PHASE1:
Brainstorm & Idea Prioritization

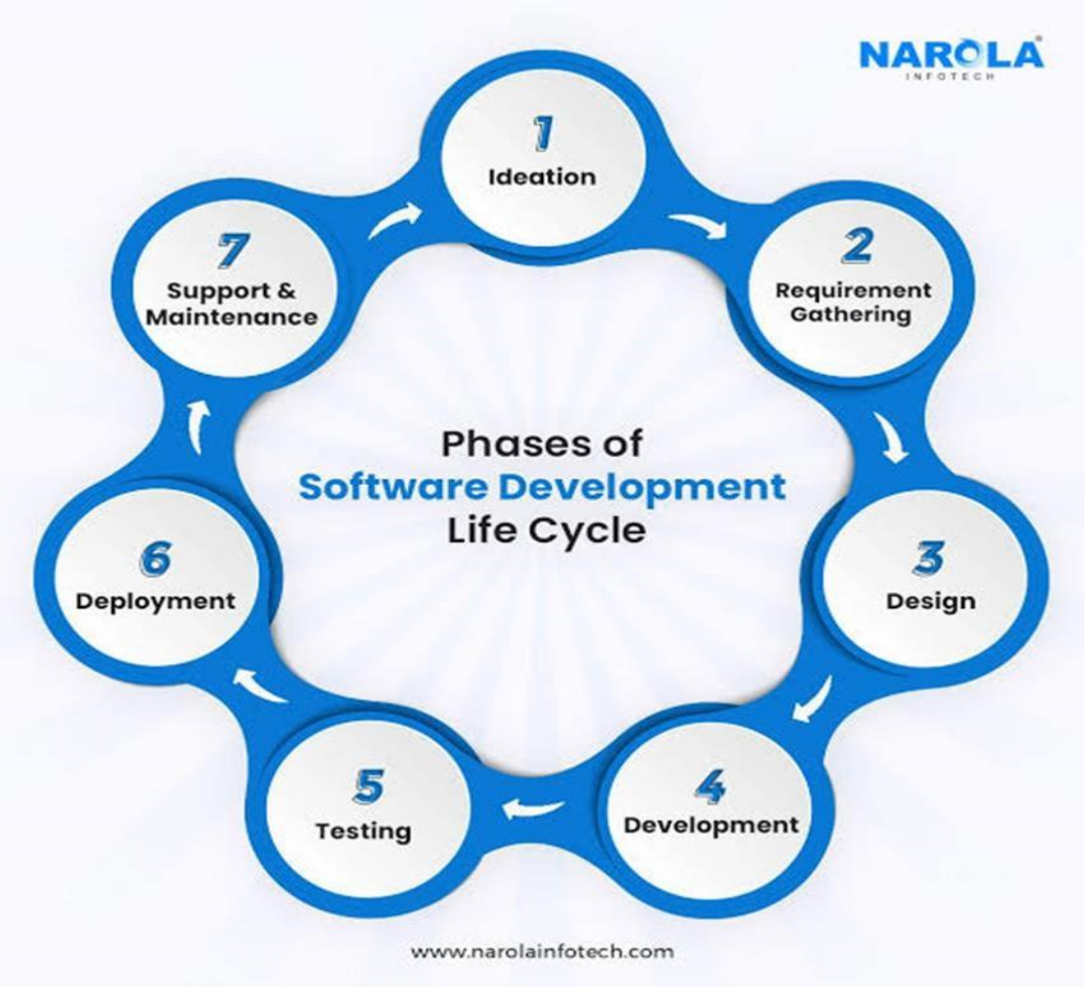
Brainstorm & Idea Prioritization

Date	1 july 2025
Team ID	LTVIP2025TMID34717
Project Name	smartsdlc – ai-enhanced software development lifecycle
Maximum Marks	4 Marks

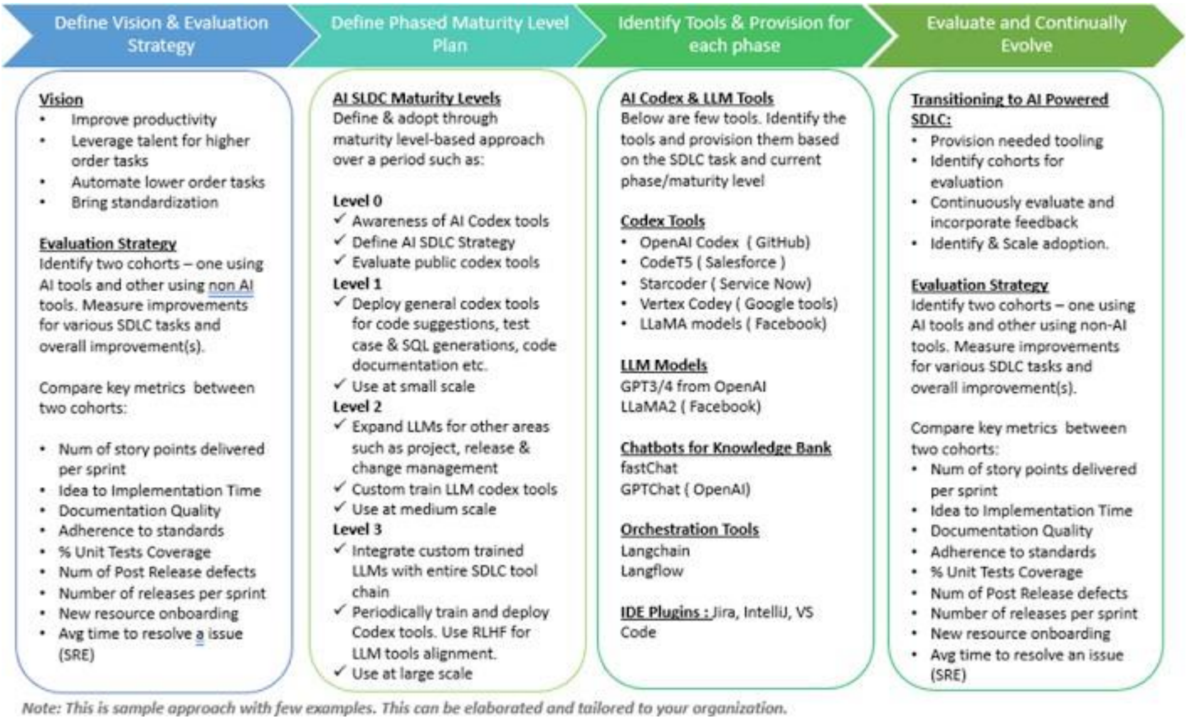
Brainstorm & Idea Prioritization:

