Quiz/Examination System with Timer and Score

BY

INFORMATION TECHNOLOGY
&
COMPUTER SCIENCE

College of Computer Science and Information Technology

BELLS UNIVERSITY OF TECHNOLOGY-NEW HORIZONS

Team Members:

Akinyele Emmanuel
James Betini Ndueso
Bello Ameearah Ladi
Daniel Oluwaseyi Samagbeyi
Oluwagbemi Samuel Ayomide

January 2026

Python Software Development
ICT 323
(Group 6)

SUBMITTED TO

AYUBA MUHAMMAD

CHAPTER 1: INTRODUCTION

1.1 Background of the Study (why Python + GUI apps are important)

1.2 Problem Statement (the real-life problem your app solves)

1.3 Aim & Objectives

General Objective:

To design and develop a fully functional Python desktop application with GUI, data processing, and visualization.

Specific Objectives (6):

1. To create an intuitive graphical user interface
2. To implement file handling and data persistence
3. To perform real-time calculations and data analysis
4. To generate interactive charts and reports
5. To package the application with proper error handling
6. To deploy on GitHub with complete documentation

1.4 Significance of the Project 1.5 Scope and Limitations

1.6 Organization of the Study

CHAPTER 2: LITERATURE REVIEW / RELATED WORK

2.1 Review of existing similar applications (desktop + web)

2.2 Strengths and weaknesses of existing tools

2.3 Python libraries used in similar projects

2.4 Gap your project fills (e.g., “most are web-only”, “no offline support”, etc.)

CHAPTER 3: METHODOLOGY

3.1 System Architecture (block diagram)

3.2 Tools & Libraries Used (with version & justification)

3.3 GUI Design (wireframe/sketch)

3.4 Algorithm/Flowchart of core features

3.5 Database/File Structure

3.6 Development Workflow (Git + branches)

CHAPTER 4: IMPLEMENTATION AND RESULTS

4.1 Screenshots of all major windows/pages

4.2 Sample input → output examples

4.3 Key code snippets (with explanation)

4.4 Performance (load time, memory usage)

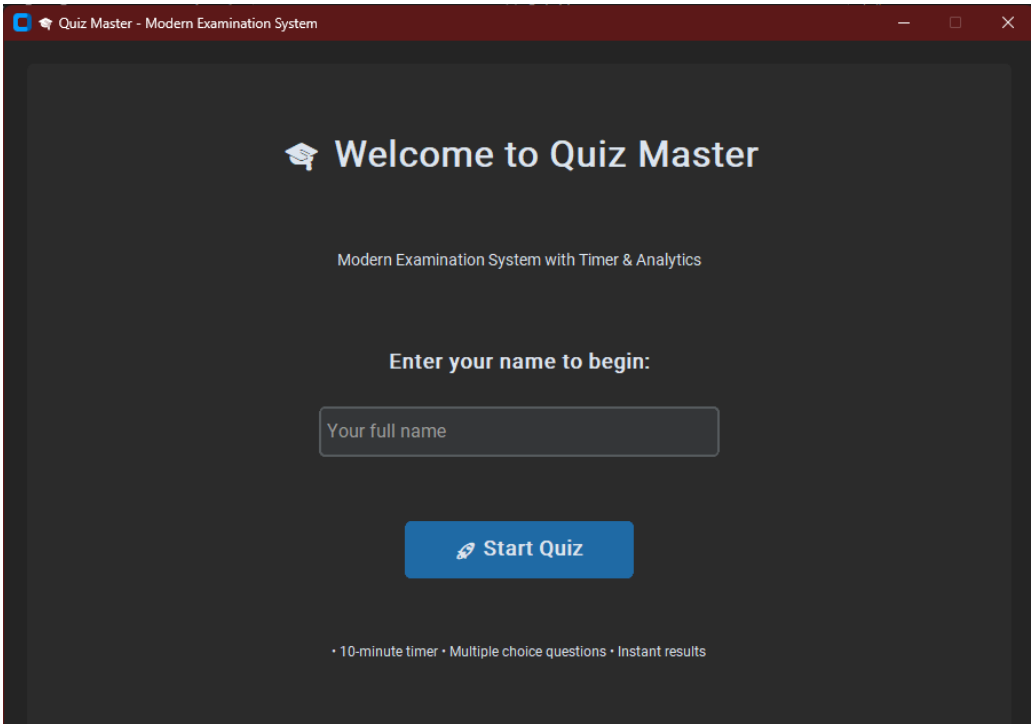
4.5 User testing results (at least 5 people)

