

# A Minor Mid-Term Defence On

## **Python Bug Finder**

Submitted in Partial Fulfilment of the Requirements for the degree of  
Bachelor of Engineering in Information Technology  
Under Pokhara University

Submitted by:

**APSARA ARYAL, 201743**  
**BIKASH BANJADE, 201705**  
**DHIRAJ YADAV, 201708**  
**SANDESH ADHIKARI, 201730**

Date:

15 JUNE 2024



Department of Software Engineering

## **NEPAL COLLEGE OF INFORMATION TECHNOLOGY**

---

Balkumari, Lalitpur, Nepal

## **Abstract**

The “Bugs Finder system” is a web application designed to assist users in identifying and resolving bugs in their code. With the increasing complexity of software development, debugging remains a significant challenge for developers. This platform aims to empower users by providing a user-friendly interface to upload their code, identify errors, and receive guidance on debugging techniques. The ultimate goal is to enhance the efficiency and effectiveness of code debugging processes, benefiting both individual developers and the software development community as a whole.

For the project’s development, we are using HTML, CSS, JavaScript, and mainly React.JS for the frontend and Backend DJANGO framework for handling user requests, managing data flow, and integrating with the debugging model. Database MYSQL for storing user code submissions and debugging reports.

### **Keywords**

*Debugging, Code Analysis, Web application*

## Table of Contents

	<b>Abstract.....</b>	<b>I</b>
<b>1</b>	<b>Introduction.....</b>	<b>1</b>
<b>2</b>	<b>Problem Statement.....</b>	<b>2</b>
<b>3</b>	<b>Project Objectives .....</b>	<b>3</b>
<b>4</b>	<b>Significance of the study .....</b>	<b>4</b>
<b>5</b>	<b>Scope and Limitations .....</b>	<b>5</b>
5.1	Scope .....	5
5.2	Limitations .....	5
<b>6</b>	<b>Literature Review .....</b>	<b>6</b>
6.1	Related Projects.....	6
<b>7</b>	<b>Proposed Methodology .....</b>	<b>7</b>
7.1	Software Process Model.....	7
<b>8</b>	<b>Tools and Technology .....</b>	<b>9</b>
<b>9</b>	<b>Task Done So Far .....</b>	<b>10</b>
<b>10</b>	<b>Results .....</b>	<b>11</b>
10.1	Sign In Page .....	11
10.2	Sign Up page .....	11
10.3	Home Page .....	12
10.4	Features Page.....	12
10.5	Highlights Page .....	13

10.6	Let's Debug page .....	14
10.7	Admin Dashboard page .....	15
<b>11</b>	<b>Task Remaining .....</b>	<b>16</b>
<b>12</b>	<b>Project Task and Time Schedule .....</b>	<b>17</b>
<b>13</b>	<b>List of figures.....</b>	<b>18</b>
13.1	Use Case Diagram.....	18
13.2	Sequence Diagram.....	19
13.3	Database Diagram .....	20
<b>References.....</b>		<b>21</b>

# 1 Introduction

In the rapidly evolving landscape of software development, managing code quality and debugging efficiently are critical challenges that can significantly impact project timelines and overall software performance. Traditional debugging methods can be time-consuming and often depend heavily on the developer's expertise and experience, which can lead to inconsistent results and overlooked errors. To address these challenges, we introduce the concept of the "Bugs Finder" a sophisticated online platform that leverages the power of Large Language Models (LLMs) for software debugging.

Our project, "Bugs Finder", is designed to be an accessible tool on the World Wide Web, allowing developers to upload their code and receive automated insights into potential bugs and their fixes. This system is intended for a diverse range of users, From novice programmers who require guidance to seasoned developers who seek to optimize their debugging process. By integrating generative AI models, specifically LLMs, our system analyses the uploaded code, identifies errors, and suggests actionable solutions efficiently.

The primary issues that our project aims to solve include the reduction of debugging time, providing a learning platform for less experienced developers to improve their coding skills, and offering a robust tool for complex software projects that require rigorous testing and maintenance. With the Bugs Finder, users can track their code's performance, view detailed reports of issues, and receive notifications about potential fixes, making the entire process more manageable and efficient.

This project stands to revolutionize the way developers interact with their code, fostering a culture of precision and efficiency in software development that aligns with modern technological advancements and the demands of current software engineering landscapes.

## 2 Problem Statement

The modern software development, debugging is a critical process that can be time-consuming and often complex, particularly for newer developers or those working on sophisticated systems. The “Bugs finder” aims to streamline this process by utilizing a web-based platform where developers can upload their code to detect and rectify errors with the help of LLms model. Despite the advances in development tools several key challenges continue in the debugging domain:

1. Difficulties in finding bugs and fix them in less time.
2. Enhanced Error Resolution.
3. Accessible Debugging.

### **3 Project Objectives**

The primary goal of the “Bugs finder” project is to develop a comprehensive and user-friendly platform that assists developers in identifying, understanding and resolving coding errors effectively through automated and intelligent diagnostics. The specific objectives are as follows:

1. Develop a system to automatically identify bugs in user-submitted code.
2. Generate comprehensive reports detailing identified bugs and potential fixes.
3. Present users with actionable suggestions for fixing identified bugs, facilitating debugging processes.

## **4 Significance of the study**

The “Bugs Finder” represents a pivotal advancement in the field of software development, addressing critical challenges associated with the debugging process. As software systems become increasingly complex and integral to all aspects of modern life, the efficiency of developing, testing, and maintain software solutions becomes crucial.

The Bugs Finder significantly improves the quality and reliability of software applications, By enabling developers to swiftly identify and correct bugs, the system ensures that software products are more stable and performant upon release. This reduces the occurrence of costly failures and enhances user satisfaction.