SmartSdlc – AI-enhanced software development lifecycle" leverages AI to transform traditional brainstorming and planning into a data-driven, collaborative, and efficient process. By integrating AI tools for market analysis, requirements gathering, and feature prioritization, the project sets a strong foundation for innovation and success in subsequent SDLC phases.

Step-1: Team Gathering, Collaboration and Select the Problem Statement



Step-2: Brainstorm, Idea Listing and Grouping



Define Vision & Evaluation Strategy

Vision
- Improve productivity
- Leverage talent for higher order tasks
- Automate lower order tasks
- Bring standardization

Evaluation Strategy
Identify two cohorts – one using AI tools and other using non AI tools. Measure improvements for various SDLC tasks and overall improvement(s).

Compare key metrics between two cohorts:

- Num of story points delivered per sprint
- Idea to Implementation Time
- Documentation Quality
- Adherence to standards
- % Unit Tests Coverage
- Num of Post Release defects
- Number of releases per sprint
- New resource onboarding
- Avg time to resolve a issue (SRE)

Define Phased Maturity Level Plan

AI SLDC Maturity Levels
Define & adopt through maturity level-based approach over a period such as:

Level 0
- ✓ Awareness of AI Codex tools
- ✓ Define AI SDLC Strategy
- ✓ Evaluate public codex tools

Level 1
- ✓ Deploy general codex tools for code suggestions, test case & SQL generations, code documentation etc.
- ✓ Use at small scale

Level 2
- ✓ Expand LLMs for other areas such as project, release & change management
- ✓ Custom train LLM codex tools
- ✓ Use at medium scale

Level 3
- ✓ Integrate custom trained LLMs with entire SDLC tool chain
- ✓ Periodically train and deploy Codex tools. Use RLHF for LLM tools alignment.
- ✓ Use at large scale

Identify Tools & Provision for each phase

AI Codex & LLM Tools
Below are few tools. Identify the tools and provision them based on the SDLC task and current phase/maturity level

Codex Tools
- OpenAI Codex (GitHub)
- CodeT5 (Salesforce)
- Starcoder (Service Now)
- Vertex Codey (Google tools)
- LLaMA models (Facebook)

LLM Models
GPT3/4 from OpenAI
LLaMA2 (Facebook)

Chatbots for Knowledge Bank
fastChat
GPTChat (OpenAI)

Orchestration Tools
Langchain
Langflow

IDE Plugins : Jira, IntelliJ, VS Code

Evaluate and Continually Evolve

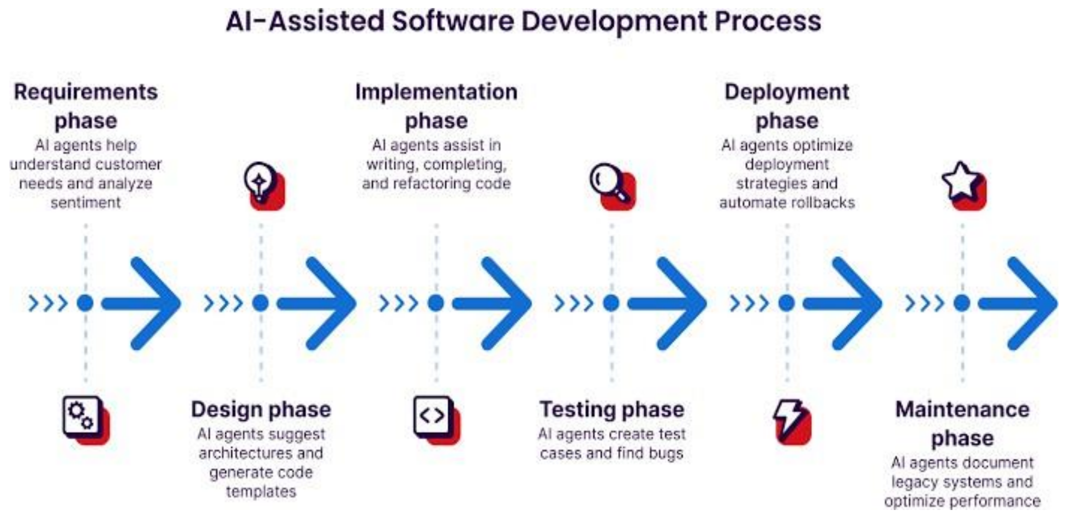
Transitioning to AI Powered SDLC:
- Provision needed tooling
- Identify cohorts for evaluation
- Continuously evaluate and incorporate feedback
- Identify & Scale adoption.