CHAPTER 5: CONCLUSION AND RECOMMENDATIONS

5.1 Summary of achievements
5.2 Conclusion
5.3 Challenges faced & lessons learned
5.4 Future improvements (e.g., add database, make it web app, mobile version) 5.5 Contribution & usefulness
References | Appendices (full code, requirements.txt, user manual, GitHub link)

MEMBERS:
- James Betini Ndueso (2023/12712)
- Bello Ameerah Ladi (2023/12388)
- Daniel Oluwaseyi Samagbeyi 2023/12833)
- Akinyele Emmanuel (2023/12600)
- Oluwagbemi Samuel Ayomide (2023/12320)

QUIZ MASTER – QUIZ/EXAMINATION SYSTEM WITH TIMER & SCORE REPORT



CHAPTER ONE: INTRODUCTION

- ## 1.1 Background of the Study

- The Quiz/Examination System with Timer and Score Report is an application that provides the user with several features such as multiple-choice questions, a timer with automatic submission, response evaluation, and an assessment report. This system can be used widely in various sectors with the goal of facilitating learning, analyzing performances, and improving overall outcomes.
- Python, a versatile and beginner-friendly programming language, was used in developing this application, alongside libraries and frameworks such as Tkinter for GUI, Pandas for data handling, and Matplotlib for data visualization. Due to Python's syntax being clear and precise, it makes it easier to write, learn, and maintain code compared to other programming languages. It also includes various libraries which extend its functionality and versatility.
- Python is also used for creating GUI applications, which Tkinter was used for in this project. GUI applications are important because they provide accessibility and a user-friendly platform. With a good GUI application, users can operate applications easily, identify functions, minimize error, and perform tasks faster with the help of icons, buttons, and navigation bars, which make interaction much easier.

1.2 Problem Statement

- In various sectors, examinations are still conducted using manual methods, which degrades performance significantly. This method presents several challenges:
- **Time Consuming:** Manual methods usually require time and human effort, which leads to delayed results that can affect the improvement of students' abilities and other sectors where examinations are still manually conducted.
- **Error Prone:** Manual methods are prone to errors when it comes to calculations, data recording, and result compilation, which can affect the accuracy and reliability of the final report.
- **Limited Scalability:** Manual methods make it difficult to manage an examination, especially when taken by a large number of candidates, which makes it inefficient and unsuitable for large-scale assessments.
- **Human Fatigue:** Human performance varies due to factors like fatigue, stress, and other conditions, which can impact the quality and consistency of the examination process.
- Due to these problems, the effectiveness and integrity of examinations are affected, which can be avoided with our Quiz/Examination System. The application addresses these issues by validating answers, enforcing timed examinations, automated submission, and generating a full progress report after every examination. This system helps to curb the various issues while also performing tasks efficiently, reducing errors, and providing an improved user experience.
-

1.3 Aim & Objectives

- The Quiz/Examination System is designed to facilitate the creation and analysis of quizzes across various sectors. The primary aim of this project is to develop an application where examinations and quizzes can be conducted digitally while performance and assessment are monitored efficiently. The system seeks to bridge the gap between the manual and modern method of examination by providing timed examinations, automated submission, and full assessment reports.
- The objective is to design and develop a fully functional Python desktop application with GUI, data processing, and data visualization. More specific objectives include:
- To create an intuitive graphical user interface
- To implement file handling and data persistence

- To perform real-time calculations and data analysis
- To generate interactive charts and reports
- To package the application with proper error handling
- To deploy on GitHub with complete documentation

