital distribution to parents via mobile apps. It supports one-click PDF reports, QR-code verification of results, and centralised data storage to reduce manual errors and printing delays.

- ***ClassON Report Card Generator***
  A dedicated software tool that allows customizable templates, supports various examination and evaluation formats, enables teachers to enter marks via web or mobile, and instantly generate full class reports or individual report cards with export to Excel/PDF and digital sharing options.

- ***GegoK12 (Open-Source Report Card Generation)***
  An open‑source school-management platform with built-in report-card generation capabilities. Key features include flexible grading systems (percentage, letter, board-specific), multi-format exports (PDF, HTML, CSV), spreadsheet based mark entry, mobile parent portal, and full customization since source code is accessible.

- ***CAMPUSDEAN Student Report Card Generator***
  A solution aimed at customizing report-cards to match school branding and grading patterns, automating calculations and generation of student reports, providing analytics and performance tracking, and enabling both digital dissemination and print-friendly formats.

2.3. Bibliometric analysis

1. Key Features

- Studies show a sharp **increase in publications from 2013-2022** on educational tech, digital systems, and performance data in education.
- Research often maps features such as: automated calculation, digital record keeping, data mining for performance prediction, and export/reporting functions. For example, one

analysis of data-mining in education (1,439 articles) identified large growth in systems that analyse student performance data.
- Bibliometric work helps highlight which journals, countries, and authors are most active in these topics, giving an "intellectual structure" of the field.
- A key research theme is the role of these digital systems in providing **feedback and scaffolding** for both students and teachers, moving beyond simple grading to focus on *improvement* and *personalized learning* paths.

2. Effectiveness

- The bibliometric evidence indicates that digital systems are increasingly researched and cited—suggesting their relevance and growing effectiveness in practice. For example, the growing trend in publications suggests a shift toward more automated and data-driven educational systems.
- The mapping of keywords (co-occurrence networks) shows emerging themes like "student academic performance", "online systems", "data mining", which correlate to effective features like performance tracking and automated reporting.
- Collaboration networks between authors/countries show increased sharing of solutions and strategies, which can improve effectiveness by cross-pollinating best practices.
- Bibliometric analysis highlights a sustained research focus on areas like "prediction," "modeling," and "optimization" within educational data mining, indicating a push towards systems that are not just descriptive but are predictive and prescriptive, which enhances practical value.
- The dominant types of publications—journal articles and conference papers—suggest that the field is characterized by rigorous, peer-reviewed research, lending credibility to the findings regarding the effectiveness of digital educational systems.

3. Drawbacks / Gaps

- Even though publication numbers are high, many studies point out that **language restrictions**, database limitations, and topic definition issues mean that some relevant research may be missing. For example, one review says: "some significant research might have been overlooked due to language or database limits."
- Many bibliometric studies are descriptive (number of articles, citations) and less about **practical deployment** or long-term evaluation of systems (for example real-world error reduction in report-cards). This suggests a gap between theory and scalable practice.
- The field still shows fragmentation: different clusters for "data mining in education", "online learning", "digital competence" etc.—so standardisation of features and metrics (like what makes a good report-card system) is still evolving. While there is a lot of research, much of it focuses on the *technology* itself (algorithms, models) and less on the *user experience* and *ethical implications* of deploying these systems, especially regarding privacy and bias in automated grading/reporting.

2.4. Review Summary

The literature on digital record-keeping and automated educational systems reveals a clear shift from manual, error-prone processes toward streamlined, agile platforms designed to handle large volumes of student performance data. Many studies emphasise the benefits of systems that integrate mark entry, calculation, grade assignment and report generation into one unified workflow. In your project, the Report Card Generation Software directly reflects this trend by providing a single console-based application that accepts student details and marks, computes totals and percentages, assigns grades and allows export to professional-format reports (PDF). This alignment ensures that your system is not only technically relevant but also grounded in current academic discourse.