Evaluation Strategy
Identify two cohorts – one using AI tools and other using non-AI tools. Measure improvements for various SDLC tasks and overall improvement(s).

Compare key metrics between two cohorts:
- Num of story points delivered per sprint
- Idea to Implementation Time
- Documentation Quality
- Adherence to standards
- % Unit Tests Coverage
- Num of Post Release defects
- Number of releases per sprint
- New resource onboarding
- Avg time to resolve an issue (SRE)

Note: This is sample approach with few examples. This can be elaborated and tailored to your organization.

Step-3: Idea Prioritization



AI-Assisted Software Development Process

Requirements phase
AI agents help understand customer needs and analyze sentiment

Design phase
AI agents suggest architectures and generate code templates

Implementation phase
AI agents assist in writing, completing, and refactoring code

Testing phase
AI agents create test cases and find bugs

Deployment phase
AI agents optimize deployment strategies and automate rollbacks

Maintenance phase
AI agents document legacy systems and optimize performance

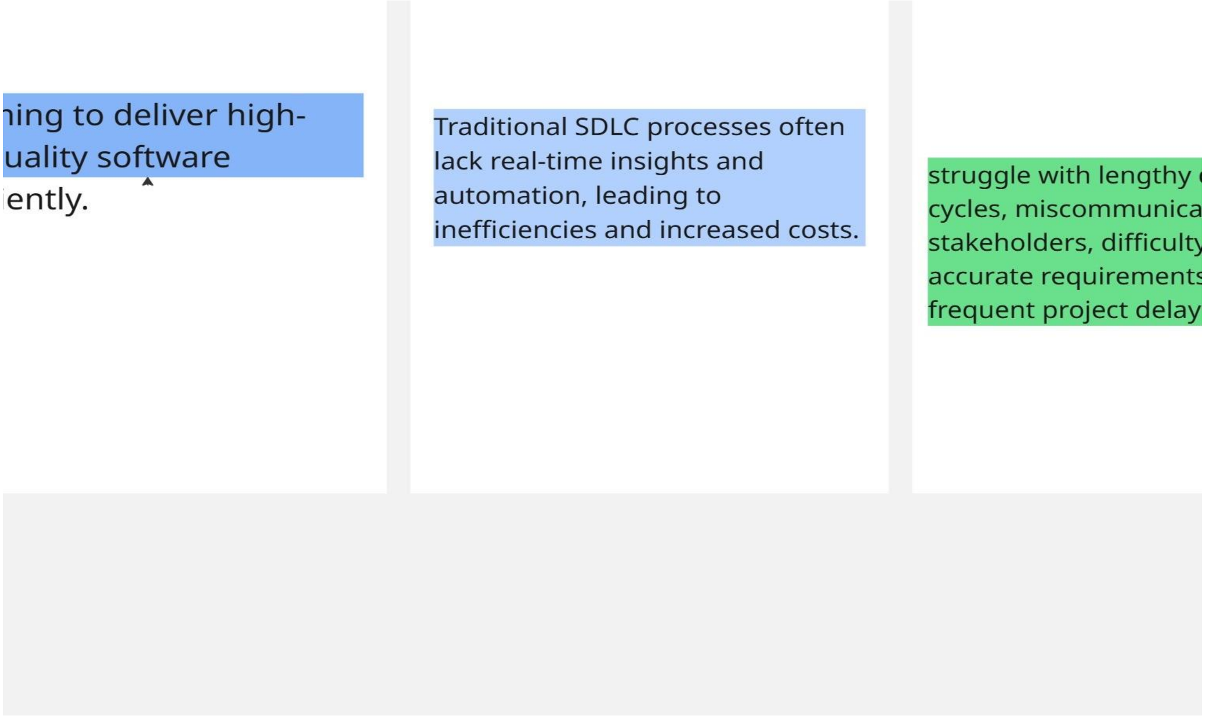
DEFINE PROBLEM STATEMENTS

Define the Problem Statements

Date	31 May 2025
Team ID	LTVIP2025TMID34717
Project Name	Smartsdlc – ai-enhanced software development lifecycle
Maximum Marks	2 Marks

Customer Problem Statement :

Software development teams face challenges like lengthy development cycles, miscommunication, and unclear requirements, especially during the ideation and requirements gathering phases. These issues lead to project delays, increased costs, and products that may not fully meet user needs. Traditional SDLC processes lack real-time insights and automation, causing inefficiencies and frustration. Teams need an AI-enhanced solution that streamlines communication, automates repetitive tasks, and provides better understanding of customer needs. By addressing these pain points, smartsdlc aims to help teams deliver high-quality software faster and create experiences users will love.



ing to deliver high-
uality software
ently.

