

Design and Evaluation of a GUI-Based GPA Tracker System Using Python and Sqlite for Student Academic Record Management

Capati, Jazmine Claire M., Cea, Mark C., Esquilona, John Wilson M., Maligro, Christopher Dhanial P.
Maranan, Mark Daniel F., Marinay, Gabriel Andrei P., Engr. Fabro, Meshelle N.

Department of Computer Engineering Eulogio “Amang” Rodriguez Institute of Science and Technology
(EARIST) Nagtahan, Manila, Philippines

DOI: <https://doi.org/10.51583/IJLTEMAS.2026.150600048>

Received: 08 June 2026; Accepted: 13 June 2026; Published: 04 July 2026

ABSTRACT

This study focused on the development of a GUI-based GPA Tracker System designed to manage, organize, and compute student academic records efficiently. Manual recording of student grades often results in errors, misplaced records, and time-consuming computations, especially when handling multiple students. To address these concerns, the proposed system was developed using Python, Tkinter/CustomTkinter for the graphical user interface, and SQLite3 for database management. The system allows users to add, edit, search, delete, and view student records while automatically computing the Grade Point Average (GPA) based on encoded grades and subject units. Input validation was also integrated to prevent incomplete entries, invalid grades, duplicate student IDs, and other common recording errors. The results of the system testing showed that the GPA Tracker successfully performed its intended functions, including accurate GPA computation, record storage, data retrieval, report generation, and error handling. Overall, the system provides a more organized, accurate, and user-friendly approach to managing student academic records compared to manual methods. This project may also serve as a practical reference for future developers and students studying GUI design, database integration, and academic record management systems.

Keywords: GPA Tracker, Python, SQLite3, Graphical User Interface, Student Records, Academic Record Management

INTRODUCTION

Background of the Study

Managing student academic records is an important task in every educational institution. Teachers and instructors are responsible for recording, organizing, updating, and computing students' grades accurately. However, when records are handled manually through paper-based documents or separate spreadsheets, the process can become timeconsuming and prone to errors. Manual recording may lead to problems such as misplaced records, duplicate entries, incorrect grade computation, and difficulty in retrieving student information.

In academic settings, accurate grade management is essential because it helps monitor student performance and provides reliable information for academic evaluation. One important part of academic record management is the computation of the Grade Point Average, or GPA, which represents a student's overall academic performance. When GPA is computed manually, there is a higher possibility of miscalculation, especially when several subjects, grades, and units are involved. Because of this, there is a need for a system that can make the process faster, more accurate, and more organized.

To address this problem, the researchers developed a GUI-based GPA Tracker System using Python,

Tkinter/CustomTkinter, and SQLite3. The system is designed to help users manage student academic records through a simple and user-friendly graphical interface. It allows users to add, edit, search, delete, and view student records in one system. It also automatically computes GPA based on the encoded grades and subject units, reducing the need for manual computation.

The use of Python makes the system easier to develop and maintain because of its simplicity and readability. Tkinter/CustomTkinter provides the graphical user interface, allowing users to interact with the system through buttons, windows, and input fields. SQLite3 serves as the database for storing and retrieving student records efficiently. Through these tools, the GPA Tracker System provides a practical solution for improving the accuracy, speed, and organization of student record management.

Overall, this study aims to develop a functional GPA Tracker System that can assist users in managing academic records more efficiently. By reducing manual work and minimizing errors, the system can improve the process of recording grades, computing GPA, and monitoring student academic performance.

Statement of the Problem

This study aims to develop a GUI-based GPA Tracker System that can efficiently manage, organize, and compute student academic records.

Specifically, this study seeks to answer the following questions:

1. How can the system handle the process of recording students' grades?
2. How effective is the system in storing and managing student records?
3. How can the system reduce errors commonly encountered in manual grade recording and GPA computation?
4. How can a graphical user interface improve user interaction and usability of the system?

OBJECTIVES OF THE STUDY

General Objective

The general objective of this study is to develop a GUI-based GPA Tracker System using Python and SQLite3 database management.

Specific Objectives

Specifically, this study aims to:

1. Develop a system that can efficiently record and manage student academic records.
2. Determine the effectiveness of the system in storing, updating, searching, and retrieving student information.
3. Reduce errors commonly encountered in manual recording and GPA computation.
4. Improve user interaction and system usability through the implementation of a graphical user interface.
5. Provide an organized and functional system that can assist users in monitoring student academic performance.

Significance of the Study

This study is significant because it provides a computerized solution for managing student academic records. The GPA Tracker System can help improve accuracy, efficiency, and organization in recording and computing students' grades.

Users

The system will benefit users, especially teachers and instructors, by making the process of recording and managing student grades faster and more organized. Through its user-friendly graphical interface, users can easily add, edit, search, delete, and view student records without relying on complicated spreadsheets or manual records.

Organizations

Educational institutions, departments, or academic organizations may benefit from this study because the system can help reduce repetitive manual work and minimize human errors in handling student information. It can support a more systematic approach to academic record management and improve the overall grading workflow.

Students

Students may benefit from the system because accurate grade recording and GPA computation can help ensure that their academic performance is properly monitored. The system may also serve as a learning reference for students who want to understand GUI design, database integration, and software development.

Future Researchers and Developers

This study may serve as a reference for future researchers and developers who want to create similar systems related to academic record management, GPA computation, GUI-based applications, and database-connected software. It can also help future programmers improve their skills in Python programming, database management, and system development.

Scope and Limitations

This study focuses on the development of a GUI-based GPA Tracker System for Bachelor of Science in Computer Engineering students. The system is designed to manage student academic records using Python, Tkinter/Custom Tkinter, and SQLite3. It covers basic record management functions such as adding, viewing, updating, searching, deleting, and generating summaries of student records. The system also includes automatic GPA computation based on predefined subjects, grades, and corresponding units.

The system is intended for offline use only. It does not support online access, cloud storage, or real-time data sharing. It also does not include advanced features such as student accounts, teacher authentication, automatic backup, multiuser access, or integration with official school enrollment systems. The GPA computation is limited to the subjects and units encoded by the developers. Furthermore, the project was developed mainly for academic and learning purposes. It serves as a foundational system that demonstrates the application of Python programming, GUI development, and database integration. It is not intended to function as a full-scale commercial software for school-wide implementation.

REVIEW OF RELEVANT THEORY, STUDIES, AND LITERATURE

This chapter presents the relevant theories, studies, and literature that support the development of the GUI-based GPA Tracker System. It discusses concepts related to technology acceptance, information system success, graphical user interface design, Python programming, SQLite database management, and academic record management. These discussions provide the foundation for understanding the importance of developing a computerized system for recording, managing, and computing student academic records.

Relevant Theories

Technology Acceptance Model

This study is anchored on the Technology Acceptance Model, which explains how users accept and use a particular technology. According to Davis (1989), two major factors influence technology acceptance: perceived usefulness and perceived ease of use. Perceived usefulness refers to the degree to which users believe that a system can improve their performance, while perceived ease of use refers to the degree to which users believe that the system is easy to operate.

The Technology Acceptance Model is relevant to the GPA Tracker System because the system is designed to help users manage student academic records faster, easier, and more accurately. If teachers, instructors, or users find the system useful in recording grades and computing GPA, they are more likely to accept and use it. Likewise, if the graphical user interface is simple and easy to understand, users can operate the system with less difficulty.

In this study, perceived usefulness is reflected in the system's ability to add, edit, search, delete, and summarize student records. It is also shown through the automatic computation of GPA, which reduces manual work and possible calculation errors. Perceived ease of use is reflected in the GUI design, where buttons, input fields, and menus allow users to navigate the system more conveniently.

Information Systems Success Model

The study is also supported by the Information Systems Success Model by DeLone and McLean. This model explains that the success of an information system can be evaluated through factors such as system quality, information quality, service quality, system use, user satisfaction, and net benefits. In system development, these factors help determine whether a system is effective, reliable, and useful to its users.

The Information Systems Success Model is applicable to the GPA Tracker System because the system aims to provide accurate, organized, and accessible academic records. System quality is shown through the proper functioning of the system's features, such as adding records, editing data, searching information, deleting entries, and computing GPA.

Information quality is reflected in the accuracy and completeness of the stored student records. User satisfaction is achieved when the system becomes easier and more convenient to use compared to manual recording methods.

Through this theory, the GPA Tracker System can be evaluated based on its functionality, accuracy, usability, and overall benefit to users. The system's success depends not only on whether it works, but also on whether it improves the process of managing student academic records.

Review of Related Studies

GUI-based applications have become useful tools in improving user interaction with computerized systems. Buditjahjanto (2022) emphasized that graphical user interface simulations and interactive applications can improve user engagement and learning outcomes by providing organized and visual interaction. This is relevant to the GPA Tracker System because the use of a graphical interface allows users to interact with the system through buttons, menus, and windows instead of relying only on text-based commands.

