Hackathon Project Phases Template

Project Title: SmartSDLC - AI-Enhanced Software Development Lifecycle
Team Name: Smartforge

Team Members:
Dwarampudi S R G N V Sasank Reddy
Ch P V S S S Naveen
Dasari Likhith

1 Phase-1: Brainstorming & Ideation

Objective: To identify critical challenges in software development and propose an AI-driven solution to enhance the Software Development Lifecycle (SDLC).
Key Points:

1. **Problem Statement:** Manual code reviews are time-consuming and error-prone, documentation is often outdated or inconsistent, and bugs detected late increase costs and risks, leading to reduced productivity and delayed releases.
2. **Proposed Solution:** SmartSDLC, an AI-powered assistant that automates code reviews, documentation generation, and bug prediction using Generative AI and machine learning.
3. **Target Users:** Software developers, QA engineers, and project managers in small to large development teams, including startups and open-source contributors.
4. **Expected Outcome:** 40–60% reduction in code review and documentation time, early bug detection, improved team productivity, and faster release cycles with enhanced code quality.

2 Phase-2: Requirement Analysis

Objective: To define technical and functional requirements for SmartSDLC and identify potential constraints.
Key Points:

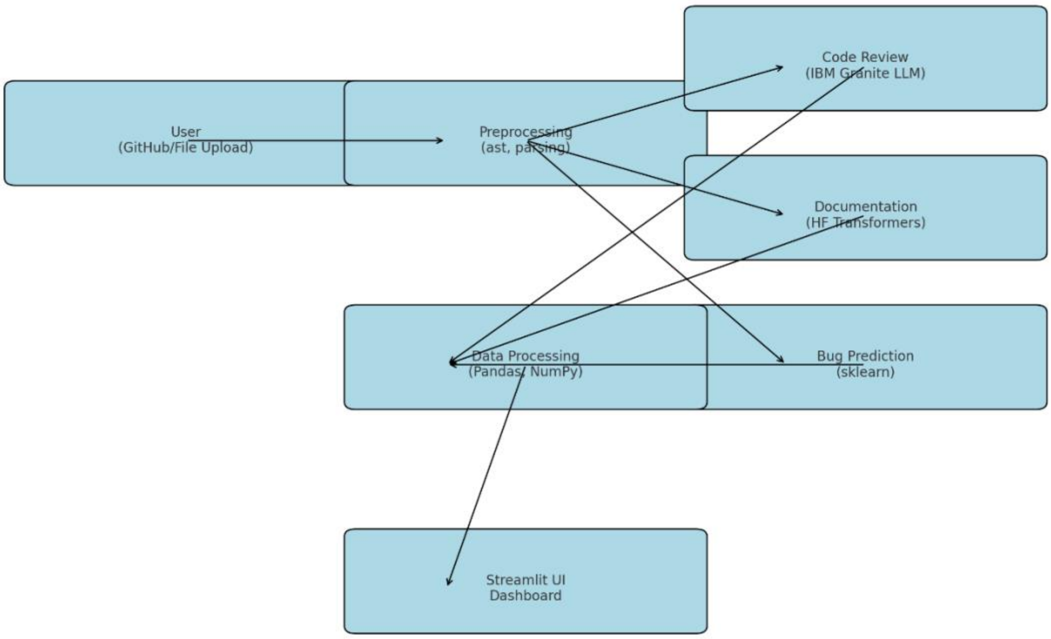
1. **Technical Requirements:**
   - IBM Granite LLM or Hugging Face Transformers for code review and documentation.
   - scikit-learn, Pandas, NumPy for bug prediction.
   - Streamlit for the user interface.
   - GitHub API for version control integration.
   - Python as the primary programming language.
2. **Functional Requirements:**

- Automated code review for logical bugs, security vulnerabilities, and style violations.
- Auto-generation of docstrings, README files, and changelogs.
- Bug prediction based on code semantics and historical patterns.
- Interactive dashboard for displaying results.