Particularly beneficial for new developers, the system provides an education platform that highlights common and uncommon programming errors, offering solutions and best practices.

By reducing the time spent on debugging, the Debug Finder system contributes to more sustainable software development practices. It allows companies to allocate their resources more efficiently and reduce the environmental impact associated with extended development cycles, such as energy consumption and electronic waste.



## **5 Scope and Limitations**

The “Bugs finder” project is designed to revolutionize how developers handle the debugging phase of software development by automating the identification and resolution of code errors. The project leverages the power of Large Language Models (LLMs) to analyse code submitted by users, pinpoint issues, and provide actionable solutions.

### **5.1 Scope**

- Real-time Feedback and Learning.
- User Account Management
- Automated Bug Detection and Resolution.

### **5.2 Limitations**

- Difficult in handle complex code.

## 6 Literature Review

We have studied and researched local as well as global places for attending meetings problem and solutions.

### 6.1 Related Projects

While researching local as well as global places, we found a web-application which are sort of similar with our web application.

Available at : <https://www.onlinegdb.com/>

In preparation for developing the “Bugs Finder” We conducted a comprehensive review of existing tools and platforms that aim to automate and enhance the debugging process in software development. The examination included academic reach, commercially available tools, and emerging technologies in the field of code analysis and error correction. Below are some key findings and how they relate to the “Bugs Finder” project:

#### **Academic Research:**

Various studies published in journals such as IEEE Xplore and ACM Digital Library have highlighted advanced algorithms for static and dynamic code analysis. These studies often focus on specific programming languages and provide insights into pitfalls and challenges in automated debugging systems.

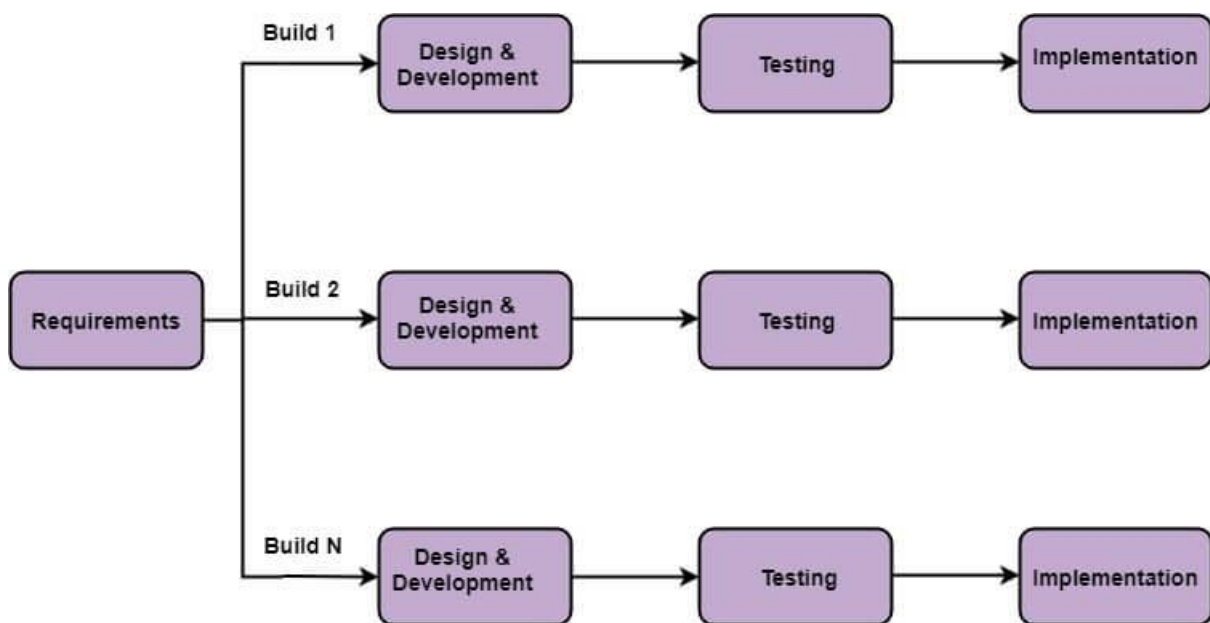
The “Bugs Finder” aims to synthesize the strengths of these existing approaches while addressing their limitations. Our project leverages LLMs for a deeper, more intuitive analysis of code, providing automated debugging that is not only reactive (identifying bugs) but also proactive (suggesting preventative measures and optimizations). Unlike many traditional tools, which often require substantial manual effort for integration and configuration, “Bugs Finder” is designed to be highly user-friendly, with minimal setup and an intuitive interface that accommodates developers of varying skill levels.

## 7 Proposed Methodology

We will use the incremental model in this project as the core features of the project will be developed at first and other features will be added as a series of increments later. The various technologies will be used to develop this project such as HTML, CSS, react JS for the frontend and DJANGO for Backend and MYSQL as the database and VS-Code as IDE and Version Control as git.

### 7.1 Software Process Model

We used the Software Process Model's Incremental Model for our project. The incremental model is a method of developing software in which the requirements are split up into numerous independent software development cycle modules. Each module in this model undergoes the phases of requirements, design, implementation, and testing. Every new version of the module adds a new feature to the previous version. The process continues until the complete system is achieved.