Traditional SDLC processes often lack real-time insights and automation, leading to inefficiencies and increased costs.

struggle with lengthy
cycles, miscommunica
stakeholders, difficulty
accurate requirements
frequent project delay

Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	developer	Deliver high quality software	Traditional Sldc's often lack time	That there is a delay in projects	frustated

Empathize & Discover

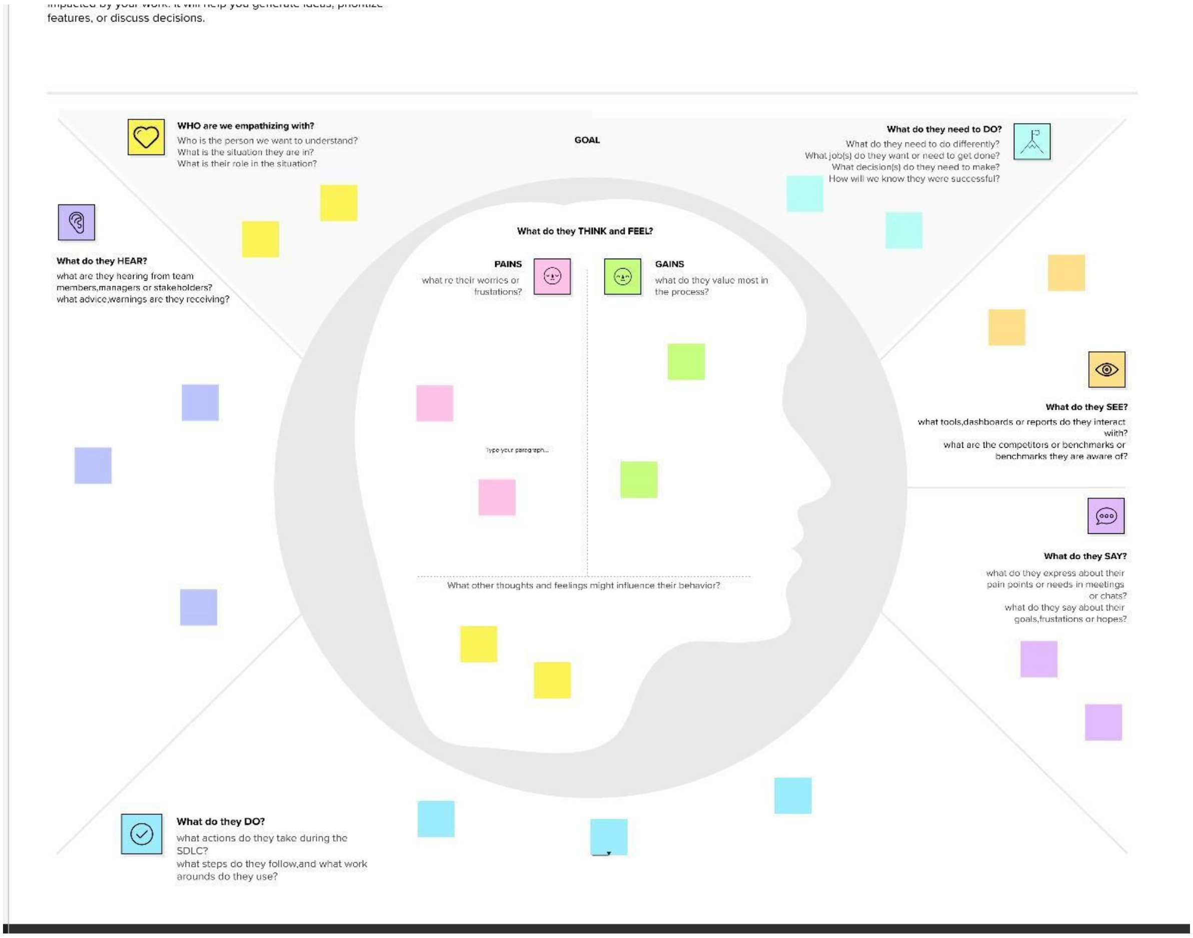
Empathize & Discover

Date	25 JUNE 2025
Team ID	LTVIP2025TMID34717
Project Name	Smartsdlc – ai-enhanced software development lifecycle
Maximum Marks	4 Marks

Empathy Map Canvas:

In a Smart SDLC project, user behavior, sentiment, and preferences are gathered from sources like social media, reviews, support tickets, and in-app activity. Advanced analytics and AI help turn this data into actionable insights, leading to better decisions and a more user-focused development process, while maintaining strong data privacy and quality standards.

Share


features, or discuss decisions.
WHO are we empathizing with?
Who is the person we want to understand?
What is the situation they are in?
What is their role in the situation?
GOAL
What do they need to DO?
What do they need to do differently?
What job(s) do they want or need to get done?
What decision(s) do they need to make?
How will we know they were successful?
What do they HEAR?
what are they hearing from team members,managers or stakeholders?
what advice,warnings are they receiving?
What do they THINK and FEEL?
PAINS
what re their worries or frustations?
GAINS
what do they value most in the process?
Type your paragraph...
What other thoughts and feelings might influence their behavior?
What do they SEE?
what tools,dashboards or reports do they interact wiith?
what are the competitors or benchmarks or benchmarks they are aware of?
What do they SAY?
what do they express about their pain points or needs in meetings or chats?
what do they say about their goals,frustations or hopes?
What do they DO?
what actions do they take during the SDLC?
what steps do they follow,and what work arounds do they use?