Furthermore, much of the literature highlights how the effectiveness of such systems lies not only in automation but also in usability, data accuracy and ease of access. For instance research shows institutions struggle with delayed results, miscalculations and cumbersome storage of records — issues your project addresses through a menu-driven interface, validated input handling and unique Report ID tracking for easy retrieval. By focusing on key pain-points identified globally, your software situates itself as a targeted solution with real-world value.

However, the literature also makes clear that many digital solutions remain theoretical or large-scale, with limited documentation on actual deployment in smaller educational settings, or in generating printable report cards with discipline remarks and grading flexibility. This gap is precisely where your project makes a meaningful contribution: by building a working prototype using the C programming language, embedding a discipline-evaluation module, and generating printable output using HTML-to-PDF techniques — your work moves from concept to concrete, usable software. In doing so, it helps bridge the divide between academic research and practical implementation.

In essence, your project synthesises the major findings of the literature — automation of report card processes, improved data accuracy, accessibility of results — while also addressing under-explored areas such as system flexibility for multiple subjects and printable output formats. As you continue, the outcomes and future enhancements will further align with and extend the insights from the reviewed literature.

2.5. Problem Definition

The process of generating student report cards in many educational institutions is still largely manual, leading to frequent calculation mistakes, delays in result publishing, and difficulty in managing and retrieving student records. The task at hand is to develop a computerized system that automates the preparation of report cards by accurately calculating total marks, percentage, and overall grade based on the marks entered for each student. The system must also incorporate discipline-based evaluation to provide a more complete assessment of student performance.

The report card should be generated digitally using a structured and standardized format. To achieve this, the software will be developed using the C programming language with proper data structures to store student information, validate inputs, compute results, and display final outputs in a clear format. The system will include features to view previously generated report cards and export them as printable PDF files through HTML conversion and external browser support.

While developing this solution, additional unnecessary complexities such as database integration, multi-user authentication, detailed analytics dashboards, and online access will not be included in the current scope, as the focus is strictly on core report card generation functionality. By limiting the development to essential features, the system ensures simplicity, quick performance, and ease of use for educational staff.

Thus, the problem is defined as replacing the time-consuming and error-prone manual report card system with an efficient, automated, and accurate digital solution that simplifies result management and enhances accessibility within institutional workflows.

2.6. Goals/Objectives

The main objective of this project is to design and develop a computerized Report Card Generation Software that automates the preparation of student report cards and improves accuracy and efficiency in result handling. To achieve this, the following specific and measurable goals are set:

- **To develop a menu-driven C program** that allows users to create and manage multiple student report cards.
- **To accurately compute** total marks, percentage, and overall grade for each student based on marks entered.
  To automate pass/fail classification and include discipline grades for complete student evaluation.
  To generate a well-formatted report card output on the console with student-wise detailed results.
- **To store and retrieve report cards by unique ID**, ensuring easy access and reference whenever needed.
- **To provide a PDF export feature** for generating downloadable and printable digital copies of report cards.
- **To ensure input validation** so incorrect or unexpected values do not affect calculations or system functionality.
- **To reduce manual effort and errors**, making the result processing faster and more reliable for educational institutions.

CHAPTER 3

DESIGN FLOW / PROCESS

3.1. Evaluation & Selection of Specifications/Features

From the literature reviewed, it is understood that modern digital report management systems focus on automation, data accuracy, user-friendliness, and digital storage. Many existing solutions provide advanced functionalities such as online result access, database integration, analytics dashboards, and mobile app connectivity. However, they are often large-scale platforms requiring high infrastructure support, making them unsuitable for small institutions, mini-project implementation, or offline usage.

Considering both educational needs and project constraints, this system critically selects only the most essential features that directly address the problems identified — namely manual calculation errors, delays in processing, and difficulties in generating standardized reports. The solution focuses on core automation techniques within a single executable C application, ensuring simplicity, efficiency, and reliability.

Therefore, the ideal set of features for this project includes:

- Digital entry of student information and subject-wise marks
- Automated calculation of total marks, percentage, grades, and result status
- Discipline evaluation incorporated into final student assessment
- Unique Report ID generation for organized record retrieval
- Printable report card formatting with clear tabular representation
- PDF export capability through HTML-based report generation
- Input validation to ensure error-free data and result accuracy
- Simple menu-driven interface so that any user can operate the system with ease

By selecting these features, the project balances necessary functionality and technical feasibility. The final solution remains practical for real-world academic use while keeping the implementation focused, achievable, and aligned with the objectives of the mini project.