**Fig: Incremental Model**

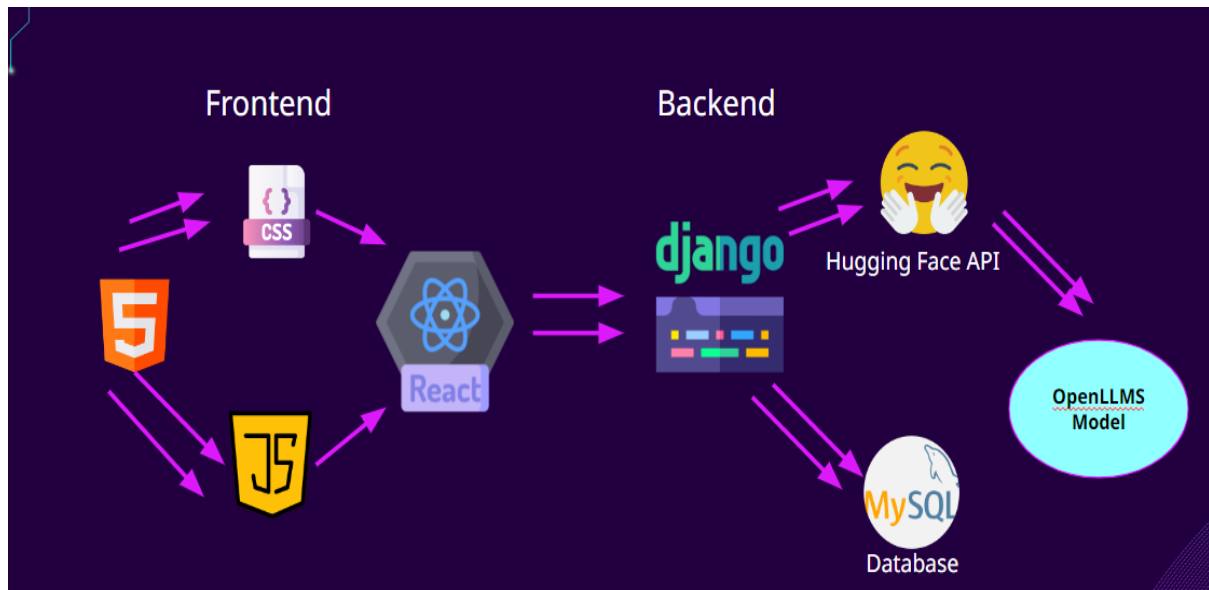
Figure 1: Incremental software model

In the first increment, we'll analyse the project's requirements, defining its scope, objectives, and constraints. For Bugs Finder, this translates to pinpointing the systems essential characteristics, such as user registration and authentication, user profile, uploading file showing file with user interaction. We'll then embark on crucial research, gathering and examining data that aligns with these objectives. This entails investigating existing platforms providing similar services. We'll gather the required data for the application using platforms providing dataset for testing purposes.

The second increment focuses on design, prototyping, and early testing. We'll thoroughly plan the Bugs Finder implementation, designing the system architecture, selecting suitable technologies, and developing streamlined prototypes to demonstrate core functionalities. This is followed by coding the core business logic and integrating them with the data sources. Rigorous tests will be conducted to validate initial functionality and identify potential areas for improvement. While initial deployment isn't the primary focus here, preliminary feedback from the supervisor and the test results are valuable for ensuring the system is on track to meet the defined requirements.

The third increment marks the deployment, monitoring, and continuous improvements stage. The requirement analysis becomes an ongoing process, adapting to evolving business needs and incorporating user feedback. We'll continuously benefit from new research findings allowing for ongoing adaptation. Finally, the evaluation and improvement become long-term endeavours, monitoring the system's performance, gathering user feedback, and continuously improving the system.

## 8 Tools and Technology



### Frontend

- HTML
- CSS
- JavaScript
- ReactJS

### Backend

- Django

### Database

- MySQL

### Debug Engine (Model)

- Gemini-1.5-flash :

Gemini-1.5-flash is a code-specialized version of Gemini Pro that was created by further training Gemini Pro on its code-specific datasets, sampling more data from that same dataset for longer. Essentially, Gemini-1.5-flash features enhanced coding capabilities. It can generate code, and natural language about code, from both code and natural language prompts.

## 9 Task Done So Far

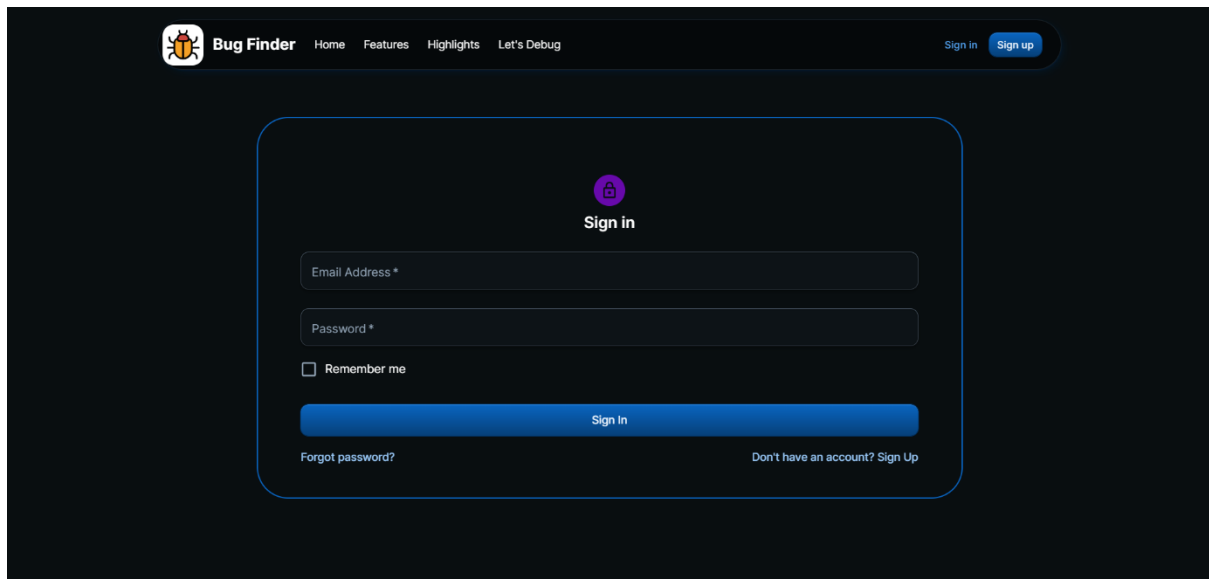
In this python Bugs finder system we are on the way as our plan. The task we done at give interval of time till mid defense are:

- We select our front end and backend coding language.
- We design our login and signup page.
- We design our admin dashboard and home page.
- We design Debug Page and integrate Gemini-1.5-flash model in it.
- API is developed for fetching data from front-end to backend.
- We also design the database to authenticate the user in Bugs Finder System

## 10 Results

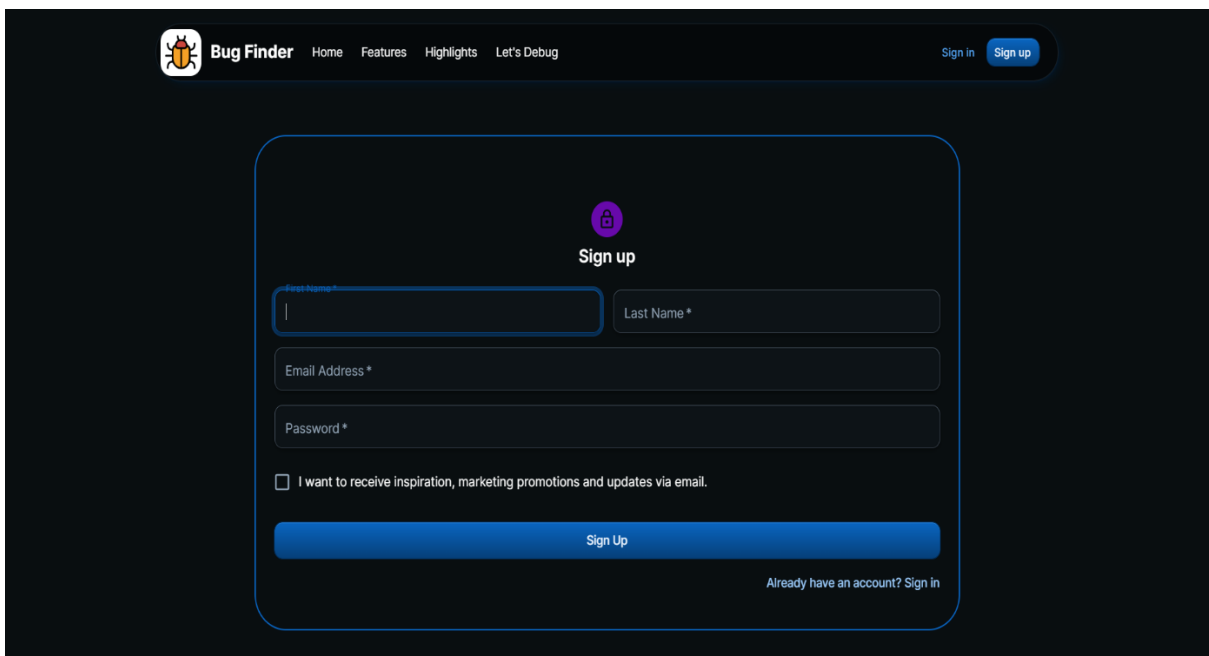
The results obtained from our bugs finder system like sign in page sign up page home page and Dashboard page .Here are some result .

### 10.1 Sign in page



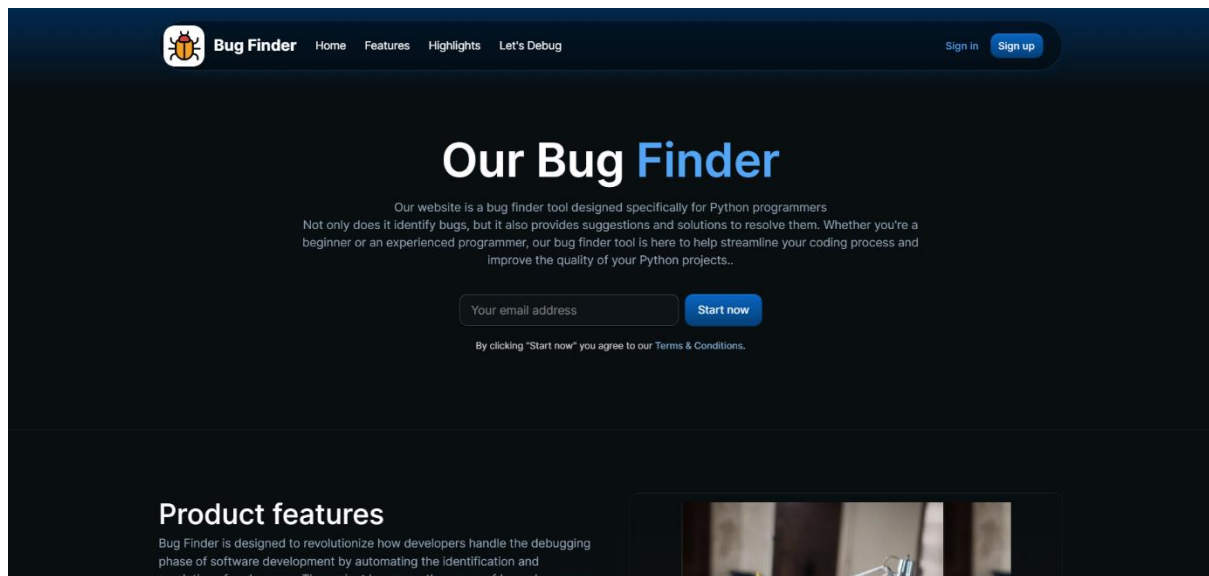
The screenshot shows the 'Sign in' page of the 'Bug Finder' application. The page has a dark blue background. At the top, there is a navigation bar with the 'Bug Finder' logo (a bug icon) and the text 'Bug Finder'. To the right of the logo are links for 'Home', 'Features', 'Highlights', and 'Let's Debug'. Further right are 'Sign in' and 'Sign up' buttons. The main content area is a light blue rounded rectangle. Inside, there is a purple lock icon and the text 'Sign in'. Below this are two input fields: 'Email Address \*' and 'Password \*'. There is a checkbox labeled 'Remember me'. A large blue button labeled 'Sign In' is centered below the input fields. At the bottom of the form, there are two links: 'Forgot password?' on the left and 'Don't have an account? Sign Up' on the right.