PHASE-II:

REQUIREMENT ANALYSIS

- Python 3.10.0
- FastAPI
- Streamlit
- IBM Watsonx AI & Granite Models
- LangChain
- Uvicorn
- PyMuPDF (fitz)
- Git & GitHub
- Frontend Libraries

Phase III:

Project

Design

(STEP-1)

Problem – Solution

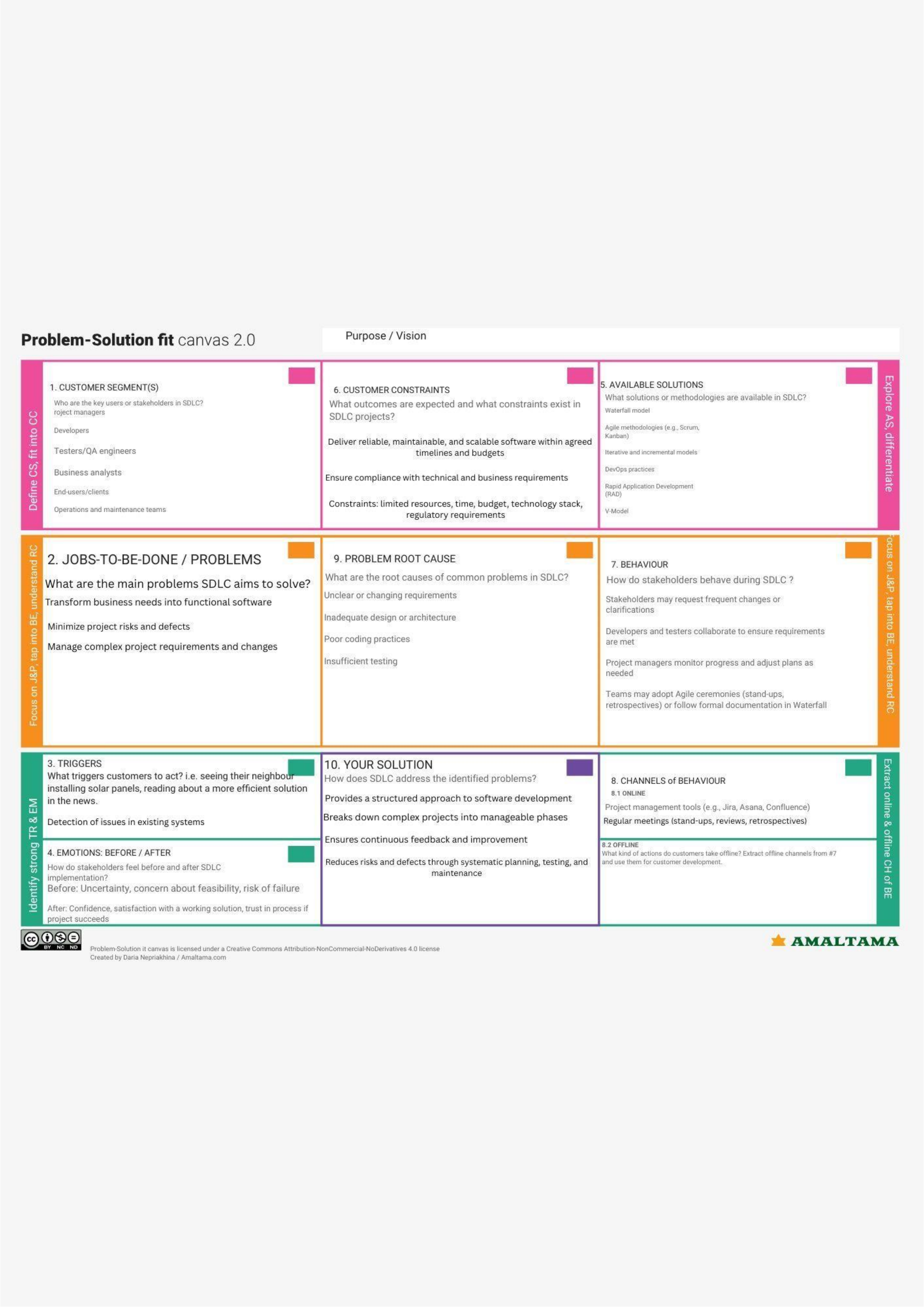
Date	25 june 2025
Team ID	LTVIP2025TMID34717
Project Name	smartsdlc – ai-enhanced software development lifecycle
Maximum Marks	2 Marks

Problem – Solution :

The solution to the challenges faced in traditional SDLC—such as lengthy cycles, miscommunication, unclear requirements, and lack of real-time insights—lies in integrating AI and automation throughout the software development lifecycle. AI-enhanced SDLC leverages technologies like natural language processing (NLP), generative AI, and machine learning to streamline and optimize every phase

Purpose:

- □ **Requirements Gathering:** AI analyzes client briefs, stakeholder interviews, and user feedback to extract clea , actionable requirements, reducing ambiguity and accelerating the analysis ph ase by up to 60%.
- □ **Design and Architecture:** Generative AI proposes system architectures, generates tes visual diagrams, and simula performance scenarios to optimize designs and future-proof applications .
- □ **Implementation:** AI acts as a digital copilot, providing real-time code suggestions, automated bug detection, refactoring, and code quality improvements, allowing developers to focus on complex tasks and reducing manual effort.
- **Testing:** AI automates test case generation, optimizes coverage, executes tests, and provides smart reporting, improving test efficiency by 25-50% and enhancing software quality.
- **Deployment and Maintenance:** AI-driven CI/CD pipelines predict failures, optimize build and deployment processes, and monitor system performance to ensure smoother releases and faster time-to-market.
- **Collaboration and Documentation:** AI automates documentation, meeting transcription, and task management, improving communication and reducing misalignment among teams.