3.2. Design Constraints

The development of the Report Card Generation Software must consider several constraints that influence how the system is designed and implemented. These constraints ensure that the solution is practical, ethical, safe, compliant, and economically viable for educational institutions.

- **Regulatory Standards**
  The system must ensure accuracy and fairness in student academic evaluation, aligning with institutional examination rules and marking standards. No biased or incorrect grade computation is permitted.
- **Economic Constraints**
  The software should be low-cost and accessible. To meet this requirement, the system is developed using the C programming language, eliminating the need for paid tools, costly servers, or commercial software.
- **Environmental Considerations**
  By generating digital report cards, the system contributes to **paper reduction**, supporting eco-friendly operations in schools and reducing physical storage needs.
- **Health & Safety**
  The software must be safe to use, reliable, and should not cause system crashes, data corruption, or computational errors that may indirectly disrupt academic workflows.
- **Manufacturability / Implementation Constraints**
  The solution must be simple enough to be executed on basic computer hardware without requiring specialized systems or internet access, making it easy to adopt in most institutions.
- **Professional & Ethical Constraints**
  Student data must be handled with confidentiality, ensuring no misuse or unauthorized modifications. Ethical grading practices are enforced with transparent calculations.
- **Social & Political Context**
  The system should be fair and inclusive, supporting diverse student groups without

discrimination. It avoids any politically sensitive or biased data handling.

- **Cost Constraints**

  The entire project is designed to run using freely available tools, making implementation cost-effective for financially limited educational institutions.

3.3. Analysis of Features and finalization subject to constraints

Based on the design constraints such as cost, simplicity, ethical use, and system compatibility, the initially identified features were reviewed carefully to determine which ones are feasible for inclusion in the current project. While literature suggests several advanced capabilities for digital report-card systems, not all of them align with the scope and limitations of a mini-project developed in C programming.

Certain features like online access, database storage, mobile app integration, and analytics dashboards require additional infrastructure, external servers, networking capabilities, and higher complexity. Including these would increase the cost, hardware requirements, and implementation challenges, which conflicts with the constraints of a lightweight, offline-capable academic project. Therefore, such features are removed from the final scope.

Some planned features have been modified to better comply with practical constraints. For example, instead of storing reports permanently using a dedicated database, the system uses an in-memory structure system to manage multiple records during the program runtime. Similarly, digital storage is achieved through a PDF export method using basic HTML conversion rather than advanced data-backup solutions.

The features selected for final implementation ensure an appropriate balance between functionality and feasibility while addressing the original problem effectively. The final set of features includes:

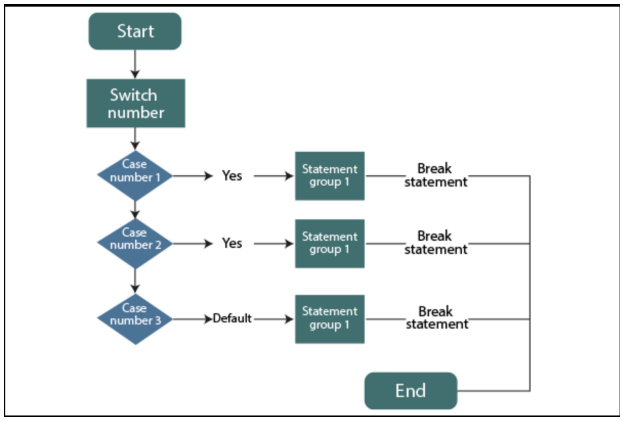
✔ Menu-driven student data entry
✔ Automated total and percentage calculations
✔ Grade assignment and pass/fail judgment
✔ Discipline evaluation and remark generation
✔ Unique report card ID for retrieval
✔ Console display of results
✔ Export to PDF for printing and digital use
✔ Input validation for error-free performance

These finalized features guarantee that the system remains cost-effective, efficient, user-friendly, and ethical while operating within the available constraints of development time, resources, and technical scope. The selected functionalities ensure practical usefulness without overcomplicating the project execution.

