International Journal of Trend in Scientific Research and Development (IJTSRD)
Special Issue on Innovations in Computer Science and Applications
Available Online: www.ijtsrd.com e-ISSN: 2456 – 6470

AI Error Decoder a Python Based Application for Understanding Programming Errors

Anjali Ghagare

Department of Science and Technology,
G. H. Raisoni Skill Tech University, Nagpur, India

Abstract

Programming is an important part of computer science, but many beginners face difficulty in understanding error messages generated by compilers or interpreters. These messages are often technical and confusing, which makes debugging difficult for new learners.

This project presents an application called AI Error Decoder, which helps users understand programming errors in a simple and clear way. Users can paste error messages into the system, and the application analyzes them to provide easy explanations along with possible solutions.

The system is developed using Python, with PyQt5 for the graphical user interface and SQLite for storing error history. The main aim of this project is to make debugging easier for beginners and improve their learning experience by simplifying complex error messages.

KEYWORDS: *Artificial Intelligence, Programming Errors, Error Detection, Automated Debugging, Python Application, Machine Learning, Natural Language Processing, Code Analysis, Syntax and Runtime Errors, Intelligent Error Prediction.*

1. Introduction

In programming, errors are very common, especially for beginners. Whenever a mistake occurs in the code, the compiler or interpreter generates an error message. While experienced programmers can understand these messages quickly, beginners often find them difficult and confusing.

Many students spend a lot of time trying to understand what an error actually means before fixing it. This can slow down the learning process and sometimes reduce confidence.

To solve this problem, the **AI Error Decoder** system is developed. This application converts complex error messages into simple language so that users can easily understand what went wrong in their code.

The system is built using Python, with PyQt5 for designing the interface and SQLite for storing past error records. It is designed in a simple and user-friendly way so that even beginners can use it easily.

1.1 Problem Statement

Programming is an essential skill in today's technology-driven world, but beginners often struggle while learning it. One of the major challenges they face is understanding error messages generated by programming languages.

These error messages are usually written in technical language and assume prior knowledge, which beginners may not have. As a result, students often get confused when they see errors like SyntaxError, TypeError, or IndexError.

Even though the error message points to a problem, it does not always clearly explain what exactly went wrong or how to fix it. Because of this, learners spend a lot of time searching online or asking others for help.

This process can be frustrating and can slow down learning. Sometimes, students fix errors without fully understanding them, which prevents them from improving their debugging skills.

Therefore, there is a need for a system that can simplify these error messages and provide clear explanations so that beginners can learn more effectively.

2. Objective of project

The main objective of this project is to develop a simple and easy-to-use application that helps beginners understand programming errors.

The system aims to convert technical error messages into simple explanations so that users can quickly understand the issue in their code.

Another goal is to support beginner programmers by reducing confusion and helping them gain confidence while coding.

The project also focuses on providing suggestions or guidance to fix errors, so users not only solve the problem but also learn from it.

Additionally, the system stores previously decoded errors so users can review them later, which improves learning over time.

3. Proposed System

The proposed system, AI Error Decoder, is designed as a desktop-based application that helps programmers, especially beginners, understand programming error messages in a simple and clear way.

In many programming environments, error messages are displayed in a technical format which is difficult for beginners to understand. This system solves that problem by converting those complex messages into easy explanations.

The working of the system is simple. The user enters or pastes an error message into the application. After receiving the input, the system processes the message and checks for specific keywords related to common programming errors. Based on these keywords, the system identifies the type of error.

Once the error type is identified, the system provides a simplified explanation along with basic suggestions to fix the problem. For example, if the system detects a SyntaxError, it explains that there is a mistake in the code structure. If an

IndexError is detected, it explains that the program is trying to access an element that does not exist.

The system also includes a history feature, where all previously decoded errors are stored in a database. This helps users review past mistakes and learn from them.

