GitGrade Hackathon

Theme:AI + Code Analysis + Developer Profiling

Problem Statement:

You are tasked with creating an intelligent system that can **evaluate a student's GitHub repository** and convert it into a meaningful **Score + Summary + Personalized Roadmap**.

In today's tech world, a GitHub repository is a developer's tangible work but most students don't know how good, clean, or complete their code looks to a recruiter or mentor.

Your challenge is to design a system that:

1. Accepts a GitHub Repository URL as Input

The user will paste a link to **any public repository** they want analyzed.

2. Automatically Fetches Repository Data

Your system must gather publicly available details such as:

- Number of files and folder structure
- Code quality indicators (e.g., linting, complexity)
- README & documentation quality
- Test coverage (if present)
- Commit history & contribution consistency
- Language / tech stack usage
- Real-world applicability of the project
- Use of version control best practices (branches, PRs)

3. Evaluates the Repository on Multiple Dimensions

Your system should analyze the repo to judge:

- Code quality & readability
- Project structure & organization
- Documentation & clarity
- Test coverage & maintainability
- Real-world relevance & usefulness
- Commit & development consistency

4. Generates Three Key Outputs:

A. Score / Rating

Example formats:

- 0–100
- Beginner / Intermediate / Advanced
- Bronze / Silver / Gold

B. Written Summary

A short evaluation describing the repository's current quality.

Example:
"Code structure is clean and consistent, but documentation and tests need improvement."

C. Personalized Roadmap

Actionable steps the student must follow, such as:

- Improve folder structure
- Add README.md with project overview and instructions
- Write unit / integration tests
- Follow Git best practices (branches, PRs, commits)
- Add CI/CD pipelines
- Optimize code for readability and maintainability

This roadmap should feel like **guidance from an AI coding mentor**.

Goal of the Problem:

The goal is to build a "Repository Mirror", a system that reflects the real strengths and weaknesses of a project **based entirely on its GitHub repository**.

Your solution should prioritize:

- **Accuracy**
  (Make reasonable judgments based on real repository data.)
- **Honest Feedback**
  (Tell students exactly what's strong and what needs work.
- **Actionable Improvement**
  (A roadmap they can immediately follow.)

This challenge is not about marks, theory, or memorization. It's about evaluating a developer's **practical coding output**.

Sample Input/Output Examples

Example 1

Input:
`https://github.com/rahul-dev-ai/todo-app`

Output:
Score: **78 / 100**
Summary: Strong code consistency and folder structure; needs more tests and documentation.
Roadmap:

- Add unit tests
- Improve README with project instructions
- Introduce CI/CD using GitHub Actions

Example 2

Input:
`https://github.com/sneha-codes/weather-dashboard`

Output:
Score: **42 / 100**
Summary: Basic project structure but poor documentation and inconsistent commits.
Roadmap:

- Add README with setup instructions
- Restructure folders
- Commit regularly with meaningful messages

Example 3

Input:
`https://github.com/manish-projects/ecommerce-site`

Output:
Score: **91 / 100**
Summary: Excellent project depth and clean codebase.
Roadmap:

- Add automated tests
- Improve issue tracking
- Contribute project to open-source

Submission Guidelines:

Participants must submit a **GitHub Repository Link** containing:

- Source Code
- README explaining the approach
- A **screen recording** showing the project working end-to-end