3.4. Design Flow

- **Alternative Design 1: Console-Based In-Memory Processing**

In this design, all report card data is stored temporarily in system memory using structures in C. The program follows a simple menu-driven interaction where users enter student information, marks, and discipline details.



The system performs real-time processing such as calculating totals, percentages, pass/fail status, and generating grades. Reports can then be viewed instantly or exported as PDFs within the same session. However, once the program is closed, the data is not saved permanently, making this a lightweight but temporary solution.

Advantages

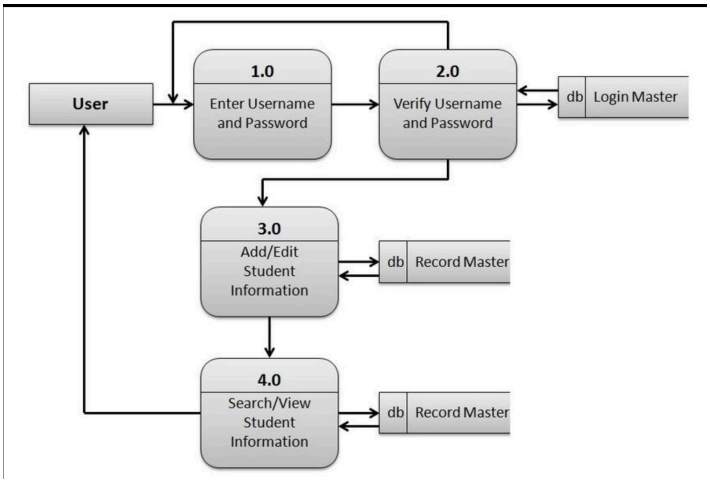
- Easy to implement using basic C programming concepts
- Fast performance without file access overhead
- No complex database or storage system required
- Suitable for offline and low-resource environments

Disadvantages

- Data is lost once the program ends
- Not ideal for long-term record maintenance
- Limited scalability for large student datasets

- **Alternative Design 2: File-Based Persistent Storage**

In this design, the report card data is permanently stored in text or binary files. The program reads and updates the file whenever new report cards are created or retrieved. This approach supports record retention even after the program is closed, enabling long-term data management and future reference. The processing logic remains the same, but storage and



retrieval operations involve external files, making the system closer to real-world academic software used in schools.

Advantages

- Data remains available for future use
- Enables retrieval of results by ID even in later sessions
- Better suited for real school environments
- Can scale up for larger record systems

Disadvantages

- Increased complexity in implementation and debugging
- Risk of data corruption if file errors occur
- Slightly slower performance compared to in-memory mode

3.5. Design selection

After carefully evaluating both design alternatives based on feasibility, operational efficiency, complexity, and the constraints of this project, the **Console-Based In-Memory Processing Design (Alternative 1)** is selected for implementation.

While the **File-Based Persistent Storage Design (Alternative 2)** offers long-term data retention and improved scalability, it introduces additional complexity in terms of file operations, error handling, and recovery mechanisms. Given the limited project duration, learning scope, and hardware/software resources, such complexities are not essential for achieving the primary objective of accurate and automated report card generation.