### 10.2 Sign Up

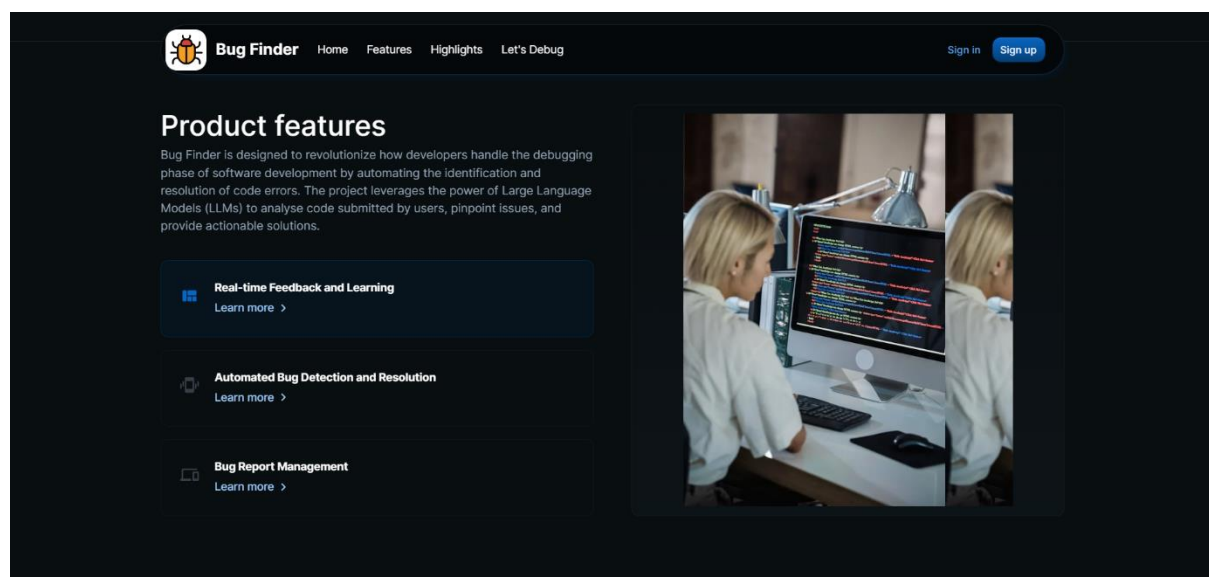


The screenshot shows the 'Sign up' page of the 'Bug Finder' application. The page has a dark blue background. At the top, there is a navigation bar with the 'Bug Finder' logo (a bug icon) and the text 'Bug Finder'. To the right of the logo are links for 'Home', 'Features', 'Highlights', and 'Let's Debug'. Further right are 'Sign in' and 'Sign up' buttons. The main content area is a light blue rounded rectangle. Inside, there is a purple lock icon and the text 'Sign up'. Below this are four input fields: 'First Name \*', 'Last Name \*', 'Email Address \*', and 'Password \*'. There is a checkbox labeled 'I want to receive inspiration, marketing promotions and updates via email.'. A large blue button labeled 'Sign Up' is centered below the input fields. At the bottom of the form, there is a link: 'Already have an account? Sign In'.