Problem-Solution fit canvas 2.0

Purpose / Vision

Define CS, fit into CC

1. CUSTOMER SEGMENT(S)

Who are the key users or stakeholders in SDLC?
roject managers

Developers

Testers/QA engineers

Business analysts

End-users/clients

Operations and maintenance teams

6. CUSTOMER CONSTRAINTS

What outcomes are expected and what constraints exist in SDLC projects?

Deliver reliable, maintainable, and scalable software within agreed timelines and budgets

Ensure compliance with technical and business requirements

Constraints: limited resources, time, budget, technology stack, regulatory requirements

5. AVAILABLE SOLUTIONS

What solutions or methodologies are available in SDLC?

Waterfall model

Agile methodologies (e.g., Scrum, Kanban)

Iterative and incremental models

DevOps practices

Rapid Application Development (RAD)

V-Model

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS

What are the main problems SDLC aims to solve?

Transform business needs into functional software

Minimize project risks and defects

Manage complex project requirements and changes

9. PROBLEM ROOT CAUSE

What are the root causes of common problems in SDLC?

Unclear or changing requirements

Inadequate design or architecture

Poor coding practices

Insufficient testing

7. BEHAVIOUR

How do stakeholders behave during SDLC ?

Stakeholders may request frequent changes or clarifications

Developers and testers collaborate to ensure requirements are met

Project managers monitor progress and adjust plans as needed

Teams may adopt Agile ceremonies (stand-ups, retrospectives) or follow formal documentation in Waterfall

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS

What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.

Detection of issues in existing systems

4. EMOTIONS: BEFORE / AFTER

How do stakeholders feel before and after SDLC implementation?

Before: Uncertainty, concern about feasibility, risk of failure

After: Confidence, satisfaction with a working solution, trust in process if project succeeds

10. YOUR SOLUTION

How does SDLC address the identified problems?

Provides a structured approach to software development

Breaks down complex projects into manageable phases

Ensures continuous feedback and improvement

Reduces risks and defects through systematic planning, testing, and maintenance

8. CHANNELS of BEHAVIOUR

8.1 ONLINE

Project management tools (e.g., Jira, Asana, Confluence)

Regular meetings (stand-ups, reviews, retrospectives)

8.2 OFFLINE

What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.

Extract online & offline CH of BE

Created by Daria Nepriakhina / Amaltama.com

AMALTAMA

Proposed Solution

Date	25 june 2025
Team ID	LTVIP2025TMID34717
Project Name	smartsdlc – ai-enhanced software development lifecycle
Maximum Marks	2 Marks

Proposed Solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Software development teams face challenges like lengthy development cycles, mis- communication and unclear requirements, especially during the ideation and requirements gathering phases.
2.	Idea / Solution description	AI-enhanced SDLC leverages technologies like natural language processing (NLP), generative AI, and machine learning to streamline and optimize every phase.

3.	Novelty / Uniqueness	Smart SDLC uniquely embeds AI-driven automation and real-time insights throughout the development lifecycle, enhancing speed and quality. It automates requirements, improves collaboration, and continuously integrates user feedback for better alignment. Proactive risk prediction and personalized support make the process more efficient and user-centric than traditional SDLC.
4.	Social Impact / Customer Satisfaction	Smart SDLC improves social impact by enabling faster, more inclusive software delivery that addresses real-world needs. It boosts customer satisfaction through real-time insights and continuous user feedback, ensuring high-quality, user-focused products.
5.	Business Model (Revenue Model)	Smart SDLC can generate revenue through a subscription-based SaaS model, offering tiered plans for organizations of different sizes and needs. Additional revenue streams may include one-time setup fees, premium support, and consulting services for customization and integration. This approach ensures predictable recurring income while providing flexibility and value to customers seeking efficient, AI-driven software development solutions