Overall, the system is designed to be simple, efficient, and helpful for beginners

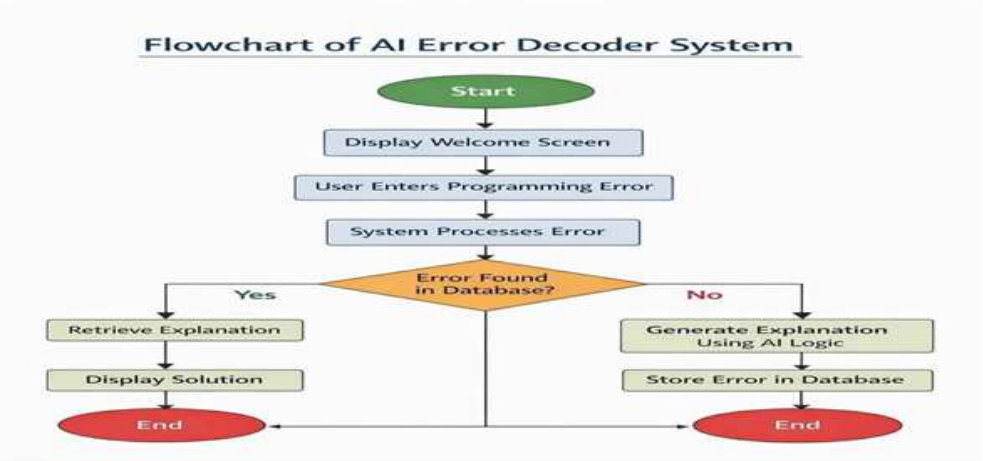


Fig : flowchart

4. System Architecture

The system architecture of the AI Error Decoder is divided into multiple layers to make the system organized and easy to manage.

1. User Layer

This is the top layer where the user interacts with the application. The user enters error messages and views the explanations provided by the system.

2. Presentation Layer (GUI)

This layer is responsible for displaying the interface. It is developed using PyQt5 and includes different screens such as:

- Welcome screen
- Error input screen
- History screen

The interface is designed in a simple way so that users can easily understand and use it.

3. Application Logic Layer

This is the core part of the system. It processes the input error message and identifies keywords related to common programming errors. Based on this analysis, it generates a simplified explanation.

4. Database Layer

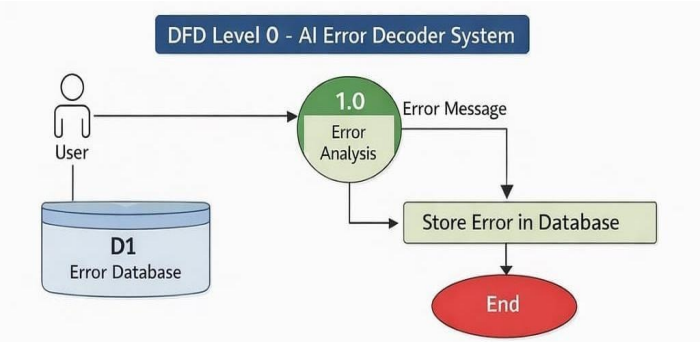
This layer handles data storage. SQLite is used to store:

- Error messages
- Explanations
- Date and time

5. Data Storage

The stored data helps users revisit previously decoded errors, which improves their learning and understanding.