In academic record management, ease of use is important because users need to perform tasks such as encoding, updating, searching, and viewing records quickly. A well-designed interface can reduce confusion and improve task completion. In the GPA Tracker System, the GUI helps users access the main functions of the system, including Add Record, Edit Record, Report/Summary, Search Record, Delete Record, and Exit System. This makes the system more organized and convenient for users who need to manage student grades.

Python has also been widely used in software and system development because of its readability, simplicity, and flexibility. Virtanen et al. (2020) discussed Python's importance in scientific computing and system development, particularly because it supports efficient programming and data processing. In the GPA Tracker System, Python serves as the main programming language used to develop the system's logic and functions. It allows the system to perform operations such as input validation, GPA computation, data retrieval, and record management.

Database management is another important part of the system. Gaffney et al. (2022) explained that SQLite is a

lightweight and reliable database engine commonly used in applications that require organized data storage and efficient data processing. This supports the use of SQLite3 in the GPA Tracker System because the system needs to store student names, student IDs, ages, subjects, grades, and computed GPA. With a database, records can be stored more securely and retrieved more easily compared to paper-based or spreadsheet-based methods.

These related studies show that GUI design, Python programming, and SQLite database management are suitable tools for developing a simple but functional academic record management system. They support the idea that computerized systems can improve the speed, accuracy, and organization of student record handling.

REVIEW OF RELATED LITERATURE

Graphical User Interface in Academic Systems

A graphical user interface, or GUI, allows users to interact with a system using visual components such as windows, buttons, labels, menus, and text fields. Compared to command-line systems, GUI-based applications are easier for many users because commands are presented visually. This type of interface is helpful in academic systems because users can complete tasks with less technical difficulty.

In the GPA Tracker System, the GUI allows users to perform different record management tasks in a simple and direct manner. Instead of typing complex commands, users can click buttons to add, edit, search, delete, or view student records. This improves usability and reduces the possibility of user confusion. The GUI also helps organize the system's features, making the program more understandable and accessible.

Python as a Programming Language

Python is a high-level programming language known for its simple syntax, readability, and flexibility. It is commonly used in developing educational systems, desktop applications, automation tools, and database-connected programs. Python is also suitable for beginners because its structure is easier to understand compared to many other programming languages.

In this study, Python was used to develop the GPA Tracker System because it can handle mathematical computations, data processing, and system operations efficiently. The system uses Python to process user inputs, validate records,

compute GPA, and connect with the SQLite3 database. Its support for GUI libraries such as Tkinter and CustomTkinter also makes it appropriate for creating a user-friendly desktop application.

SQLite3 Database Management

A database is important in any record management system because it provides a structured way of storing and retrieving information. SQLite3 is a database engine that is lightweight, simple to use, and suitable for offline applications. It does not require a separate server, making it appropriate for small-scale systems and academic projects.

In the GPA Tracker System, SQLite3 is used to store student records, including personal information and academic grades. This makes the system more organized than manual recording because data can be saved, searched, edited, and deleted efficiently. The use of SQLite3 also helps prevent the loss of records and supports faster retrieval of student information.

Academic Record Management

Academic record management refers to the process of storing, organizing, updating, and retrieving student information such as names, identification numbers, subjects, grades, and academic performance. Accurate academic records are important because they serve as the basis for evaluating student progress.

Manual record management can be difficult, especially when there are many students and subjects involved. Common problems include incorrect entries, duplicate records, misplaced documents, and inaccurate grade computation. These problems can affect the reliability of student records and make academic monitoring less efficient.

The GPA Tracker System addresses these issues by providing a computerized method for managing records. Through its database and automatic GPA computation, the system reduces manual work and improves accuracy. It also allows users to retrieve records faster, making academic monitoring more convenient.

GPA Computation

Grade Point Average, or GPA, is a numerical representation of a student's academic performance. It is usually computed based on the grades earned in different subjects and the corresponding units or weights of those subjects. Manual GPA computation can be prone to errors, especially when several subjects and units are involved.

The GPA Tracker System automates this process by computing GPA based on the encoded grades and predefined subject units. This helps ensure consistency and accuracy in the computation. By automating the process, the system reduces the possibility of miscalculation and saves time for users.

Synthesis of the Reviewed Theory, Studies, and Literature

The reviewed theories, studies, and literature show that the development of a GUI-based GPA Tracker System is supported by established concepts in technology acceptance, information system success, interface design, programming, and database management. The Technology Acceptance Model supports the importance of usefulness and ease of use in system adoption. The Information Systems Success Model supports the need for system quality, information accuracy, usability, and user satisfaction.

Related studies also show that GUI-based applications can improve user interaction by making systems easier to navigate. Python is suitable for developing the system because of its simplicity and ability to handle computations and database operations. SQLite3 is appropriate for the system because it provides a lightweight and organized way to store records.

Overall, the reviewed materials support the development of the GPA Tracker System as a practical solution to the problems of manual academic record management. The system's features, such as adding, editing, searching, deleting, generating summaries, and computing GPA, are designed to improve efficiency, accuracy, and usability.

Conceptual Framework

The conceptual framework of this study follows an Input-Process-Output model.

The input includes student information, student ID, age, subjects, grades, and subject units. These data are entered into the system by the user through the graphical user interface.

The process includes data validation, record storage, record updating, searching, deletion, report generation, and GPA computation. The system checks whether the entered information is complete and valid before saving it in the database. It also performs automatic computation of GPA based on the encoded grades and units.

The output includes organized student records, updated academic information, search results, deleted or modified records, report/summary records, and computed GPA. These outputs help users monitor student academic performance more efficiently.

Through this framework, the GPA Tracker System transforms manually encoded academic information into organized, accurate, and accessible digital records.

METHODOLOGY

This chapter presents the research design, system development process, system design, database integration, tools and technologies used, system features, testing procedure, and evaluation method used in the development of the GUIbased GPA Tracker System. The methodology explains how the system was planned, designed, developed, tested, and implemented to achieve the objectives of the study.

Research Design

This study used a system development research design. This design was appropriate because the main purpose of the study was to develop a functional GPA Tracker System that can record, manage, retrieve, update, delete, and compute student academic records efficiently.

The development of the system was based on the identified problem that manual recording and computation of grades can be time-consuming and prone to errors. In many academic situations, student records are still managed manually or through separate applications, which may result in misplaced records, duplicate entries, delayed updates, and inaccurate GPA computation. Because of these challenges, the researchers developed a computerized system that can make student record management faster, more organized, and more accurate.

The GPA Tracker System was designed for Bachelor of Science in Computer Engineering students and users who need a simple tool for managing academic records. The system allows users to store student information, encode grades, compute GPA, and access academic records conveniently. It was developed using Python as the main programming language, Tkinter/CustomTkinter for the graphical user interface, and SQLite3 for database management.

System Development Process

The development of the GPA Tracker System followed a step-by-step process to ensure that the system would function properly and meet the objectives of the study.

Problem Identification and Planning

The first phase involved identifying the problem related to manual student grade recording and GPA computation. The researchers observed that manual recording can cause errors, delays, and difficulty in retrieving student information, especially when handling many student records.

During this phase, the researchers planned a system that could help users manage student records in one platform. The proposed system included basic academic record management features such as adding, viewing, editing, searching, deleting, and computing student GPA.

System Design

The second phase focused on designing the flow and structure of the system. A flowchart was created to guide the developers in understanding how the system should operate. The system starts by displaying the main menu, where the user can choose from several options such as Add Record, Edit Record, Report/Summary, Search Record, Delete Record, and Exit System.

The system design also considered how data would be entered, validated, stored, retrieved, updated, and deleted. The researchers planned the interface and system functions to ensure that users could interact with the system easily and that the records would be handled properly.