3. **Constraints & Challenges:**
   - Limited computational resources (reliance on free/student tiers).
   - Support initially limited to Python and JavaScript.
   - Time constraint of 10 days for prototype development.
   - Ensuring LLM accuracy for diverse codebases.

3 Phase-3: Project Design

Objective: To design the system architecture and user experience for SmartSDLC.



Key Points:

1. **System Architecture:**
   - Code Input Layer: Upload files or fetch from GitHub via API.
   - Preprocessing Module: Parse code using Python libraries (e.g., ast).
   - AI Modules: Code review (IBM Granite LLM), documentation (Hugging Face Transformers), bug prediction (scikit-learn).
   - Data Processing: Pandas and NumPy for data handling.
   - UI Layer: Streamlit dashboard for results.

2. **User Flow:**
   - User uploads code or connects GitHub repository.
   - System processes code and displays review comments, documentation, and bug alerts.
   - User navigates dashboard to view or download results.
3. **UI/UX Considerations:**
   - Clean, intuitive Streamlit interface.
   - Clear categorization of feedback (reviews, docs, bugs).
   - Responsive design for ease of use.

4 Phase-4: Project Planning (Agile Methodologies)

Objective: Break down development tasks for efficient completion using Agile sprints.

Sprint	Task	Priority	Duration	Deadline	Assigned To	Dependencie
Sprint 1	Setup GitHub repo	High	4 hours	Day 1	Sasank	None
Sprint 1	Install dependencies	High	4 hours	Day 1	Naveen	None
Sprint 1	Integrate IBM Granite LLM	Medium	8 hours	Day 2	Likhith	Repo setup
Sprint 2	Develop code review module	High	12 hours	Day 4	Sasank	LLM integration
Sprint 2	Implement documentation generator	High	12 hours	Day 5	Naveen	LLM integration
Sprint 2	Build bug prediction model	Medium	12 hours	Day 6	Likhith	Preprocessing module
Sprint 3	Develop Streamlit UI	High	12 hours	Day 8	Sasank	All modules
Sprint 3	End-to-end testing	High	8 hours	Day 9	Naveen	UI completion
Sprint 3	Prepare demo	Medium	8 hours	Day 10	Likhith	Testing

Sprint Planning with Priorities:

- **Sprint 1 - Setup & Integration (Day 1–2):** Environment setup, repository creation, and LLM integration.
- **Sprint 2 - Core Features & Debugging (Day 3–6):** Develop code review, documentation, and bug prediction modules.
- **Sprint 3 - Testing, Enhancements & Submission (Day 7–10):** UI develop-

ment, testing, and demo preparation.

5 Phase-5: Project Development

Objective: Implement core features of the SmartSDLC system.
Key Points:

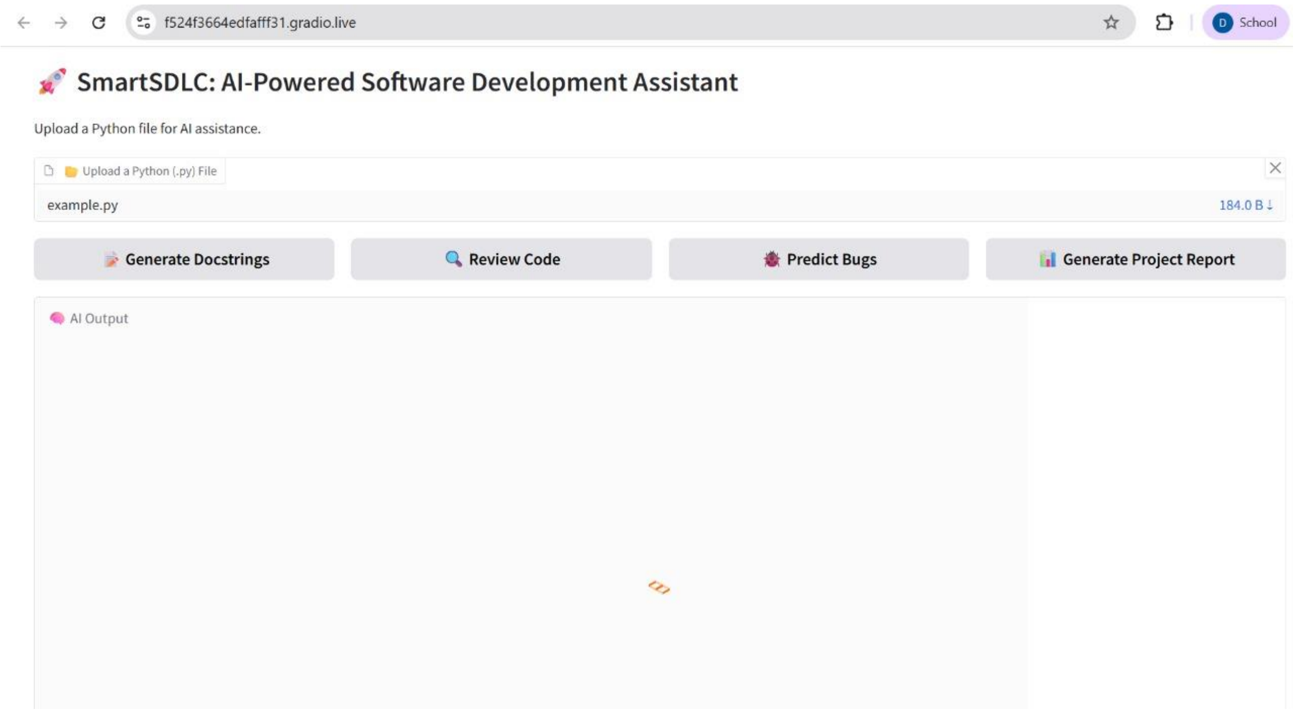
1. **Technology Stack Used:**
   - Python, IBM Granite LLM, Hugging Face Transformers.
   - scikit-learn, Pandas, NumPy.
   - Streamlit, Git, GitHub API.
2. **Development Process:**
   - Code input via file upload or GitHub API.
   - Preprocessing using Python ast library.
   - AI-driven analysis for reviews, documentation, and bug prediction.
   - Integration into Streamlit dashboard.
3. **Challenges & Fixes:**
   - Challenge: Limited LLM accuracy for complex codebases. Fix: Use sample repositories for fine-tuning.
   - Challenge: Resource constraints. Fix: Leverage free tiers and optimize code.
   - Challenge: Tight timeline. Fix: Prioritize core features and use Agile sprints.

6 Phase-6: Functional & Performance Testing

Objective: Validate the functionality and performance of SmartSDLC.

Test Case ID	**Category**	**Test Scenario**	**Expected Outcome**	**Status**	**Tester**
TC01	Functional	Upload Python code file	Code parsed and reviewed correctly	completed	Sasank
TC02	Functional	Fetch code from GitHub	Code fetched and processed	Pending	Likhith
TC03	Functional	Generate docstrings	Accurate docstrings generated	completed	Naveen
TC04	Functional	Predict buggy code	Bug-prone segments flagged	completed	Sasank
TC05	Performance	Process 1000-line	Review completed in <10s	completed	Naveen

		code			
TC06	UI	Display results in dashboard	Results clear and navigable	completed	Likhith



f524f3664edfaff31.gradio.live

SmartSDLC: AI-Powered Software Development Assistant

Upload a Python file for AI assistance.

Interface

7 Final Submission

1. **Project Report:** Based on this template, detailing all phases.
2. **Demo Video:** 3–5-minute video showcasing SmartSDLC features.
3. **GitHub/Code Repository Link:** Public repository with code and documentation.
4. **Presentation:** Slides summarizing the project, to be presented during the hackathon.