4.1. Data Flow Diagram- level 0



4.2. Data Flow Diagram- level 1

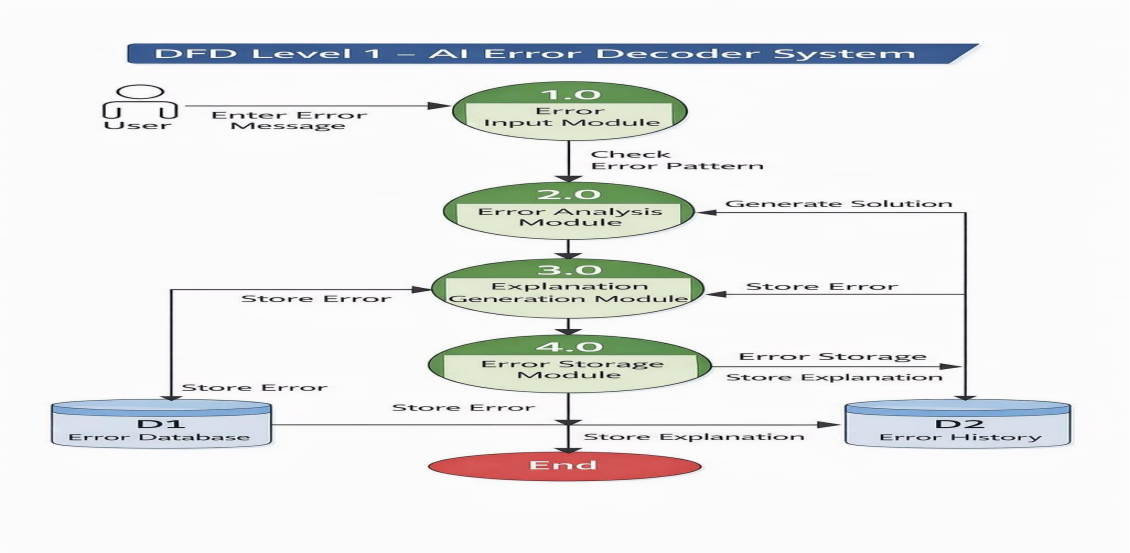


Figure 1

5. Research Methodology

The development of this project followed a step-by-step approach.

First, requirement analysis was done to understand the problems faced by beginner programmers. It was observed that many students struggle to understand technical error messages.

After this, the system design phase was carried out. The application was divided into different modules such as the decoder module, history module, and database module.

Next, the development phase started. Python was used as the main programming language because it is simple and easy to use. PyQt5 was used to design the graphical interface, and SQLite was used for storing data.

The system uses a rule-based approach, where it matches keywords in the error message to identify the error type.

Finally, testing was performed using different types of error messages to ensure that the system works correctly.

5.1. Algorithm

Step 1: Start the application
Step 2: User enters the programming error message
Step 3: System receives the input message
Step 4: System check keywords in the error message
Step 5: Identify the type of error (SyantaxError, IndexError, TypeError)
Step 6: Retrieve explanation from predefined rules
Step 7: Display a simplified explanation to the user
Step 8: Save the error message and explanation in database
Step 9:Stop

6. Implementation Details

The AI Error Decoder was implemented as a desktop application using Python.

Python was chosen because it is beginner-friendly and widely used. The graphical user interface was created using PyQt5, which provides tools like buttons, labels, and text boxes.

The backend logic is responsible for analyzing error messages. It checks for keywords and matches them with predefined rules to identify the error type.

SQLite is used as the database to store error history. It records the error message, explanation, and time of analysis.

The application was developed using Visual Studio Code and tested with multiple error inputs to ensure proper functionality.

7. Testing and Results

After development, testing was done to check the performance of the system.

Different types of errors such as SyntaxError, TypeError, and IndexError were entered into the system. The application successfully identified these errors and provided correct explanations.

The history feature was also tested, and it was found that all decoded errors were properly stored and displayed.

The user interface was smooth and easy to use. The system responded quickly and did not crash even when incorrect inputs were given.

Overall, the results show that the system works efficiently and is helpful for beginners.