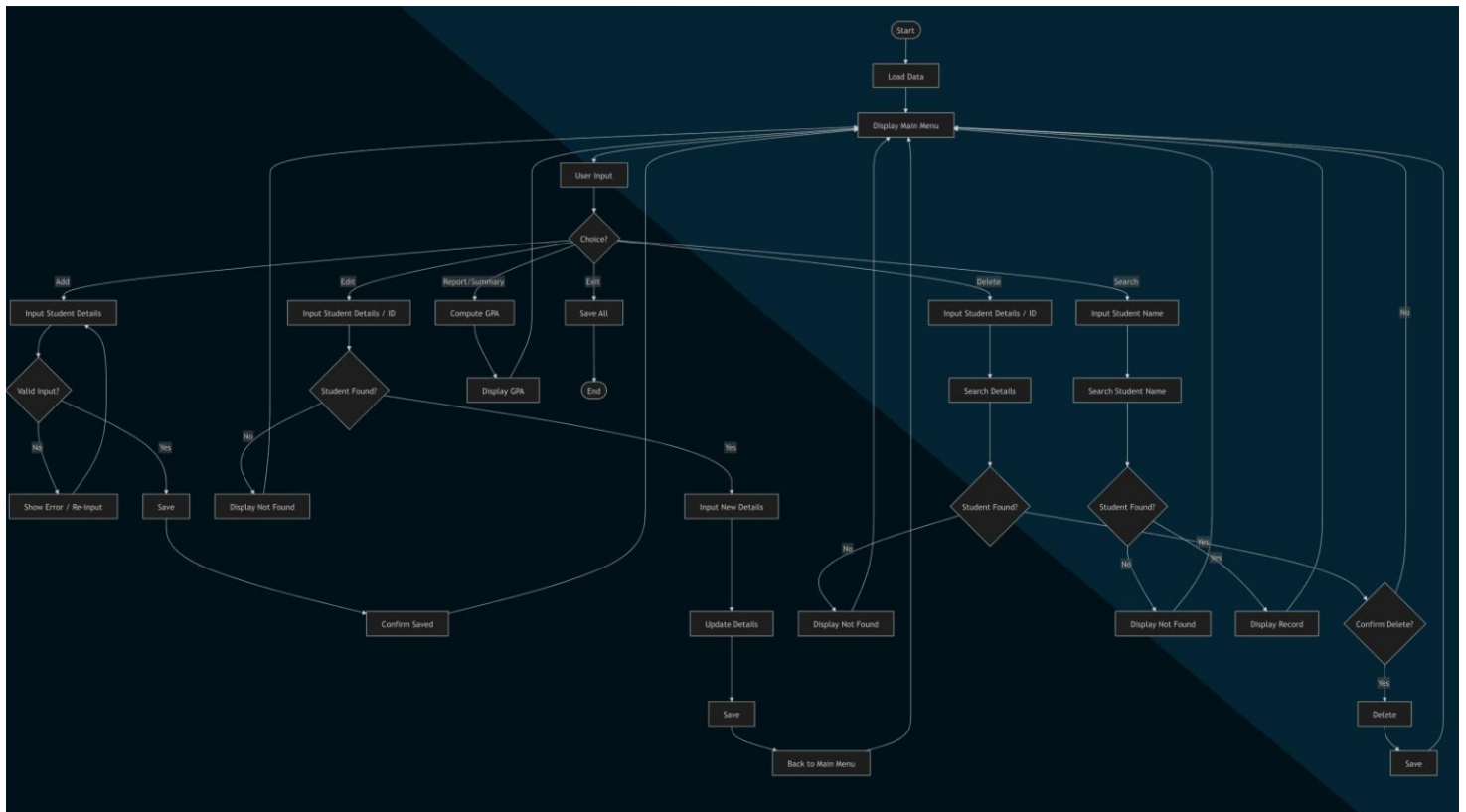


Figure 1. Flowchart

This flowchart shows the process of how the GPA Tracker works. The system starts by displaying the menu first, wherein there are six buttons: Add, Edit, Report/ Summary, Search, Delete, and Exit. The user will be able to choose the task they want to perform. If “Add” is chosen, the user will input the students’ information and grades. The system checks if the inputs are invalid, and if they are, it will display an error message and ask the user to re-enter a valid record. After that, the data can be saved and stored. The “Edit” button allows the user to enter a student ID and name to search for a record. If the student is found, new details can be entered and updated. If not found, the system displays “Not Found.” After editing, the updated record can be saved and stored. The “Report/ Summary” displays the students’ records and grades and also computes their GPA. The “Search” button is used for finding student records by searching for a student’s name. If the student exists, the record is displayed, otherwise, a “Not Found” message appears. Next is the “Delete” button, which allows users to search for a student record and confirm whether to delete it. If confirmed, the record is deleted. Lastly, the “Exit” button is used to end the program while ensuring that all records are saved.

System Development

The third phase involved the actual coding and development of the GPA Tracker System using Python. The system was first designed to process user input through a menu-driven structure. It used important programming concepts such as loops, conditional statements, and functions to control the flow of the program and organize the code properly.

Loops were used to allow the system to continue running until the user selected the Exit option. Conditional statements were used to check user choices and validate inputs. Functions were used to separate the major tasks of the system, making the program more organized, readable, and easier to maintain.

The major functions developed in the system include adding student records, editing existing records, searching records, deleting records, generating reports, and computing GPA. These functions allow users to manage academic records more conveniently and accurately.

Database Integration

The fourth phase involved integrating a database into the system. The GPA Tracker System used SQLite3 as its database management system. SQLite3 was selected because it is lightweight, easy to use, compatible with Python, and suitable for offline desktop applications.

The system used a database file named `gpa_tracker.db` to store student information and academic records permanently. Unlike temporary storage using lists or variables, the SQLite database keeps records saved even after the program is closed.

The database structure included two main tables: `students` and `grades`. The `students` table stores personal information such as student ID, name, age, and the date the record was created. The `grades` table stores academic data such as subject, units, and grade. The `grades` table is connected to the `students` table through a student ID reference, allowing each student's academic records to be properly linked to their personal information.

Through database integration, the system can add, retrieve, update, delete, and search records efficiently. This makes the system more reliable and organized compared to manual recording or temporary data storage.

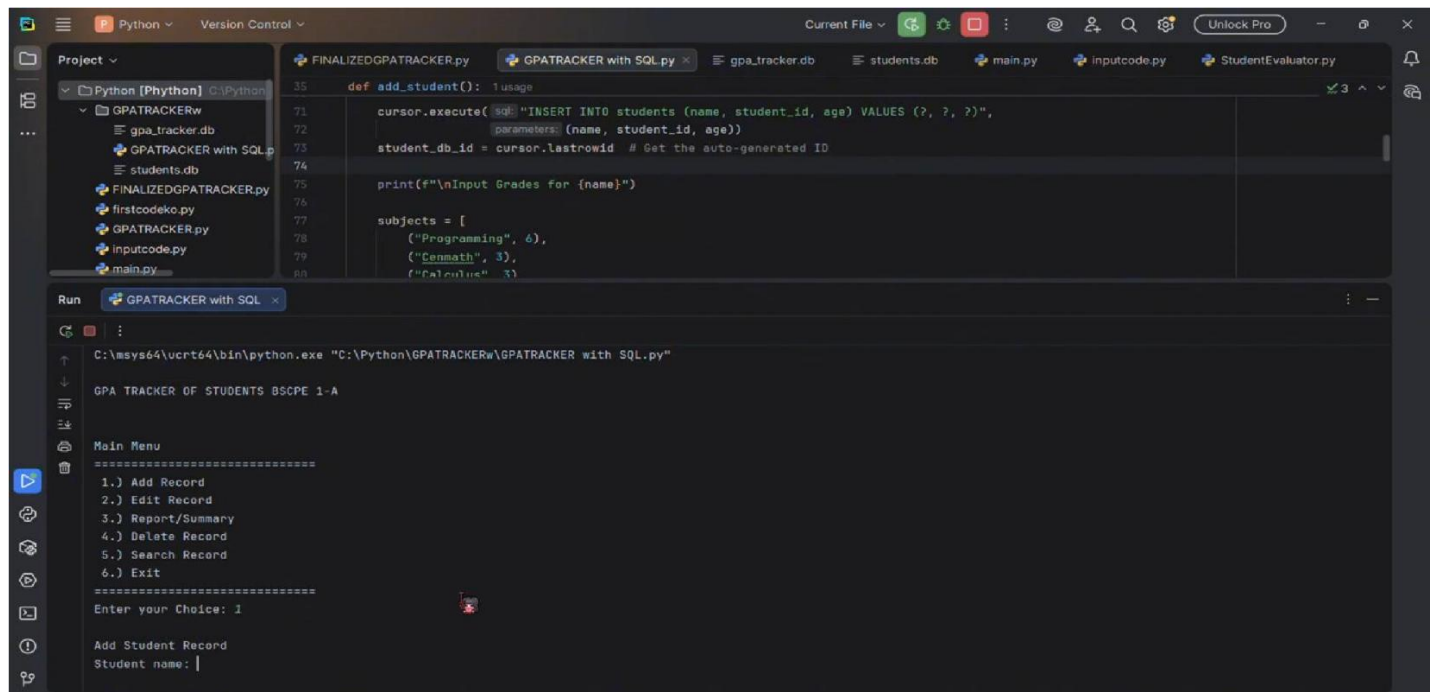


Figure 2. Database Integration of the GPA Tracker System using SQLite3

This figure shows the database integration of the GPA Tracker System in Python. The system uses SQLite3 to store student information and grade records permanently, allowing users to add, edit, search, delete, and retrieve student data efficiently.

Graphical User Interface Implementation

The fifth phase involved implementing the graphical user interface using Tkinter/CustomTkinter. The GUI was added to make the system more user-friendly and visually organized. Instead of using only text-based commands, users can interact with the system through buttons, input fields, labels, and windows.

The GUI allows users to access the major system functions easily. It includes buttons for Add Record, Edit Record, Report/Summary, Search Record, Delete Record, and Exit System. This improves usability because users do not need to memorize commands. The visual layout also makes the system easier to understand, especially for beginners.