## 10.3 Home Page

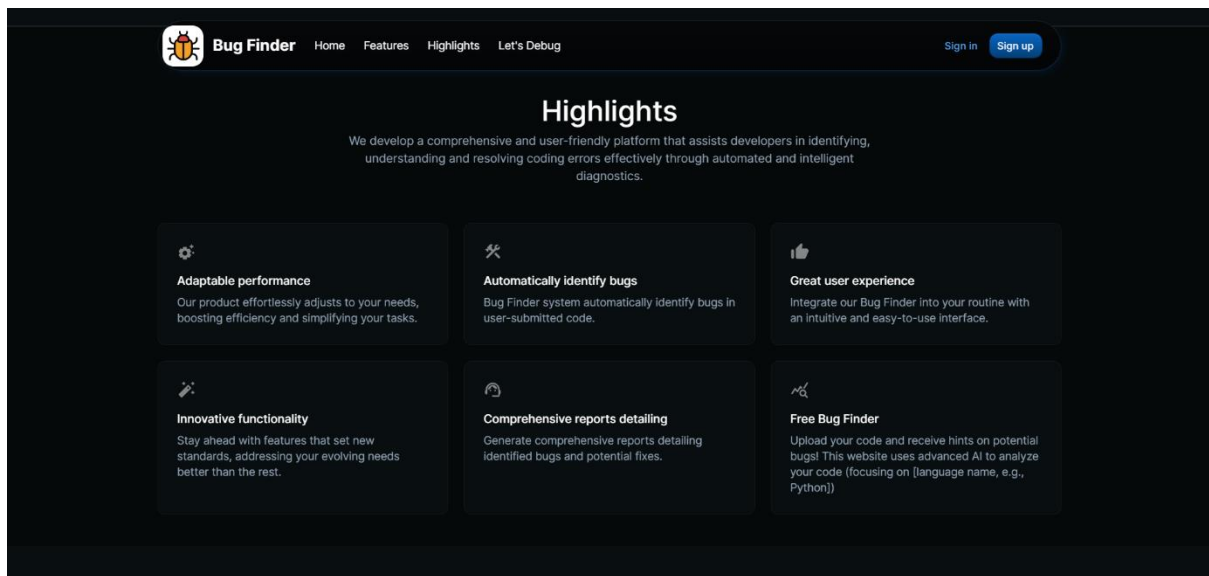


## 10.4 Features page






## 10.5 Highlights page



## 10.6 Let's Debug page

 **Bug Finder** [Home](#) [Features](#) [Highlights](#) [Let's Debug](#) [Sign In](#) [Sign up](#)

# Let's Debug Code

Our website is a bug finder tool designed specifically for Python programmers. Not only does it identify bugs, but it also provides suggestions and solutions to resolve them. Whether you're a beginner or an experienced programmer, our bug finder tool is here to help streamline your coding process and improve the quality of your Python projects..

### Write Your Python Code Here To Debug




```
1 def add(a,b):
2     return a+b;
3
4 add(a)
```

Let's Debug Code

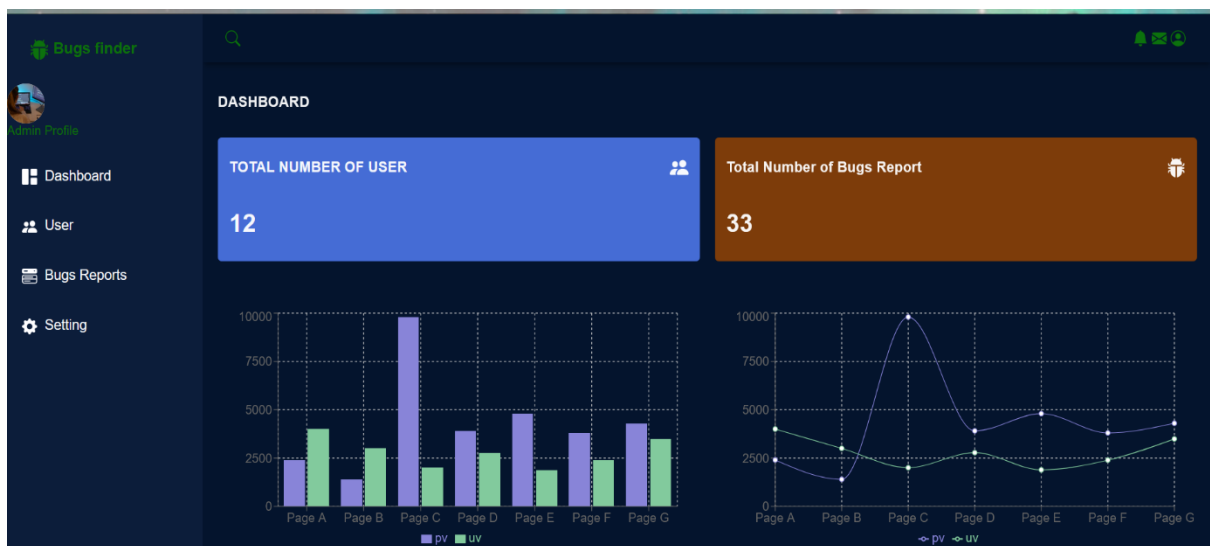
**Response :**

undefinedYou're absolutely right! This code snippet is in Python. The bug lies in the way the 'add' function is called: `python add(a)` **The Problem:** The 'add' function is defined to take **two** arguments ('a' and 'b'). When you call 'add(a)', you only provide one argument, resulting in a 'TypeError': `TypeError: add() missing 1 required positional argument: 'b'` **Solution:** To fix the code, you need to provide both arguments when calling the 'add' function. For example: `python def add(a, b): return a + b result = add(5, 3) # Call the function with two arguments print(result) # Output: 8` Now the code will correctly add the two values and print the sum.

[Privacy Policy](#) • [Terms of Service](#)  
Copyright © Bug Finder 2024

## 10.7 Admin Dashboard page



## **11 Task Remaining**

Task remaining for the completion for our work are:

- To work on user authentication mainly in login part with API.
- To integrate admin pages sufficiently.
- To provide the bugs reporting system.

## 12 Project Task and Time Schedule

Team Member	Role	Responsibilities
BIKASH BANJADE	Frontend Developer	Prepare the User Interface(UI) of Bug Finder Website.
APSARA ARYAL	Frontend Developer	Prepare the User Interface (UI) of the Admin Panel.
DHIRAJ YADAV	Backend Developer	Integrate the Gemini-1.5-flash model, Handle API endpoints requests from model and send responses to the Website.
SANDESH ADHIKARI	Backend Developer	Manage the database and handle documentation.

## 13 List of figures

### 13.1 Use Case Diagram

A use case diagram is a way to summarize details of a system and the users within that system. It is generally a graphical representation of interaction among different elements in a system.