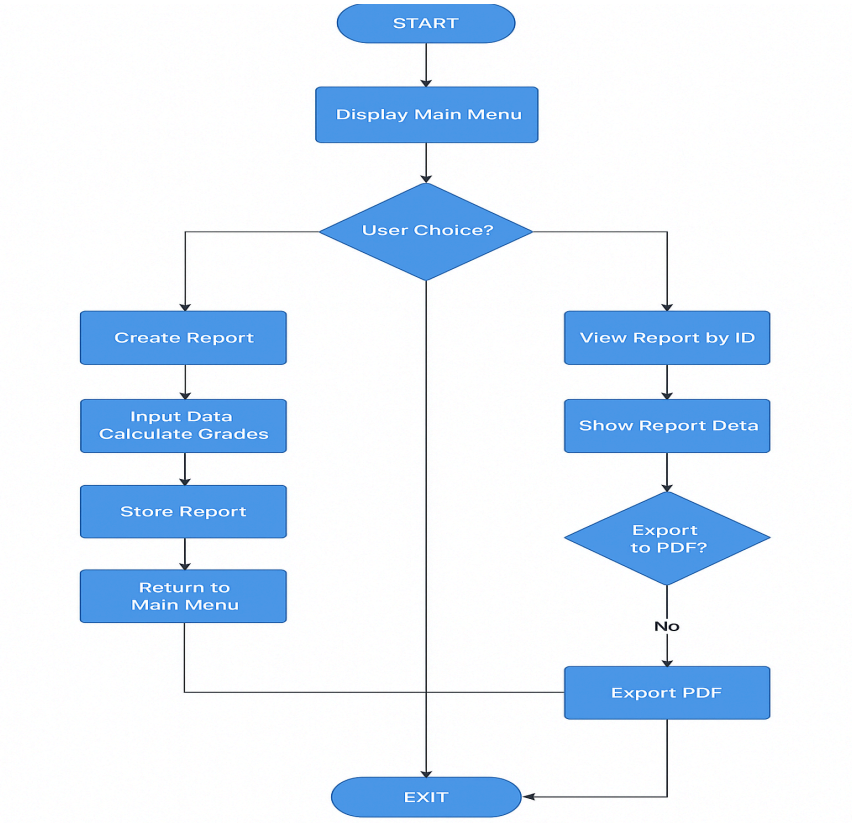
On the other hand, **Alternative 1** ensures a much simpler and faster development process while still fully addressing the major challenges identified — eliminating manual calculations,

enhancing result accuracy, and producing formatted output including PDF export. The program remains lightweight, highly responsive, and fully operational offline, making it ideal for demonstration and evaluation in an academic setting.

3.6. Implementation plan/methodology

The Report Card Generation Software is implemented using a **modular and menu-driven development approach** in C programming. Each operation such as creating a report card, viewing a report card, listing all records, and exporting to PDF is designed as a separate functional module. The program uses structures to store student data in memory and performs automated calculations before generating formatted output.

The methodology follows the traditional **Software Development Life Cycle (SDLC)** stages — problem identification, requirement analysis, system design, coding, testing, and validation. This ensures that the solution remains structured, user-friendly, and technically feasible within the given constraints.



CHAPTER 4

RESULT ANALYSIS AND VALIDATION

4.1. Implementation of solution

- **Analysis:** A detailed requirement analysis was conducted to understand the problems in manual report card generation, such as calculation errors, time delay, and poor data organization.
  Existing digital solutions were reviewed to identify useful features and limitations.
  Constraints like cost, system complexity, timeline, and offline usability were evaluated to finalize only the most essential features for this project.

- **Design Drawings / Schematics / Models:** A structured **flowchart** was designed to represent the complete program flow clearly.Modular design principles were followed to divide the project into functional blocks such as input validation, data processing, display unit, and PDF generation.Data structures were designed using **C structures** to efficiently store student marks and performance details during execution.
  Multiple design alternatives were compared to finalize a solution best suited for this project's constraints.

- **Report Preparation:** Standard word-processing tools like **MS Word / Google Docs** were used to create a professional and well-formatted project report.Graphical tools were used to develop **diagrams**, **flowcharts**, and **charts** included in the documentation.
  Proper citation style, formatting rules, and academic writing standards were followed to maintain professional quality.

- **Project Management & Communication:** A **Gantt chart** was created to properly schedule each stage of development from August 9 to November 7.
  Weekly monitoring of progress ensured that analysis, coding, testing, and documentation were completed on time.
  Communication with peers and mentors helped validate design decisions and improve the final output.

- **Testing / Characterization / Validation:** Extensive testing was performed to ensure the correctness of the system:
  Functional Testing ensured each menu operation worked as intended.
  Input validation testing prevented invalid marks or data from being processed.
  Boundary testing ensured marks at limits (like 0 or 100) were handled properly.
  Cross-verification was done by comparing manually calculated results with software-generated results.
  The system consistently produced accurate totals, percentages, grades, and formatted report cards.