System Testing and Debugging

After the system was developed, it was tested to check if all features were working correctly. The researchers entered different types of valid and invalid data to determine whether the system could handle common errors.

Testing was conducted on the system's main functions, including adding records, editing records, searching records, deleting records, generating reports, computing GPA, and exiting the system. Input validation was also tested to ensure that the system could detect invalid names, duplicate student IDs, empty fields, invalid ages, and invalid grades.

Errors encountered during testing were corrected through debugging. The researchers reviewed the code, identified mistakes, and improved the system to make it more reliable and user-friendly.

System Implementation

After testing and debugging, the system was implemented as an offline GUI-based desktop application. The final system allowed users to manage student academic records through a graphical interface and save records permanently in an SQLite database.

The implemented system provides users with a faster and more organized way of managing student grades and GPA compared to manual methods. It also helps reduce human errors through validation and automatic GPA computation.

System Design

The GPA Tracker System was designed to follow a simple and organized process. When the program starts, the main menu is displayed. The user can then choose from the available system functions.

If the user selects Add Record, the system asks for the student's information, such as student ID, name, age, subjects, units, and grades. The system checks whether the input is complete and valid. If the input is invalid, the system displays an error message and asks the user to correct the information. If the input is valid, the record is saved in the database.

If the user selects Edit Record, the system allows the user to search for an existing student record. If the record is found, the user can update the student's information and grades. The updated record is then saved in the database.

If the user selects Report/Summary, the system displays student records and computes the GPA based on the encoded grades and subject units. This feature helps users view academic performance in an organized way.

If the user selects Search Record, the system allows the user to search for a specific student by name or student ID. If the record exists, the system displays the student's information. If the record is not found, the system displays a message informing the user that no matching record was found.

If the user selects Delete Record, the system allows the user to search for a student record and confirm whether it should be removed. Once confirmed, the record is deleted from the database.

If the user selects Exit System, the program closes properly while ensuring that saved records remain stored in the database.

Data Structure and Database Design

During the early design of the system, the researchers considered the use of lists and dictionaries as data structures in

Python. A list can store multiple student records, while a dictionary can store each student's information such as name, ID, age, subjects, and grades. This structure helped the researchers understand how student records could be organized and accessed in the program.

However, the final system used SQLite3 to provide permanent data storage. This was necessary because lists and dictionaries only store data temporarily while the program is running. Once the program is closed, temporary data may be lost. By using SQLite3, the system can store records permanently and retrieve them whenever needed.

The database design of the GPA Tracker System includes two main tables:

Students Table

The students table stores personal information about each student. It includes fields such as the student's unique ID, name, age, and date created. This table helps organize the personal records of students.

Grades Table

The grades table stores academic information, including subject names, subject units, and grades. It is connected to the students table so that each grade record belongs to a specific student.

This database structure allows the system to manage student information and academic grades in a more organized and reliable way.

Tools and Technologies Used

The following tools and technologies were used in the development of the GPA Tracker System:

Python

Python was used as the main programming language of the system. It was chosen because of its simple syntax, readability, and flexibility. Python was used to create the system logic, input validation, GPA computation, and database operations.

Tkinter/CustomTkinter

Tkinter/CustomTkinter was used to create the graphical user interface of the system. It provided the buttons, windows, input fields, and labels needed for user interaction. Through the GUI, users can manage records without using complex commands.

SQLite3

SQLite3 was used as the database management system of the GPA Tracker. It stores student information, grades, and academic records permanently. It was selected because it is lightweight, reliable, and supported by Python.

PyCharm/Python IDE

PyCharm or another Python IDE was used for writing, editing, testing, and debugging the program code. This tool helped the developers identify errors and improve the system during development.

System Features

The GPA Tracker System includes the following features:

1. **Add Record** – allows users to encode and save new student information and grades.
2. **Edit Record** – allows users to update existing student information and academic records.
3. **Report/Summary** – displays student records and computes GPA.
4. **Search Record** – allows users to find student records quickly using student information.
5. **Delete Record** – allows users to remove records from the database after confirmation.
6. **Input Validation** – checks invalid names, duplicate student IDs, empty fields, invalid age input, and invalid grade input.
7. **Database Storage** – stores student records permanently using SQLite3.
8. **Exit System** – allows users to close the program properly.

Testing Procedure

The system was tested using functional testing. Each feature was tested to determine whether it produced the expected result. The researchers tested both valid and invalid inputs to check the accuracy, reliability, and usability of the system.

The following test conditions were used:

Test Case	Expected Result
Invalid name input	The system displays an error message for invalid characters or incomplete names.
Duplicate student ID	The system shows a warning that the student ID already exists.
Empty field	The system asks the user to complete all required fields.
Invalid age input	The system shows an error message if the age is not a whole number.
Invalid grade input	The system shows an error message if the grade is outside the accepted range.
Add student record	The record is saved successfully in the database.
Edit student record	The student information is updated successfully.
Search student record	The system displays the correct student information.
Delete student record	The record is removed successfully from the database.
GPA computation	The system computes and displays GPA accurately.
Report generation	The system displays student records and GPA summary properly.
Exit system	The system closes properly without errors.

The purpose of the testing procedure was to ensure that the system could perform its functions correctly and prevent common user errors.

Evaluation of the System

The system was evaluated based on functionality, accuracy, usability, reliability, and data management.

Functionality was evaluated by checking whether the system could successfully add, edit, search, delete, display, and compute student records. Accuracy was evaluated by checking whether the system computed GPA correctly and stored student information properly. Usability was evaluated based on the clarity and simplicity of the graphical user interface. Reliability was evaluated by checking whether the system could save, retrieve, update, and delete records consistently. Data management was evaluated by checking whether the SQLite database stored records permanently and organized them properly.

The evaluation showed that the GPA Tracker System was able to perform its intended functions. It successfully managed student records, validated inputs, computed GPA, generated summaries, and saved records in the database.

Limitations Encountered During Development

During the development of the system, the researchers encountered several limitations. Since the developers were still beginners in Python and SQLite, they experienced challenges in understanding new commands, functions, and database operations. Some coding errors also occurred during testing and debugging.

Another limitation is that the system is intended for offline use only. The records are stored in a local SQLite database, which means the system can only be accessed on the device where it is installed. It does not yet support online access, cloud storage, automatic backup, multi-user access, or user authentication.

The system also handles only the predefined subjects and units encoded by the developers. Therefore, it may need further improvement if it will be used for a larger number of students or for different courses and grading systems.

Summary

The methodology of this study followed a system development process that included problem identification, planning, system design, development, database integration, GUI implementation, testing, and evaluation. The GPA Tracker System was developed using Python, Tkinter/CustomTkinter, and SQLite3. Its main purpose is to provide an organized, accurate, and user-friendly system for managing student academic records and computing GPA.

Through the development process, the researchers were able to apply programming concepts such as loops, conditional statements, functions, input validation, database management, and graphical user interface design. The system also helped the developers improve their programming, problem-solving, teamwork, and system development skills.

RESULTS AND DISCUSSION

This chapter presents the results and discussion of the developed GUI-based GPA Tracker System. It discusses the system overview, completed system features, interface design, testing results, and overall performance of the system. The results show how the GPA Tracker System was able to perform its intended functions in managing student records, computing GPA, validating user input, and storing records using a database.

System Overview

The GPA Tracker System was developed as a student academic record management system for Bachelor of Science in Computer Engineering students. Its main purpose is to help users record, organize, update, search, delete, and compute student academic records in a faster and more accurate way.

The system was developed using Python as the main programming language, Tkinter/CustomTkinter for the graphical user interface, and SQLite3 as the database management system. Through these tools, the developers were able to create a functional desktop application that allows users to manage student information and grades in one system.

The system stores important student information such as student name, student ID, age, subjects, grades, and computed GPA. It also allows users to perform basic record management operations such as adding, editing, searching, deleting, and viewing student records. Since the system uses SQLite3, the records are saved permanently in a local database and can still be accessed even after the program is closed.

Overall, the system provides a more organized and efficient method of managing student academic records compared to manual recording. It reduces the possibility of misplaced records, duplicate entries, and incorrect GPA computation.

System Features and Results

The developed GPA Tracker System includes several features that support student record management and GPA computation. Each feature was designed to address the problems commonly encountered in manual recording and computation.

Add Record

The Add Record feature allows the user to input and save new student information. The user can enter the student's name, student ID, age, subjects, units, and grades. Before saving the record, the system checks if the entered information is valid and complete.

The result showed that the Add Record feature successfully saved student information and grades into the database. It also prevented invalid entries such as empty fields, invalid names, duplicate student IDs, invalid age input, and invalid grade values. This feature helps ensure that the stored records are accurate and complete.

Edit Record

The Edit Record feature allows the user to update existing student information and grades. This feature is useful when there are corrections or changes in a student's record. Instead of deleting the entire record and adding it again, the user can simply edit the existing information.