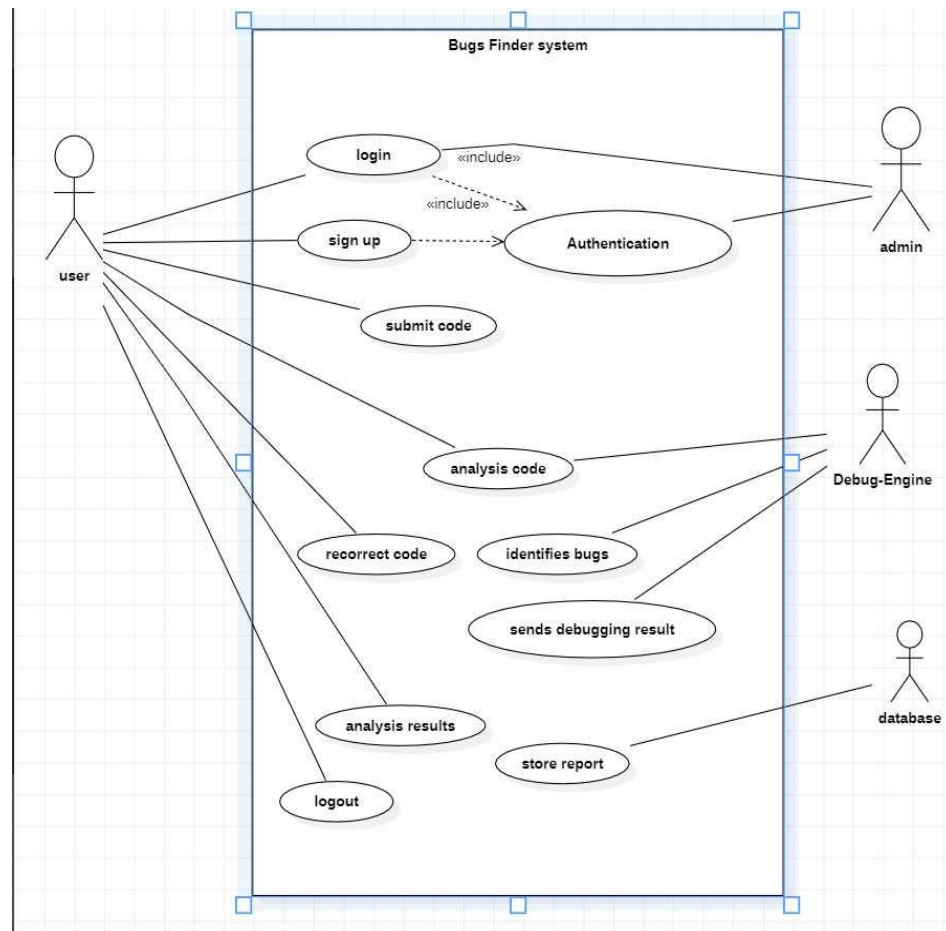


Figure 2: Use case Diagram

In our system, we have four actors: user, admin, debug-Engine and database. In the above diagram, various interactions of various actors with the system are illustrated. User can log in the system, and submit the code. Debug-Engine can analysis the code and find the bugs, database can store the report and admin can manage the user account and whole system in the show above figure.

## 13.2 Sequence Diagram

Sequence diagrams are a popular dynamic modelling solution in UML because they specifically focus on lifelines or the processes and objects that live simultaneously, and the messages exchanged between them to perform a function before the lifeline ends.

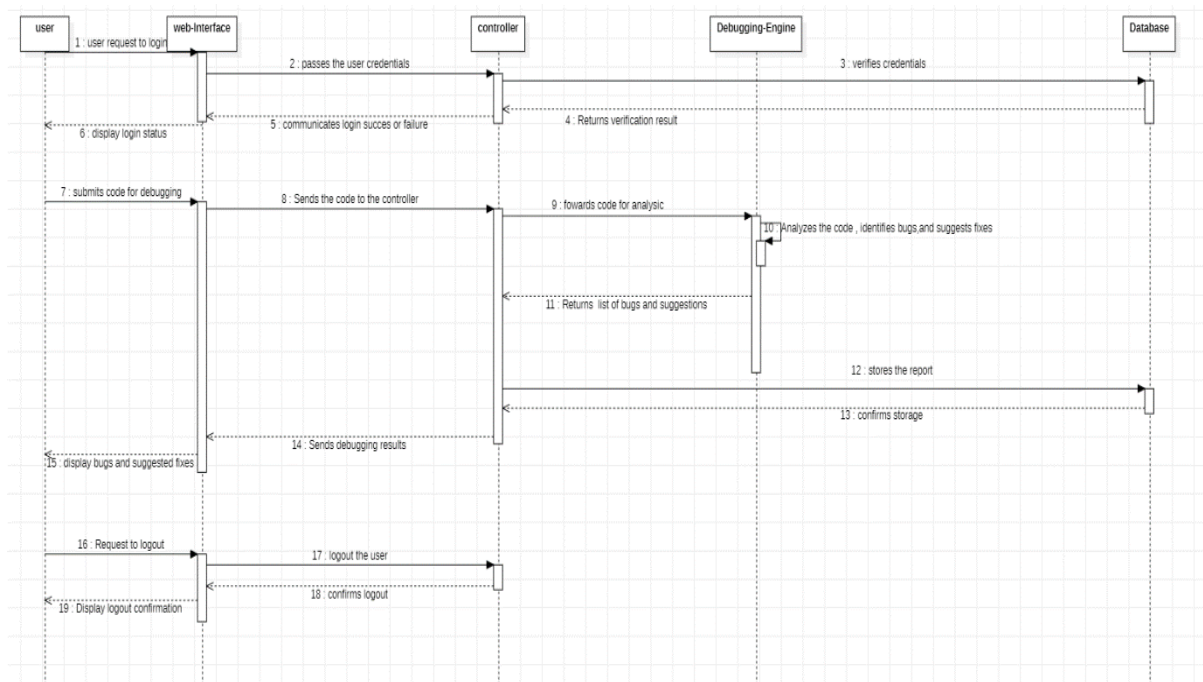


Figure 3: Sequence Diagram

Three object user, web interface, controller, debugging-Engine and database are shown in the above diagram.