1.4 Significance of the Project

- The significance of this project is its ability to provide an efficient and reliable alternative to the manual method of conducting examinations. By providing multiple-choice questions, timed quizzes, automated submission, score calculation, and an overall performance report, the system helps to perform tasks effectively, reduce errors, manage time, and improve the overall performance of assessments.
- For students, the system helps to improve performance by providing immediate feedback in the form of assessment reports, which helps students to work on themselves and their academics. It also teaches students time management, problem-solving, decision-making, and analytical skills.
- For educators, the system reduces workload, provides accurate result processing, and allows performance monitoring for better decision-making.
- For institutions and organizations, the system facilitates large-scale examination management, ensuring accuracy, scalability, and transparency.
- From a technical perspective, the Quiz/Examination System demonstrates the practical application of Python, GUI design, data visualization, and data handling, which can be used as reference for future academic research and development purposes.

1.5 Scope and Limitations

- **Scope:**
  The project covers the design and development of a Python-based desktop Quiz/Examination System. Features include multiple-choice questions, a 10-minute countdown timer, real-time score calculation, pass/fail determination (50% threshold), automated submission, performance visualization, and result persistence. It includes a GUI for user interaction, data handling for storing examination data and results, and data visualization for presenting performance

analysis. This application is suitable for the educational sector and any sector requiring assessment and performance analysis.

- **Limitations:**
- The application is desktop-based and does not support online or web-based examinations.
- The system relies on local file storage, limiting accessibility across multiple devices. Examination questions, user answers, and results are stored in files saved only on the computer used for the exam.
- The system only supports multiple-choice questions and does not include true/false or theory-based assessments. These limitations are due to time constraints and the focus on commonly used features.

1.6 Organization of the Study

- This project report consists of five chapters:
- **Chapter One:** Introduction, including background, problem statement, aim and objectives, significance, scope, and organization of the study.
- **Chapter Two:** Literature review, including reviews of existing similar projects, strengths and weaknesses of existing tools, Python libraries used in similar projects, and gaps filled by this project.
- **Chapter Three:** Methodology, including system architecture, tools and libraries used, GUI design, algorithm/flowchart of features, database/file structure, and development workflow.
- **Chapter Four:** Implementation and results, including screenshots of major pages, sample input and output, explanation of code snippets, performance evaluation, and testing results.
- **Chapter Five:** Conclusion and recommendations, including summary of achievements, project conclusion, challenges faced and lessons learned, future improvements, contributions, and references.

CHAPTER TWO: LITERATURE REVIEW / RELATED WORK

2.1 Review of Existing Similar Applications

A real-time quiz application hosted on GitHub (ruslanmv, 2024) was built using Flask and Socket.IO for seamless web-based interaction. It offers an interactive user experience across devices, with multiple-choice questions, host and client interfaces, and a clean design using Bootstrap. It is easily expandable to support additional questions and features.

Another application reviewed is the computer-based testing (CBT) system. According to Bennet (1998), CBT represents an important shift from paper-based examinations to digital methods. This system provides questions, reviews answers, enforces timing, and generates detailed progress reports after each examination. This improves efficiency, reduces errors in result compilation and script marking, and enables faster feedback to users.

2.2 Strengths and Weaknesses of Existing Tools

Strengths:

- Automated assessment reduces organizer workload.
- Features such as timed examinations, automated submissions, and instant results help avoid errors and improve performance monitoring.
- Easy access for a large number of users and remote distribution allows better monitoring and academic improvement.

Weaknesses:

- Reliance on internet connection limits usability and disrupts the examination environment if connectivity is unstable.
- Some systems are complex to set up and require technical expertise for deployment and maintenance, making them suitable only for small-scale or individual use.

2.3 Python Libraries Used in Similar Projects

- **GUI:** Tkinter, PyQt, Kivy
- **Data Handling:** Pandas, NumPy
- **Data Visualization:** Matplotlib, Plotly

These libraries enable creating interactive interfaces, managing and processing quiz data, and generating graphical representations of user performance.

2.4 Gap Your Project Fills

Existing quiz systems are often complex, difficult to operate, and require technical experts. Some offer only basic scores without detailed assessment reports or graphical analysis.