The result showed that the Edit Record feature successfully updated student records in the database. It allowed users to modify details while maintaining data accuracy through validation. This improves convenience and saves time in managing student records.

Report/Summary Record

The Report/Summary feature displays the saved student records and computes the Grade Point Average. The GPA is computed based on the encoded grades and corresponding subject units. This feature helps users view the academic performance of students in an organized way.

The result showed that the Report/Summary feature successfully displayed student information and computed GPA accurately. It helped reduce manual computation and made the evaluation of student academic performance faster and easier.

Search Record

The Search Record feature allows the user to quickly find a student record using the student's name or student ID. This feature prevents users from checking every record manually.

The result showed that the Search Record feature successfully displayed the correct student information when a matching record was found. If no matching record existed, the system displayed a "Not Found" message. This feature made record retrieval faster and more convenient.

Delete Record

The Delete Record feature allows the user to remove student records that are no longer needed. Before deleting a record, the system asks for confirmation to prevent accidental deletion. The result showed that the Delete Record feature successfully removed selected student records from the database. This helps keep the system clean, updated, and organized.

Exit System

The Exit System feature allows the user to properly close the program. Since the system uses a database, saved records remain stored even after the program is closed. The result showed that the Exit System feature closed the program properly without errors. The records remained saved in the SQLite3 database and could still be accessed when the system was opened again.

System Interface

The GPA Tracker System uses a graphical user interface to make the system easier to use. The interface includes buttons, input fields, labels, and windows that guide the user in performing different tasks.

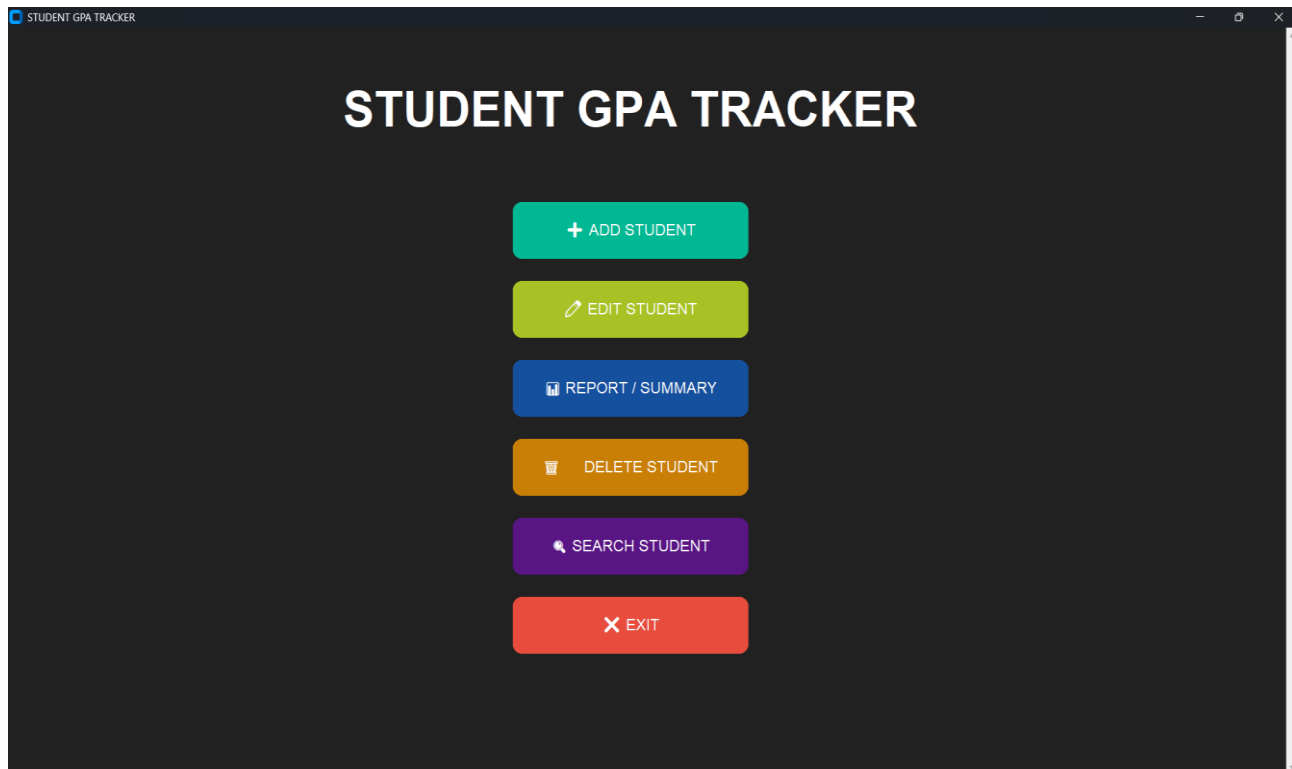


Figure 3. Main GUI Window

The Main GUI Window serves as the main menu of the system. It contains the main buttons for Add Record, Edit Record, Report/Summary, Search Record, Delete Record, and Exit System. This allows users to access all major functions from one organized window.

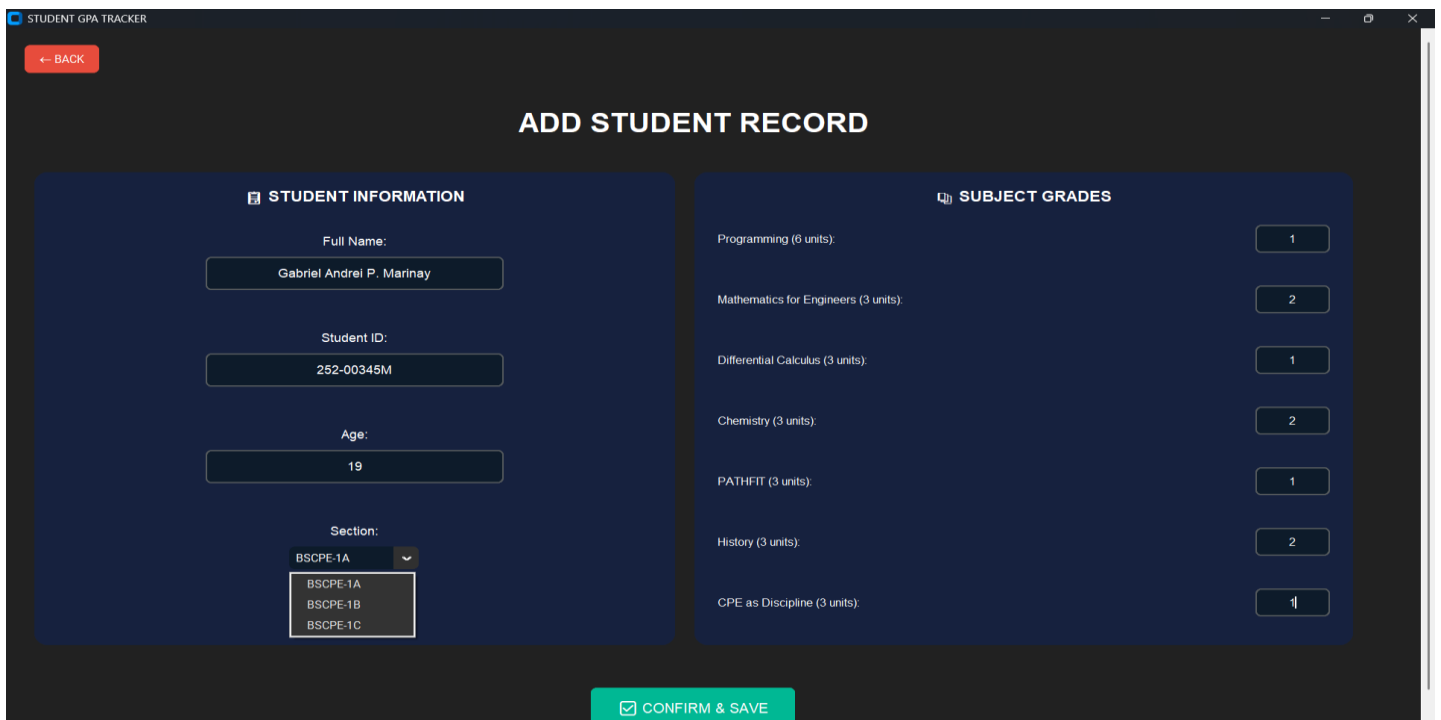
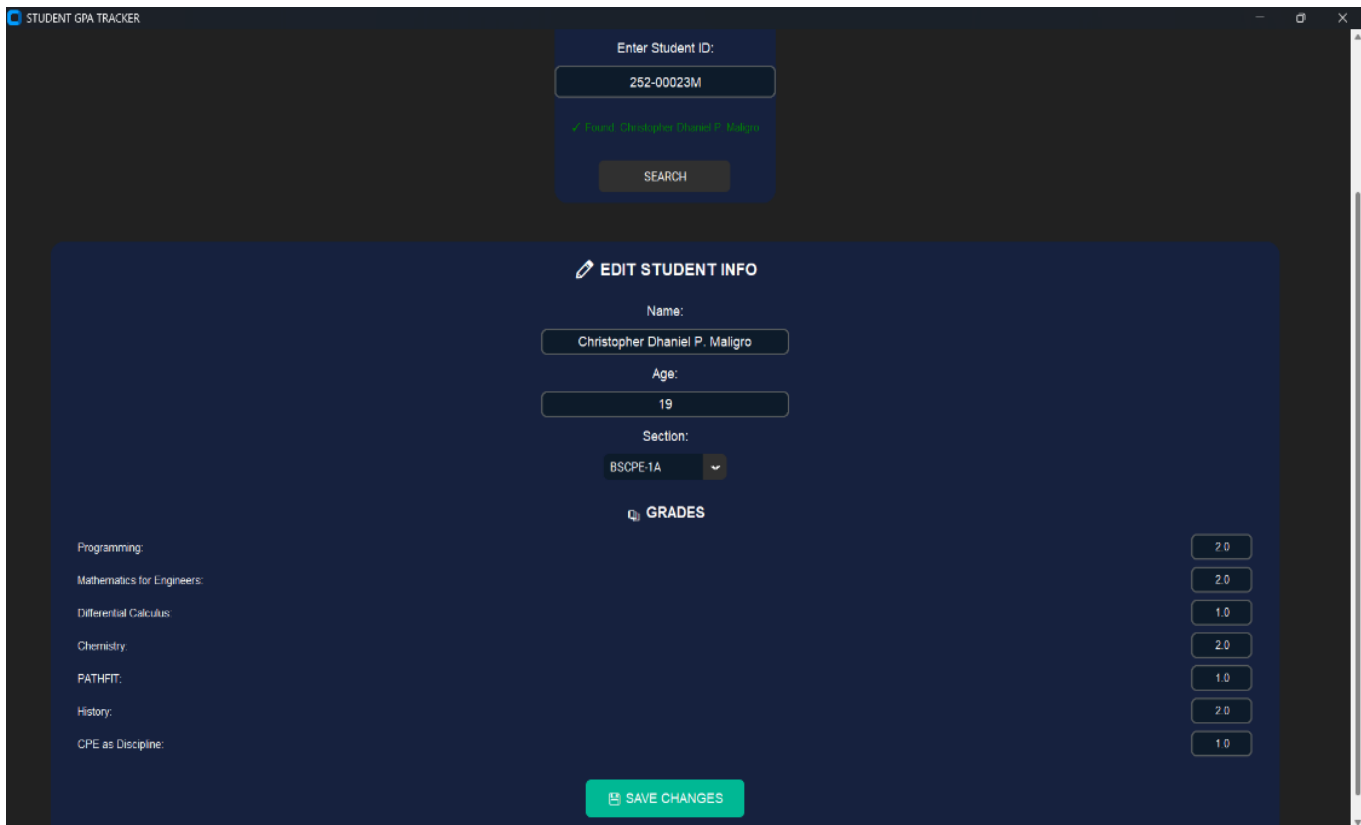