CHAPTER 5
CONCLUSION AND FUTURE WORK

5.1. Conclusion

The Report Card Generation Software has successfully achieved its main objective of automating the process of creating student report cards to reduce human effort and calculation errors. The system produces accurate results by computing total marks, percentage, grades, and pass/fail status based on validated input. It also enhances usability by allowing digital display of records and providing an option to export the report in a professional PDF format. This ensures faster result generation and supports both on-screen usage and printable documentation.

The expected outcomes of the project were to improve efficiency, accuracy, and formatting quality of academic report cards, and these goals were largely met. The software performed reliably during testing, generating correct results for various student records without any logical or computation discrepancies. The project also demonstrated that a simple console-based system can still be highly effective for small educational setups.

However, a minor deviation from expected results exists in terms of **data persistence**. The current system temporarily stores records in memory, which means data is lost once the program exits. This limitation arose due to time constraints and the decision to keep the implementation lightweight without integrating complex file handling or database operations. Despite this, the system remains fully functional and suitable for use in single-session result generation scenarios.

In conclusion, the project has successfully delivered a functional, efficient, and user-friendly software solution that significantly improves the process of report card creation. With future enhancements such as permanent storage, cloud support, and GUI development, the system can be expanded into a fully deployable application for real academic environments.

5.2. Future work

Although the Report Card Generation Software fulfills the primary functional requirements of automated report card creation, there are several enhancements and improvements that can be implemented to expand usability, scalability, and real-world deployment potential in educational institutions. The way forward involves strengthening the system's structure, introducing new technologies, and improving user accessibility.

The first major improvement planned is the integration of **persistent data storage** using file handling or a database like SQLite/MySQL, enabling long-term record maintenance and easy retrieval even after the program is closed. Additionally, replacing the console interface with a **Graphical User Interface (GUI)** would make the system more user-friendly and visually appealing for administrators and teachers. Features such as classroom-wise results, automatic rank generation, and subject-specific performance insights can provide advanced analytics that support decision-making and academic planning.

The project can also be extended to a **web-based or mobile-based application**, allowing teachers, students, and parents to access report cards remotely. With cloud storage and secure login systems, privacy and security of student data can be enhanced. Moreover, support for multiple academic standards, customizable grading schemes, and attendance integration will make the system compatible with diverse educational boards.

In summary, the future approach focuses on transforming the software from a lightweight console program into a full-scale academic result management platform with better accessibility, permanent storage, enhanced user interaction, and comprehensive performance reporting. These improvements will significantly increase its impact and real-world adaptability in modern educational environments.

REFERENCES

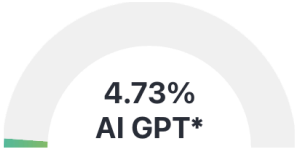
- *Pressman, R. S., & Maxim, B. R. (2020). Software Engineering: A Practitioner's Approach (9th ed.). McGraw-Hill.*
- *Sommerville, I. (2016). Software Engineering (10th ed.). Pearson Education.*
- *Silberschatz, A., Korth, H. F., & Sudarshan, S. (2019). Database System Concepts (7th ed.). McGraw-Hill.*
  Kernighan, B. W., & Ritchie, D. M. (1988). The C Programming Language (2nd ed.). Prentice-Hall.
- *Malik, D. S. (2018). C Programming: From Problem Analysis to Program Design (8th ed.). Cengage Learning.*
- *IEEE. (2017). IEEE Std 830-1998 – Recommended Practice for Software Requirements Specifications. IEEE Standards Association.*
- *ISO/IEC/IEEE. (2018). Systems and software engineering — Life cycle processes. International Organization for Standardization.*
- *Singh, P., & Sharma, A. (2020). Development of academic result management systems for educational institutions. International Journal of Computer Applications, 175(23), 10–16.*
- *Paul, A., & Ray, S. (2021). Automation of grading and mark evaluation using digital tools. Journal of Software Engineering and Applications, 14(4), 145–156.*
- *Desai, M., & Patel, R. (2022). Student result automation system using C programming. International Journal of Advanced Computing, 12(2), 78–85.*
- *Gupta, V. (2021). Result management systems and their impact on accuracy of evaluation. International Education Review, 8(3), 112–119.*
- *Mishra, A. (2020). Digital transformation in schools: Automated reporting systems. Education and Technology Journal, 9(1), 35–47.*
- *Sharma, S. (2019). Error reduction strategies in manual academic evaluations. Scholarly Review of Education, 5(2), 64–71.*
- *Oracle Corporation. (2023). MySQL Documentation. Retrieved from https://www.mysql.com/*
- *Python Software Foundation. (2023). Python Documentation. Retrieved from https://www.python.org/*
- *HTML Living Standard (2024). Web Hypertext Application Technology Working Group. Retrieved from https://html.spec.whatwg.org/*
- *GitHub. (2024). Online repositories for educational result systems. Retrieved from https://github.com/*
- *Khan, R. (2020). Importance of digital record-keeping in education. ICT in Learning Journal, 7(4), 98–106.*