### 13.3 Database Diagram

A database diagram is a blueprint for understanding a database's structure. It visually represents relationships and hierarchies within the system, aiding developers in construction and maintenance. Figure 4 presents a database schema designed for a system that connects users, Code\_Submission, Bug\_Report, and admin.

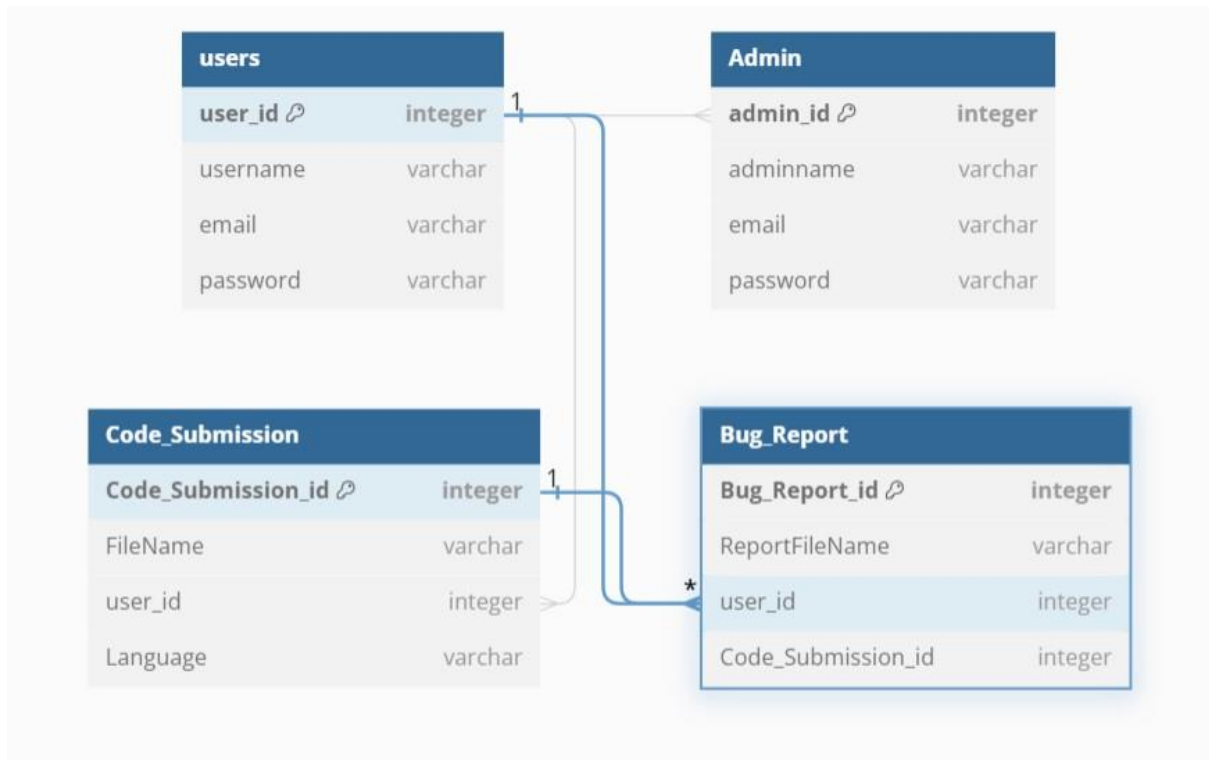


Figure 4: Database Diagram



## References

- 1) Aarnphm Aaron Pham , “*OpenLLM documentations*” [Online],  
Avaliable : <https://github.com/bentoml/OpenLLM?tab=readme-ov-file>
- 2) Krish Nayak , ”*Integration of OpenLLm model from Meta with frontend*” [Online],  
Avaliable : <https://youtu.be/cMJWC-csdK4?si=H8ROcmqPWRFOBrkt> , 2022
- 3) Xufeng Yao, Haoyang Li, Tsz Ho Chan, “HDLdebugger : Streamlining HDL debugging with Large Language Model” vol-4, p. 1-8, August 25–29, 2024
- 4) Muhammad Athar Ganaie, “*Machine Learning-Based Debugging Framework with LLMs*”  
Article : <https://www.marktechpost.com/2024/03/09/can-llms-debug-programs-like-human-developers-ucsd-researchers-introduce-ldb-a-machine-learning-based-debugging-framework-with-llms/> , 2024
- 5) Zhang, Y., Wang, L., Xie, T., & Zhang, L. (2019). A Survey of Automated Debugging Techniques for Contemporary Software Systems. *ACM Computing Surveys (CSUR)*, 52(3), 1-35.

# Project Progress Log Sheet

Nepal College of Information Technology

Balkumari, Lalitpur

## Project Progress Log Sheet

(The tabular section of this sheet shall be filled by the project supervisor every time (s)he consults with the team. The second section is the supervisor's approval, which allows the team to appear in a defense.)

Project Code: N-2020-LMS-1641

Project Title: Flicker->Debug Finder

Student's Roll No, Name:

201708 Dhiraj Kumar Yadav

201705 Bikas Banjade

201743 Apsara Aryal

201730 Sandesh Adhikari




Supervisor's Name: Mr./Mrs. Subash

Manandhar

Designation:

Institution:


Program and Batch: BESE, 2020

S.N.	Date	Discussion	Signature
1.	29 <sup>th</sup> May 2024	discuss on feedback from proposal defense	
2.	6 <sup>th</sup> June 2024	discuss on backend and frontend development of project	
3.	12 <sup>th</sup> June 2024	discuss on mid term, documentation, and features of project	
4.			
5.			

Allowed by me to participate in:

Mid-Term Defense: ☒

Final Defense: ☐

  
(Signature of the Supervisor)

\_\_\_\_\_  
(Signature of the Supervisor)