Figure 4. Add Record Window

The Add Record Window allows users to encode new student information and grades. It contains input fields for student details and academic information. This window helps users add records in a structured format.



STUDENT GPA TRACKER

Enter Student ID:
252-00023M
✓ Found: Christopher Dhanial P. Maligro
SEARCH

EDIT STUDENT INFO

Name: Christopher Dhanial P. Maligro
Age: 19
Section: BSCPE-1A

GRADES

Programming:	2.0
Mathematics for Engineers:	2.0
Differential Calculus:	1.0
Chemistry:	2.0
PATHFIT:	1.0
History:	2.0
CPE as Discipline:	1.0

SAVE CHANGES

Figure 5. Edit Record Window

The Edit Record Window allows users to search and update existing student records. This interface helps users correct mistakes or update information without creating a new record.



STUDENT GPA TRACKER

← BACK

REPORT / SUMMARY

=====

REPORT FOR BSCPE-1A

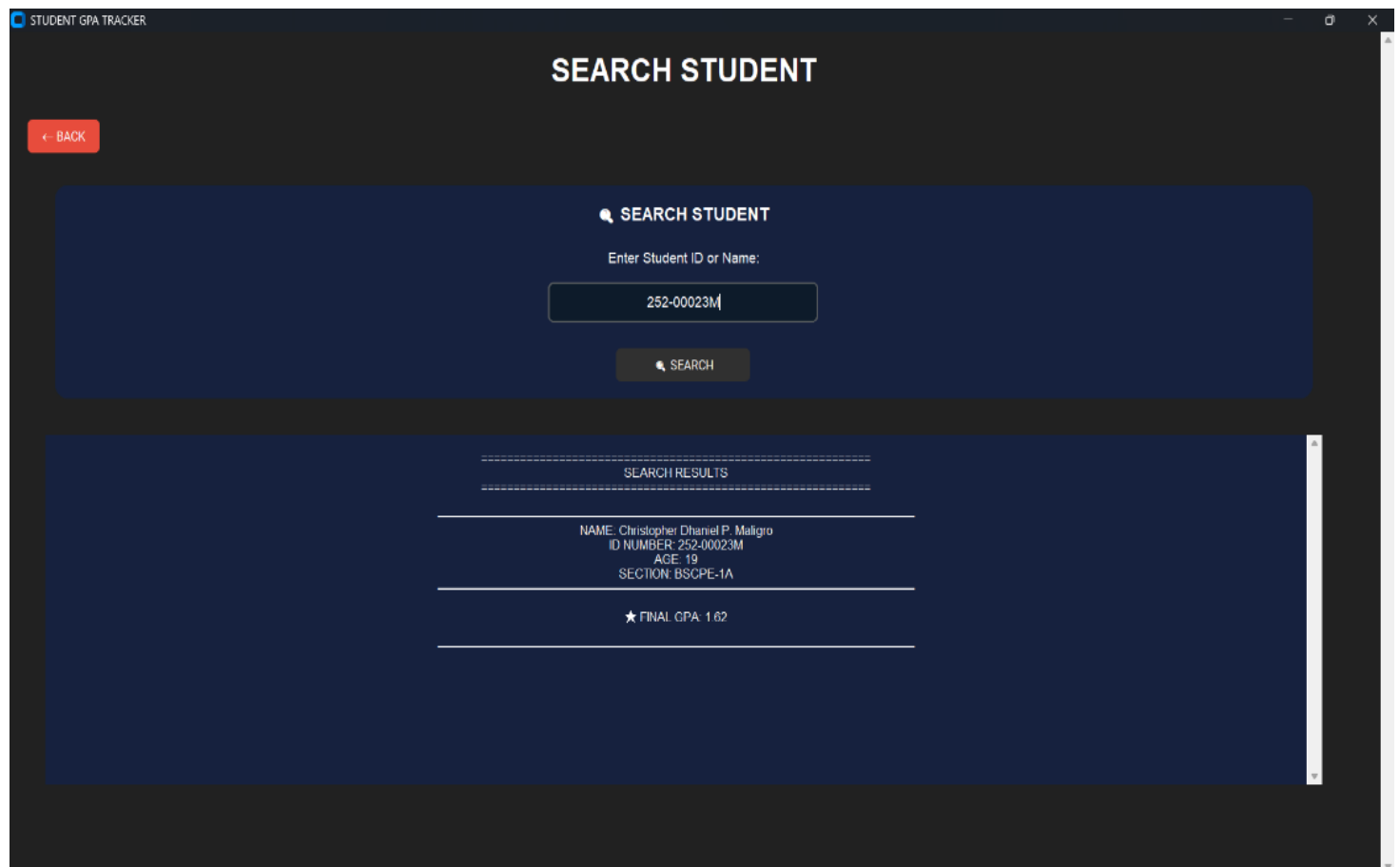
=====

NAME: Christopher Dhanial P. Maligro ID NUMBER: 252-00023M AGE: 19 SECTION: BSCPE-1A
★ FINAL GPA: 1.62
NAME: Gabriel Andrei P. Marinay ID NUMBER: 252-00234M AGE: 18 SECTION: BSCPE-1A
★ FINAL GPA: 1.00
NAME: Hannah Rosento ID NUMBER: 252-00043m AGE: 18 SECTION: BSCPE-1A
★ FINAL GPA: 1.00
NAME: Jazmine Claire M. Capati ID NUMBER: 252-00235F AGE: 19 SECTION: BSCPE-1A

← BACK TO SECTIONS

Figure 6. Report/Summary Window

The Report/Summary Window displays student records and computed GPA. This allows users to view academic summaries and monitor student performance.