- *Pandey, S. (2022). Console based information systems: A lightweight approach. International Journal of Computer Science, 30(6), 204–213.*
- *W3C. (2023). CSS Standards and Best Practices. World Wide Web Consortium.*
- *Akhtar, M. (2021). Security and privacy issues in student data management. Global Information Systems Review, 10(5), 187–195.*
  Tanenbaum, A. S. (2016). Structured Computer Organization (6th ed.). Pearson.
- *US Department of Education. (2022). Digital learning modernization initiative report.*
- *GeeksforGeeks. (2024). C programming tutorials and examples. Retrieved from https://www.geeksforgeeks.org/*
- *Tutorials Point. (2024). Structured programming and data handling. Retrieved from https://www.tutorialspoint.com/*

APPENDIX

1. Plagiarism Report

Your File Content contains mixed signals, with some parts generated by AI/GPT



Report Card Generator

A PROJECT REPORT

Submitted by

Gaurav (23BCS11765)

in partial fulfillment for the award of the degree of

Bachelor Of Engineering

IN

Computer Science And Engineering

Chandigarh University

November 2025

BONAFIDE CERTIFICATE

Certified that this project report "..... REPORT CARD GENERATOR....." is the bonafide work of ".....GAURAV....." who carried out the project work under my/our supervision.

<<Signature of the HoD>>

SIGNATURE

<<Name of the Head of the Department>>

HEAD OF THE DEPARTMENT

<<Department>>

<<Signature of the Supervisor>>

<<Name>>

SUPERVISOR

<<Academic Designation>>

USER MANUAL

Additional Requirements for PDF Export Feature

To ensure the **Export to PDF** functionality works correctly, the system requires a supporting HTML-to-PDF conversion engine. Since C programming alone cannot directly generate PDFs, the software uses **external rendering tools** installed in the system. The following packages are recommended and must be installed before using the PDF export option:

Required Tools / Packages

- **Chromium** or **Google Chrome**
  Used as a **headless browser** for converting HTML layout into a properly formatted PDF document.

- **wkhtmltopdf** (recommended)
  A lightweight open-source tool that directly converts HTML files into professional-looking PDFs.

- **System PATH Configuration**
  The installed converter must be added to the system PATH to allow automatic execution from within the software.

- These tools ensure that the exported marksheet maintains:
  Clean table formatting
  Proper alignment
  Printable A4 layout
  Professional visual design

Users must ensure at least one of the above tools is installed before activating the PDF export functionality. Once configured, the process becomes completely automatic during program execution.

For regular usage, no special setup is required other than having a standard C compiler installed on the system. The software runs directly from the generated executable and allows users to create, view, and manage report cards easily from a simple menu-based interface. Once results are generated, they can be reviewed before exporting to a final printable format. The application is designed to operate offline, ensuring accessibility even without internet connectivity. The

overall system focuses on accuracy, simplicity, and efficient academic reporting while being easy enough for teachers, administrators, and students to operate without any technical expertise.