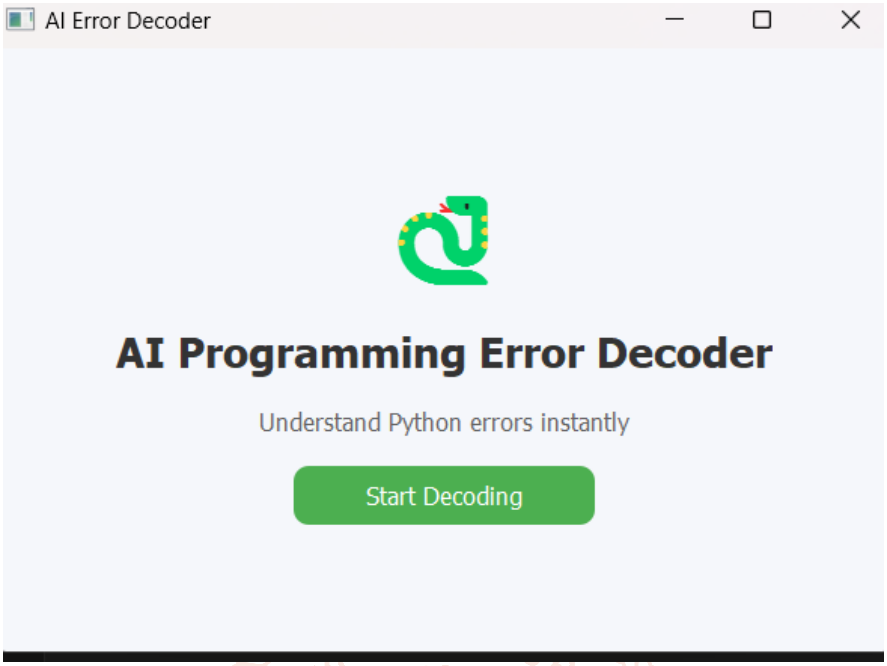


Fig: Welcome Screen

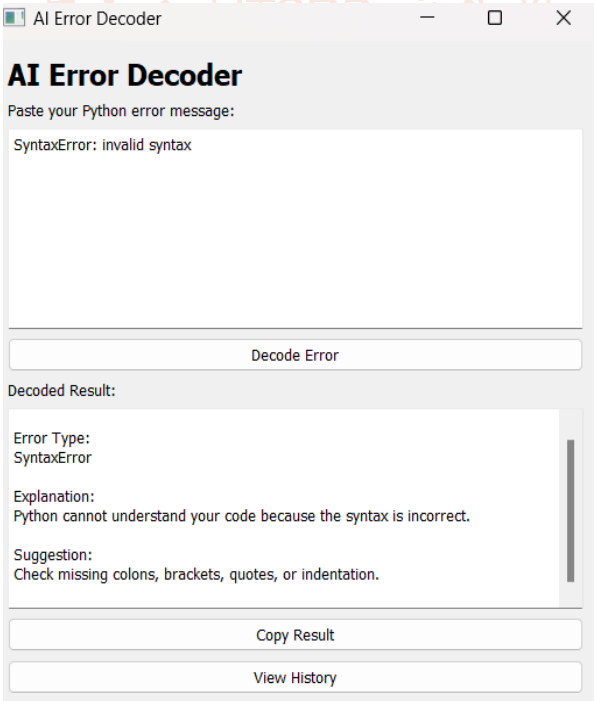


Fig: Decoder Screen

8. Conclusion

The AI Error Decoder project was developed to help programmers, especially beginners, understand programming error messages more easily. Many programming errors are difficult to understand because they are written in technical language. This system simplifies those messages and provides clear explanations so that users can identify and fix their mistakes more effectively.

The application was successfully implemented using Python for the backend, PyQt5 for the graphical user interface, and SQLite for storing error history. The system allows users to enter an error message, receive a simplified explanation, and view previously decoded errors through the history feature.

The testing results show that the application works properly and is able to identify common programming errors using a rule-based approach. The interface is simple and user-friendly, making the system suitable for beginners who are learning programming.

Overall, the AI Error Decoder provides a practical and helpful tool for improving the debugging process and supporting the learning experience of new programmers.

9. Future Scope

Although the system works well, there are several improvements that can be made in the future.

One improvement is adding support for more programming languages such as Java, C++, and JavaScript.

Another enhancement could be using machine learning or artificial intelligence to provide more accurate explanations.

The system can also include automatic code correction suggestions to help users fix errors quickly.

A web-based version of the application can also be developed so users can access it online.

Additionally, the user interface can be improved by adding features like dark mode and better design.

The application can also be improved by adding code correction suggestions. Instead of only explaining the error, the system could provide possible solutions or corrected code examples to help users fix their mistakes more quickly.

In addition, the system could include online resources and learning links. For example, when an error is detected, the application could display helpful documentation, tutorials, or reference materials related to that error. This would make the tool more useful for learning purposes.

Another future enhancement could be the development of a web-based or cloud version of the system. This would allow users to access the error decoder from any device through a browser without needing to install the application.

Finally, the user interface can be further improved by adding more interactive features such as better visualization, dark mode, and additional navigation options. These improvements would make the system more attractive and user-friendly.

Overall, with these enhancements, the AI Error Decoder system has the potential to become a more powerful and helpful tool for programmers and students who want to understand and solve programming errors more effectively.

10. References

[1] Van Rossum, G., & Drake, F. L. (2009). Python 3 Reference Manual. CreateSpace.

[2] Python Software Foundation. (2024). Python Documentation. Available at: https://docs.python.org

[3] Riverbank Computing. (2024). PyQt5 Documentation. Available https://www.riverbankcomputing.com/software/pyqt

[4] SQLite Consortium. (2024). SQLite Documentation. Available at: https://www.sqlite.org/docs.html

[5] Downey, A. B. (2015). Think Python: How to Think Like a Computer Scientist. O'Reilly Media.

[6] Lutz, M. (2013). Learning Python. O'Reilly Media.