The screenshot shows a web application window titled "STUDENT GPA TRACKER". The main heading is "SEARCH STUDENT". There is a red "← BACK" button in the top left. The search area has a dark blue background with a magnifying glass icon and the text "SEARCH STUDENT". Below this is a label "Enter Student ID or Name:" and a text input field containing "252-00023M". A "SEARCH" button with a magnifying glass icon is below the input field. The results section, titled "SEARCH RESULTS", displays the following information: NAME: Christopher Dhamiel P. Maligro, ID NUMBER: 252-00023M, AGE: 19, SECTION: BSCPE-1A, and ★ FINAL GPA: 1.62.

SEARCH RESULTS
NAME: Christopher Dhamiel P. Maligro
ID NUMBER: 252-00023M
AGE: 19
SECTION: BSCPE-1A
★ FINAL GPA: 1.62

Figure 7. Search Record Window

The Search Record Window allows users to find student information quickly. It helps users retrieve records without manually browsing through all saved data.

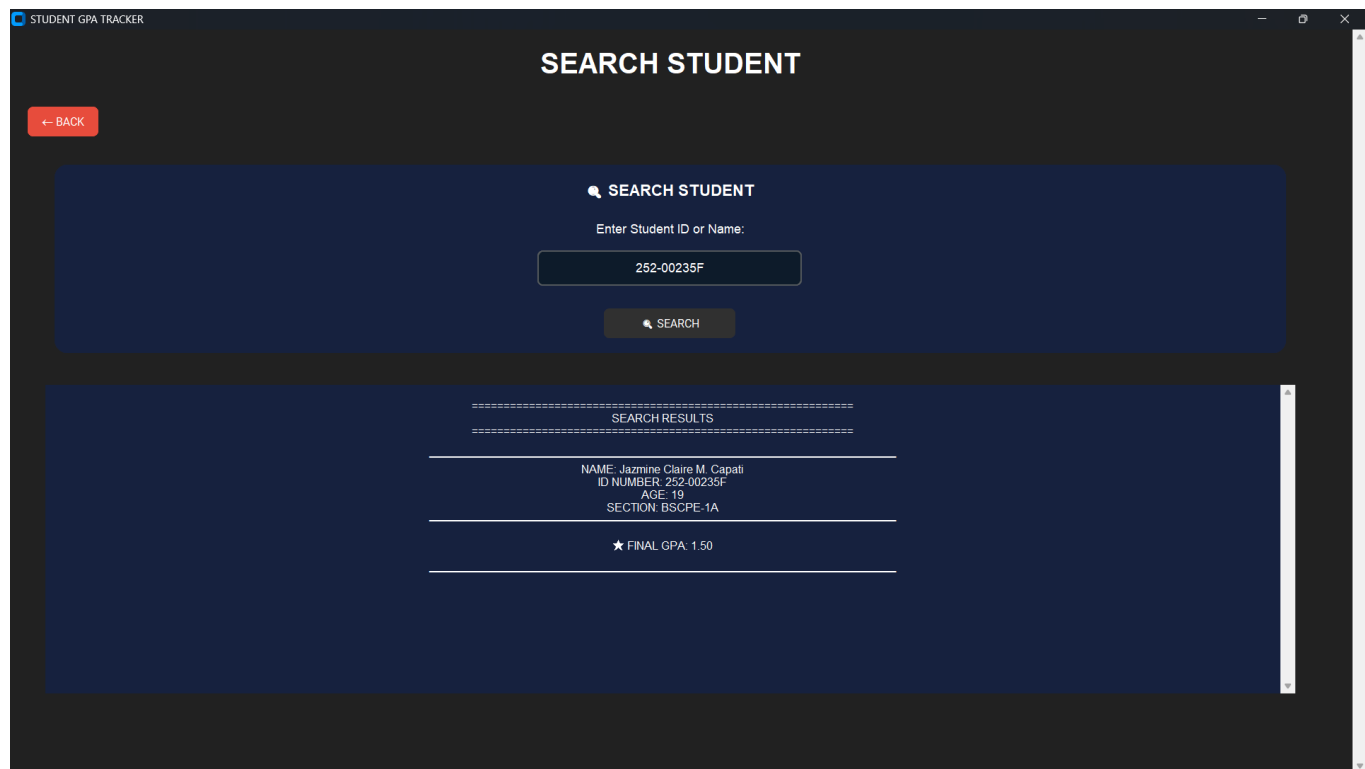


Figure 8. Delete Record Window

The Delete Record Window allows users to remove selected records from the database. It includes confirmation to prevent accidental deletion.

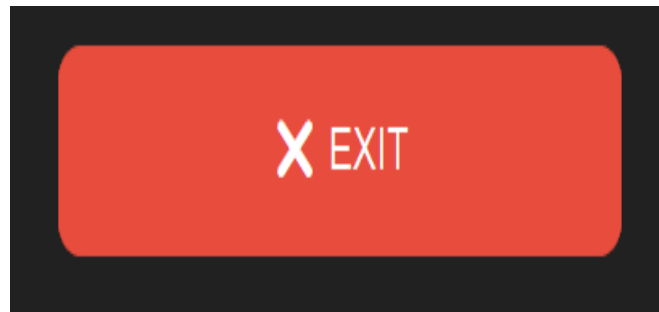


Figure 9. Exit System

The Exit System function closes the program properly while keeping the records saved in the database.

The use of a graphical user interface improved the usability of the system. It made the system easier to understand and operate, especially for users who are not familiar with command-based programs.

System Testing Results

The system was tested to check whether each feature produced the expected result. Functional testing was conducted by entering valid and invalid data into the system. The testing focused on input validation, database storage, GPA computation, record retrieval, editing, deleting, report generation, and system exit.

Test Case	Result
Invalid name input	The system displayed an error message for invalid characters or incomplete names.
Duplicate student ID	The system showed a warning that the student ID already existed.
Empty required field	The system asked the user to complete all required fields.

Invalid age input	The system displayed an error message if the age was not a whole number.
Invalid grade input	The system displayed an error message if the grade was outside the accepted range.
Add student record	The record was successfully saved in the database.
Edit student record	The student information was successfully updated.
Search student record	The system displayed the correct student information.
Delete student record	The selected record was successfully removed from the database.
GPA computation	The system computed and displayed the GPA accurately.
Report generation	The system displayed student records and GPA summary properly.
Exit system	The system closed properly without errors.

The testing results showed that the system was able to perform its intended functions. It successfully validated user inputs, stored records, updated information, retrieved records, deleted records, computed GPA, and generated summaries.

DISCUSSION OF FINDINGS

The results of the system development and testing showed that the GPA Tracker System was effective in managing student academic records. The system successfully provided the major functions needed in academic record management, including adding, editing, searching, deleting, viewing summaries, and computing GPA.

The use of SQLite3 helped improve the reliability of the system because it allowed records to be stored permanently. Unlike temporary storage using lists or variables, the database kept student records saved even after the system was closed. This made the system more useful and dependable for record management.

The system also helped reduce errors commonly encountered in manual recording. Input validation prevented invalid names, duplicate IDs, empty fields, invalid ages, and invalid grades from being saved. This improved the accuracy and completeness of the records.

Another important result was the successful implementation of automatic GPA computation. Manual GPA computation can be time-consuming and prone to miscalculation, especially when several subjects and units are involved. Through the system, GPA computation became faster and more consistent.

The graphical user interface also improved user interaction. Instead of using only command-line instructions, users could interact with the system through buttons and input fields. This made the system more understandable, organized, and user-friendly.

Overall, the findings show that the GPA Tracker System achieved its purpose of improving the management of student academic records. It provided a practical solution for recording, retrieving, updating, deleting, and computing academic information.

Strengths of the System

The system has several strengths based on the results of development and testing.

First, it organizes student records in one system, making it easier for users to manage information. Second, it uses an SQLite3 database, allowing records to be saved permanently. Third, it includes input validation, which helps prevent incorrect or incomplete data from being stored. Fourth, it automatically computes GPA, reducing the need for manual calculation. Fifth, the graphical user interface makes the system easier to use and understand.

These strengths show that the system is useful for academic record management and can help users save time while improving accuracy.

Limitations Observed

Although the system was able to perform its intended functions, some limitations were observed. The system is still intended for offline use only, which means records can only be accessed on the device where the program and database are installed. It does not yet support cloud storage, online access, or multi-user access.

The system also does not include advanced features such as user authentication, automatic backup, role-based access, semester-based records, GPA ranking, or exporting reports to Excel or PDF. In addition, the GPA computation is limited to the predefined subjects and units encoded in the system.

These limitations show that the system is functional but can still be improved in future versions.

Summary

The results showed that the GPA Tracker System successfully performed its intended functions. It was able to add, edit, search, delete, summarize, and compute student GPA records. The system also validated user input and stored records permanently using SQLite3.

The discussion showed that the system improved the accuracy, organization, and efficiency of student academic record management. It reduced manual errors, provided faster record retrieval, and made GPA computation more convenient. Through its graphical user interface, the system also became more user-friendly and easier to operate.

Therefore, the GPA Tracker System can be considered an effective and reliable tool for managing student academic records within the scope and limitations of the study.