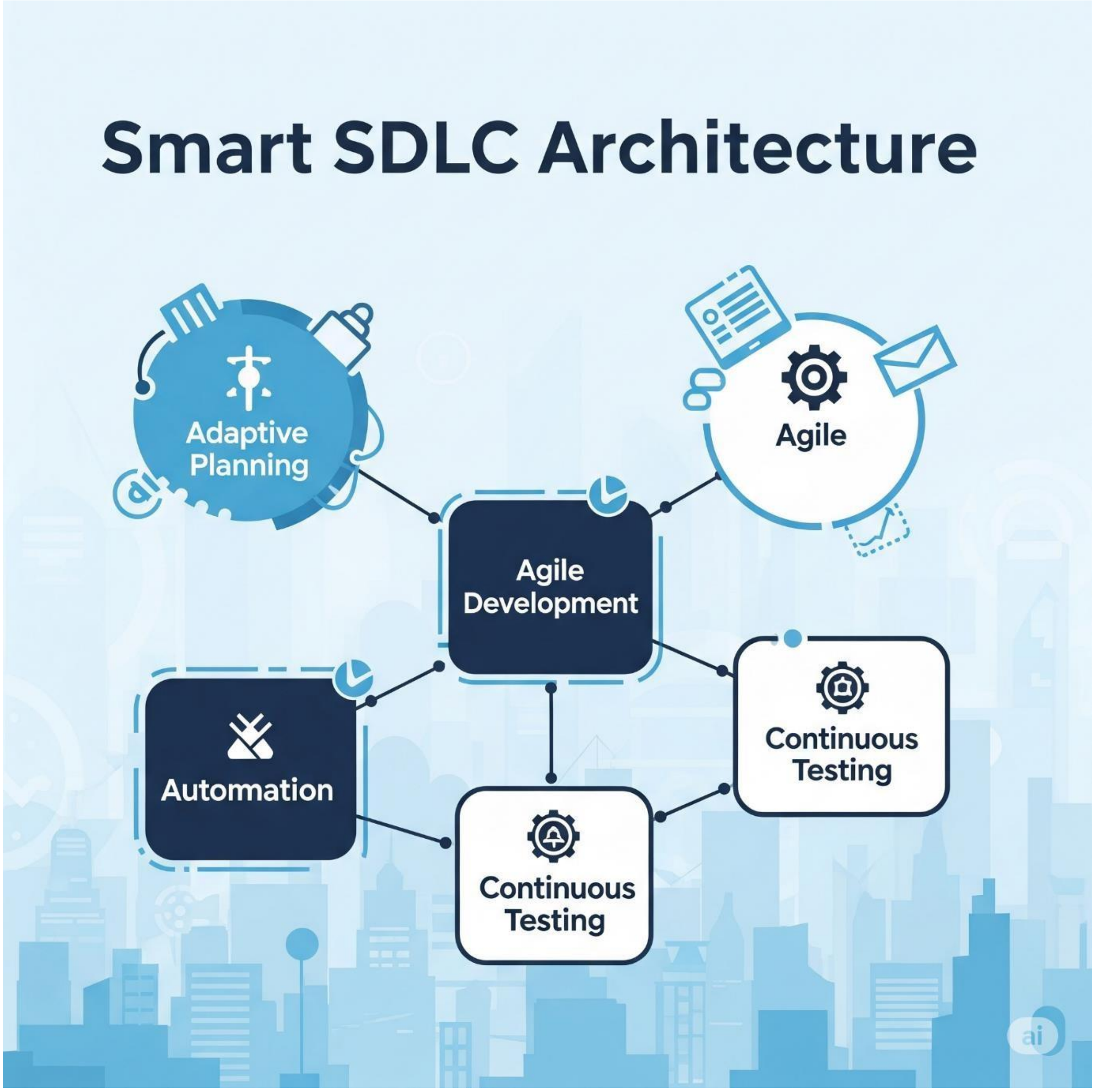
6.	Scalability of the Solution	The Smart SDLC solution is highly scalable, supporting growth in users, data, and project complexity without sacrificing performance or reliability. By leveraging microservices architecture, cloud-native technologies, and automation, it can be scaled horizontally (adding more nodes) or vertically (upgrading resources) as needed. This ensures efficient resource use, fast response times, and adaptability, making the solution suitable for organizations of any size or workload

Solution Architecture

Date	25 june 2025
Team ID	LTVIP2025TMID34717
Project Name	Smartsdlc – ai-enhanced software development lifecycle
Maximum Marks	4 Marks

Solution Architecture:

At the highest level, Smart SDLC connects end-users (developers, testers, project managers, stakeholders) with an AI-driven platform that automates and enhances every phase of the software development lifecycle.


Smart SDLC Architecture
Adaptive Planning
Agile
Agile Development
Automation
Continuous Testing
Continuous Testing
ai

Project Design

(STEP2)

Data Flow Diagram

Date	25 June 2025
Team ID	LTVIP2025TMID34717
Project Name	Smartsdlc – ai-enhanced software development lifecycle
Maximum Marks	4 Marks

Data Flow Diagrams:

A paradigm shift from traditional software development, integrating Artificial Intelligence (AI) at various stages to automate, optimize, and intelligentize the entire software development lifecycle. The core idea is to leverage AI to augment human capabilities, reduce manual effort, improve quality, and accelerate delivery.

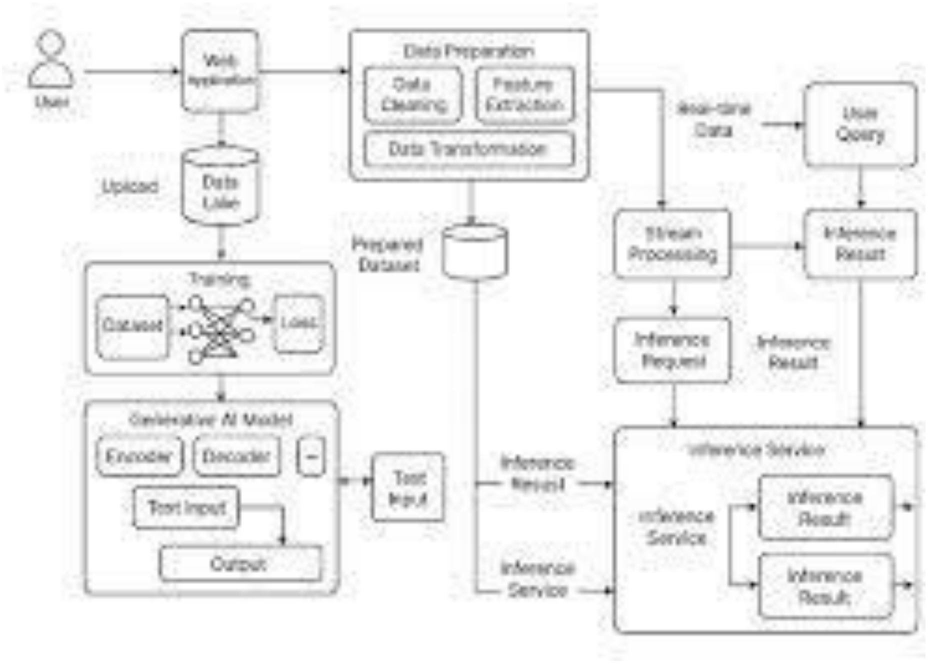
- **AI-Assisted Requirements & Planning (Process 1.0):**
  - **Traditional:** Manual requirements gathering, analysis, and documentation.
  - **AI Enhancement:** AI can analyze unstructured text from client communications, existing documentation, and industry trends to identify implicit requirements, potential conflicts, and ambiguities. It can suggest user stories, prioritize features based on predicted impact, and estimate project timelines/resources with higher accuracy by learning from historical project data. This leads to more comprehensive and well-defined requirements.