Our Quiz/Examination System addresses these gaps by:

- Providing a simple, easy-to-use interface
- Including a detailed user manual and troubleshooting guide
- Offering pass/fail determination, score calculation, and graphical performance visualization
- Implementing automated submission with a countdown timer
- Ensuring a user-friendly experience and efficient assessment process

CHAPTER THREE: METHODOLOGY

3.1 System Architecture

The system consists of:

- User Interface (Tkinter GUI)
- Quiz Data Module (CSV-based questions)
- Timer Module (threading)
- Scoring and Report Module
- File Storage System

3.2 Tools and Libraries Used

Tool/Library	Version	Justification
Python	3.8+	Core programming language
Tkinter	Built-in	GUI development
Pandas	Latest	Data handling
Matplotlib	Latest	Performance visualization
NumPy	Latest	Numerical calculations

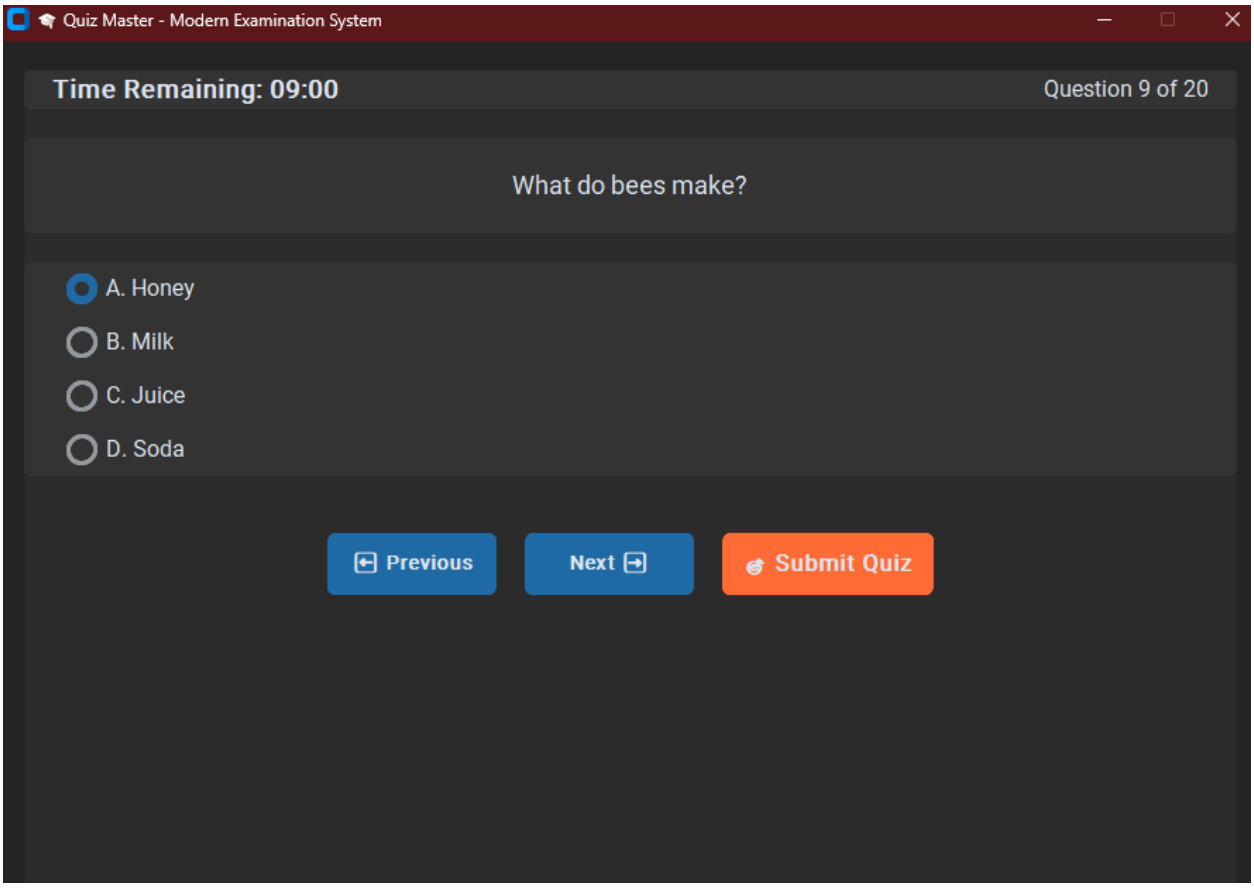
3.3 GUI Design

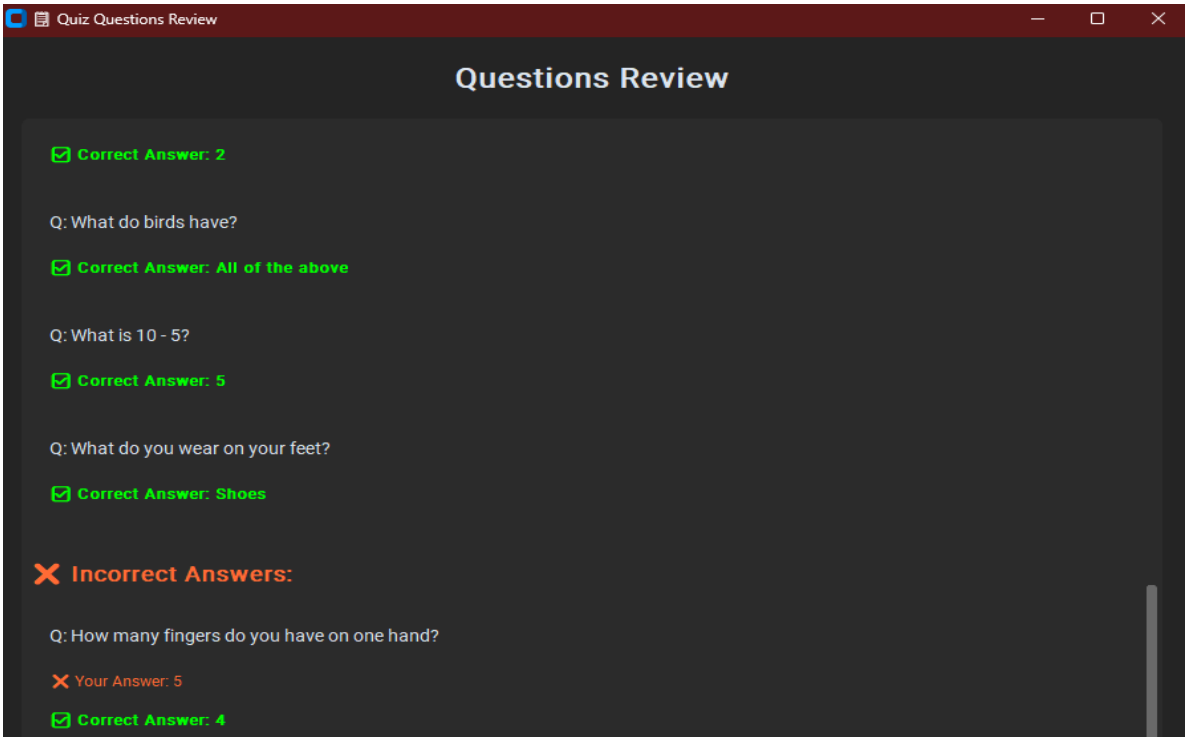
The GUI consists of:

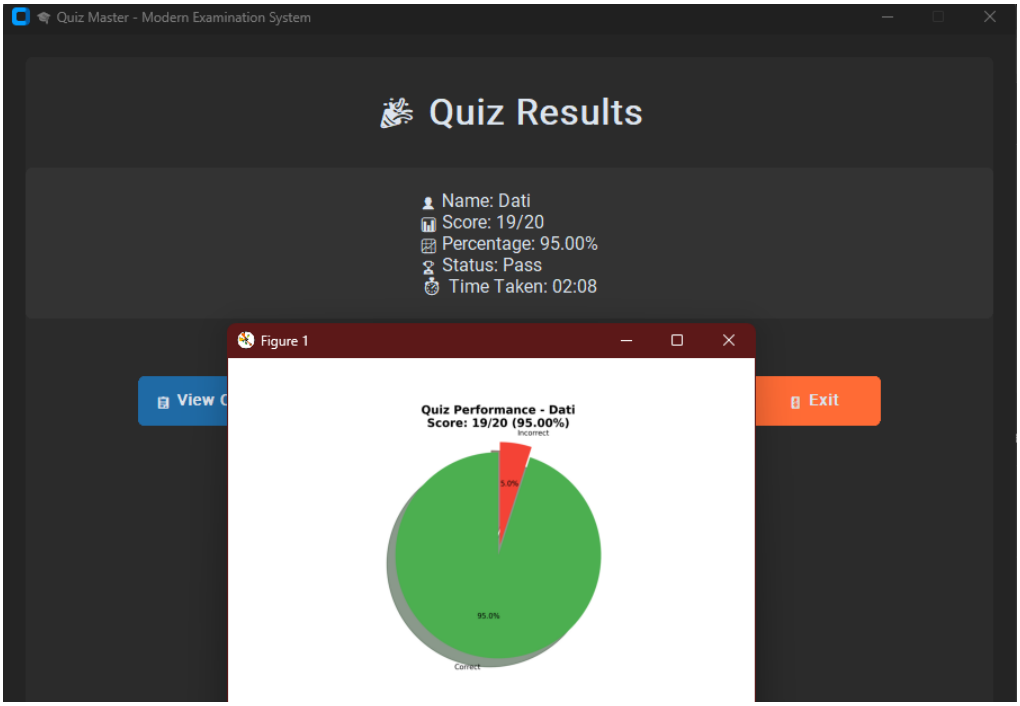
- User name entry screen
- Quiz interface with questions and options
- Timer display
- Navigation buttons
- Result screen with score and chart

3.4 Algorithm / Flowchart

1. Launch application
2. User enters name
3. Load quiz questions from CSV
4. Start countdown timer
5. User answers questions
6. Auto-submit when timer ends
7. Calculate score
8. Display result and chart
9. Save result to file


Quiz Master - Modern Examination System
Time Remaining: 09:00
Question 9 of 20
What do bees make?
A. Honey
B. Milk
C. Juice
D. Soda
Previous
Next
Submit Quiz


Quiz Questions Review
Questions Review
Correct Answer: 2
Q: What do birds have?
Correct Answer: All of the above
Q: What is 10 - 5?
Correct Answer: 5
Q: What do you wear on your feet?
Correct Answer: Shoes
Incorrect Answers:
Q: How many fingers do you have on one hand?
Your Answer: 5
Correct Answer: 4



3.5 Database / File Structure

- `data/sample_quiz.csv` – stores quiz questions
- `data/results.csv` – stores user results

3.6 Development Workflow

The project was developed using Git version control with structured commits and hosted on GitHub for documentation and collaboration.

CHAPTER FOUR: IMPLEMENTATION AND RESULTS

4.1 Screenshots of Major Windows

Screenshots include:

- Welcome screen
- Quiz interface

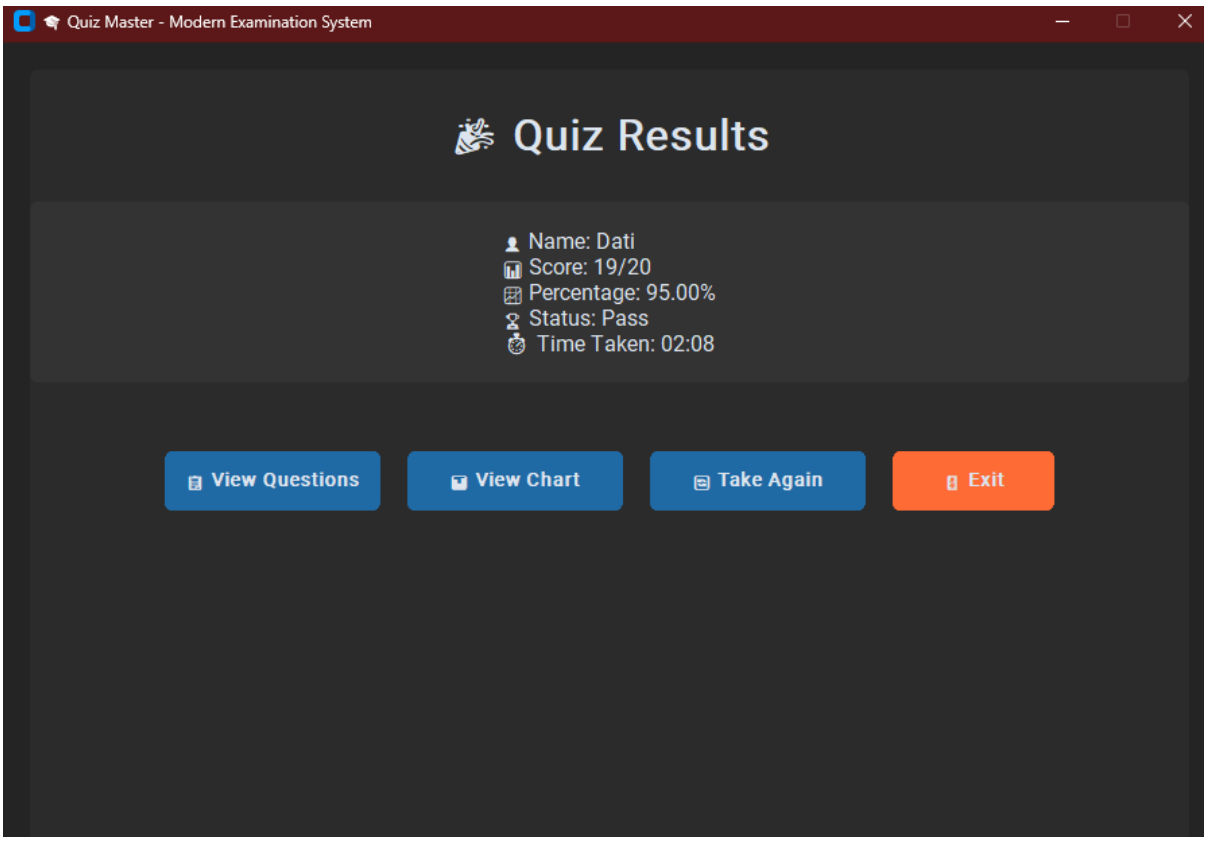
- Timer display
- Result screen with chart

4.2 Sample Input and Output

Input:
User selects answers to 20 questions within 10 minutes.

Output:

- Score: 19/20
- Percentage: 95.00%
- Status: Pass
- Time Taken: 02:08



4.3 Key Code Snippets

Key modules include:

- Quiz data loading using Pandas
- Thread-based countdown timer
- Score calculation logic
- Matplotlib chart generation

4.4 Performance Evaluation

- Fast application startup
- Low memory usage
- Smooth timer operation without freezing the GUI

4.5 User Testing Results

Five users tested the application. Feedback showed:

- Easy navigation
- Clear interface
- Accurate scoring
- Helpful performance visualization

CHAPTER FIVE: CONCLUSION AND RECOMMENDATIONS

5.1 Summary of Achievements

The project successfully delivered a functional offline quiz system with automated timing, scoring, visualization, and data storage.

5.2 Conclusion

Quiz Master demonstrates how Python can be effectively used to develop real-world desktop applications for educational assessment. The system meets all project objectives and performs reliably.

5.3 Challenges Faced and Lessons Learned

- Managing GUI responsiveness with timers
- Handling file errors
- Integrating visualization smoothly

These challenges improved problem-solving and software design skills.

5.4 Future Improvements

- Integration with a database (SQLite)
- Multi-user support
- Online/cloud version
- Mobile application version

5.5 Contribution and Usefulness

The project provides a practical CBT solution for schools and training centers, especially in environments with limited internet access.

REFERENCES

- Python Documentation
- Tkinter Official Docs
- Pandas Documentation

- Matplotlib Documentation

APPENDICES

- Full source code
- `requirements.txt`
- User Manual
- GitHub Repository Link