CONCLUSIONS AND RECOMMENDATIONS

Conclusions

Based on the development, testing, and evaluation of the GPA Tracker System, the researchers concluded that the system successfully achieved its main objective of developing a GUI-based application for managing student academic records and computing Grade Point Average (GPA). The system was able to perform the major functions needed for student record management, including adding, editing, searching, deleting, viewing reports, and computing GPA.

The system effectively handled the process of recording students' grades by allowing users to encode student information and academic records in an organized digital format. Through the use of Python, Tkinter/CustomTkinter, and SQLite3, the system provided a functional and user-friendly platform for managing student data. The use of SQLite3 allowed the records to be stored permanently in a local database, making the system more reliable compared to temporary storage or manual recording.

The GPA Tracker System also proved effective in reducing errors commonly encountered in manual recording. The system included input validation that detects invalid names, duplicate student IDs, empty fields, invalid age input, and incorrect grade values. These validation features helped prevent inaccurate and incomplete data from being saved in the database. In addition, the automatic GPA computation reduced the possibility of manual calculation errors and made the process faster and more consistent.

The implementation of a graphical user interface improved the usability and accessibility of the system. Through buttons, input fields, labels, and organized windows, users were able to interact with the system more easily. This made the system more understandable, especially for users who are not familiar with command-based programs.

Overall, the GPA Tracker System served as an effective and reliable tool for managing student academic records within the scope of the study. It improved the accuracy, organization, efficiency, and convenience of recording grades and computing GPA. The project also helped the developers apply important programming concepts such as functions, loops, conditional statements, input validation, database management, and GUI development. Therefore, the system successfully met its intended purpose as a basic academic record management application.

Recommendations

Based on the findings and limitations of the study, the following recommendations are proposed for future improvement of the GPA Tracker System:

1. Future developers should add a user authentication and login system to improve the security of student records. This feature would allow only authorized users, such as teachers or administrators, to access and manage the system.
2. The system should include role-based access control so that different users can have different permissions. For example, administrators may be allowed to add, edit, and delete records, while other users may only view reports.
3. The system may be improved by adding automatic backup and recovery features. This would help prevent data loss in case of accidental deletion, database errors, or device problems.
4. Future versions of the system should support data export features, such as exporting reports to PDF or Excel format. This would make it easier for users to print, share, and submit academic summaries.
5. The system may include semester-based and school-year-based records so that users can organize student grades according to academic periods. This would make the system more useful for long-term academic monitoring.
6. Future developers may add GPA ranking, performance remarks, and academic standing indicators to provide a clearer summary of student performance.
7. The system may be enhanced by allowing users to customize subjects and units instead of relying only on predefined subjects encoded by the developers. This would make the system more flexible for different courses and grading systems.
8. The graphical user interface should be improved further by making it more modern, responsive, and visually appealing. A cleaner and more organized layout can improve user experience and make the system easier to navigate.
9. The system may be upgraded to support multi-user access or cloud-based storage in the future. This would allow users to access records from different devices while maintaining proper data security.
10. Future researchers and developers are encouraged to conduct user evaluation or usability testing with actual teachers, students, or administrators. Their feedback can help determine the strengths and weaknesses of the system and guide further improvements.

In conclusion, while the GPA Tracker System already performs its intended functions, it can still be improved through additional security, backup, reporting, customization, and accessibility features. These enhancements would make the system more efficient, reliable, and suitable for broader academic use.

ACKNOWLEDGEMENT

The researchers would like to express their heartfelt appreciation to **Engr. Meshelle N. Fabro, PCpE**, for her dedicated mentorship, professional guidance, and expert supervision throughout the completion of this study. Her valuable insights, constructive suggestions, and continuous encouragement significantly contributed to the successful development and improvement of this research.

The researchers also extend their sincere gratitude to the **Eulogio “Amang” Rodriguez Institute of Science and Technology (EARIST)**, particularly the **Department of Computer Engineering**, together with its faculty members, for providing the academic knowledge, technical resources, and institutional support necessary for the accomplishment of this project. The institution’s commitment to excellence in education, innovation, and research played an essential role in the success of this study.

Special thanks are also given to the families and friends of the researchers for their unwavering support, patience, understanding, and motivation. Their encouragement served as a source of inspiration and strength throughout the challenges encountered during the research process.

Above all, the researchers offer their deepest gratitude to **Almighty God** for the wisdom, guidance, strength, and perseverance bestowed upon them throughout this endeavor. This work is humbly dedicated to His glory, acknowledging that every achievement was made possible through His grace and blessings.

REFERENCES

1. Buditjahjanto, I. G. P. A. (2022). Analyzing factors of GUI simulation as learning media toward students' learning outcomes. *Journal of Technology and Science Education*, 12(1), 83–95. doi: 10.3926/jotse.1317
2. Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly*, 13(3), 319–340. doi: 10.2307/249008
3. DeLone, W. H., & McLean, E. R. (2003). The DeLone and McLean model of information systems success: A ten-year update. *Journal of Management Information Systems*, 19(4), 9–30. doi: 10.1080/07421222.2003.11045748
4. Gaffney, K. P., Prammer, M., Brasfield, L. C., Hipp, D. R., Kennedy, D. R., & Patel, J. M. (2022). SQLite: Past, present, and future. *Proceedings of the VLDB Endowment*, 15(12), 3535–3547. doi: 10.14778/3554821.3554842
5. Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., ... SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, 17(3), 261–272. doi: 10.1038/s41592-019-0686.

ABOUT THE AUTHORS

Jazmine Claire M. Capati is a Computer Engineering student at Eulogio “Amang” Rodriguez Institute of Science and Technology. She is dedicated to finishing her studies and achieving her goals in the field of technology. She is hardworking, responsible, and determined to improve her knowledge and skills in programming and computer systems. Despite challenges in her studies, she continues to stay focused and motivated in reaching her ambitions. She also enjoys learning new things and exploring ideas that can help her grow academically and personally as a future computer engineer.

Mark C. Cea is a Computer Engineering student at Eulogio “Amang” Rodriguez Institute of Science and Technology. He is a hardworking student and is very dedicated to finishing his academic responsibilities. He is determined to enhance his capabilities as a computer engineering student, improve his hands-on skills, and become proficient in entering the computer engineering field. He also enjoys learning how certain devices work and dismantles them to understand more about technology and how they function. He plans to become successful in the computer engineering field.

John Wilson M. Esquilona is currently pursuing a degree in Computer Engineering at Eulogio “Amang” Rodriguez Institute of Science and Technology. He is passionate about technology and continuously works toward achieving his academic and professional goals. John is a dedicated and responsible student who is eager to enhance his technical knowledge and explore different fields in computing and engineering. He is committed to overcoming challenges in his studies while improving his analytical thinking, problem-solving, and programming skills. Through continuous learning, determination, and self-improvement, he aspires to become a skilled computer engineer in the future.

Christopher Dhaniel P. Maligro is a Computer Engineering student at Eulogio “Amang” Rodriguez Institute of Science and Technology. He is skilled in programming languages such as C++ and Python, which help him strengthen his abilities in coding, problem-solving, and system development. He is known for being responsible, hardworking, and cooperative, especially when working with others in academic activities and group tasks. As a student, he is committed to enhancing his technical knowledge and continuously exploring new skills that will help him succeed in the field of computer engineering.

Mark Daniel F. Maranan is a Computer Engineering student at Eulogio “Amang” Rodriguez Institute of Science and Technology. He is committed to completing his degree and building a successful career in the technology industry. Mark is known for being diligent, dependable, and eager to enhance his understanding of computer systems, programming, and emerging technologies. He approaches academic challenges with perseverance and determination, continuously striving to improve his skills. In addition to his studies, he values personal growth and actively seeks opportunities to expand his knowledge and gain practical experience in his chosen field.

Gabriel Andrei P. Marinay is currently pursuing a degree in Computer Engineering at Eulogio “Amang” Rodriguez Institute of Science and Technology. He is passionate about technology and works consistently toward achieving his educational and professional objectives. Gabriel is a motivated and responsible student who enjoys developing his technical abilities and learning about different aspects of computing. He remains dedicated to overcoming obstacles in his academic journey and continues to strengthen his problem-solving and analytical skills. Through continuous learning and self-improvement, he aims to become a competent and successful computer engineer in the future.

Engr. Meshelle N. Fabro is a licensed Professional Computer Engineer and educator at the Eulogio “Amang” Rodriguez Institute of Science and Technology (EARIST). She is currently pursuing a Master of Science in Computer

Engineering at Bulacan State University and has professional industry experience as a Technical Engineer at HewlettPackard (HP) and an IT Engineer at IBM. She is also a TESDA CSS NC II and Trainers Methodology Level I (TM1) holder, demonstrating her commitment to technical excellence and competency-based education. Passionate about teaching and innovation, she is dedicated to guiding Computer Engineering students in becoming industry-ready, competent, and highly skilled individuals equipped for the evolving demands of the technology industry.