- **AI-Enhanced Design & Architecture (Process 2.0):**
  - **Traditional:** Manual design of system architecture, databases, and user interfaces.
  - **AI Enhancement:** Based on the refined requirements and existing codebases, AI can propose optimal architectural patterns (e.g., microservices, monolithic), design database schemas, suggest API designs, and even generate UI mockups. It can analyze the implications of different design choices on scalability, performance, and security, guiding architects to make informed decisions.

- **AI-Driven Code Generation & Development (Process 3.0):**
  - **Traditional:** Developers write all code manually.
  - **AI Enhancement:** This is one of the most impactful areas.
    - **Code Suggestion & Completion:** AI-powered IDEs offer highly intelligent code suggestions, context-aware completions, and even entire function suggestions based on comments or partial code.
    - **Automated Code Generation:** AI can generate boilerplate code, entire components, or even full applications from high-level specifications or design models. This significantly reduces repetitive coding tasks.
    - **Code Refactoring & Optimization:** AI analyzes code for anti-patterns, performance bottlenecks, security vulnerabilities, and code smells, suggesting or automatically applying refactorings and optimizations.
    - **Automated Bug Fixing:** For certain types of bugs, AI can even propose or directly implement fixes based on learned patterns from past bug resolutions.

- **AI-Powered Testing & Quality Assurance (Process 4.0):**
  - **Traditional:** Manual test case creation, execution, and defect identification.
  - **AI Enhancement:**
    - **Test Case Generation:** AI can automatically generate comprehensive test cases (unit, integration, end-to-end) by analyzing code, requirements, and user behavior patterns.
    - **Intelligent Test Prioritization:** AI can identify high-risk areas of the code or functionalities most likely to break, prioritizing test execution.
    - **Predictive Defect Identification:** By analyzing historical bug data, code complexity, and development activity, AI can predict where defects are most likely to occur, allowing for proactive testing.
    - **Automated UI Testing:** AI can learn application UI elements and generate robust UI tests that adapt to minor UI changes, reducing test maintenance.

- **AI-Enabled Deployment & Operations (Process 5.0):**
  - **Traditional:** Manual deployment processes, reactive monitoring.
  - **AI Enhancement:**
    - **Smart Deployment Strategies:** AI can recommend optimal deployment times and strategies based on predicted system load and potential risks.
    - **Proactive Monitoring & Anomaly Detection:** AI continuously analyzes runtime logs and metrics to detect anomalies, predict potential failures before they occur, and alert operations teams.
    - **Automated Incident Response:** For common issues, AI can trigger automated remediation actions, reducing downtime.
    - **Performance Optimization Suggestions:** AI can analyze application performance data and suggest infrastructure adjustments or code changes to optimize resource utilization.



Solution Requirements (Functional & Non-functional)

Date	25 JUNE 2025
Team ID	LTVIP2025TMID34717
Project Name	Smartsdlc – ai-enhanced software development lifecycle
Maximum Marks	4 Marks

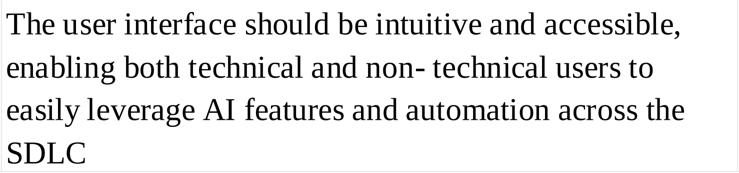
Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP

Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	 The user interface should be intuitive and accessible, enabling both technical and non- technical users to easily leverage AI features and automation across the SDLC
NFR-2	**Security**	Sensitive project data and user information is be protected using industry-standard encryption (e.g., AES- 256), secure authentication, and compliance with relevant standards (like GDPR, HIPAA, or PCI DSS)
NFR-3	**Performance**	The system must deliver real-time insights and automation, ensuring fast response times even under high workloads. For example, generating requirements or test cases using AI should not exceed a few seconds per request
NFR-4	**Availability**	The platform must guarantee high uptime (e.g., 99.99%) and robust failover mechanisms so teams can depend on it for critical development tasks. Automated recovery from failures is essential
NFR-6	**Scalability**	The architecture must handle increasing numbers of users, projects, and data volume without degrading performance. It should support seamless scaling—both vertically and horizontally—during peak demand

Phase IV

Project Planning

Project Planning (Product Backlog, Sprint Planning, Stories, Story points)

Date	25 june 2025
Team ID	LTVIP2025TMID34717
Project Name	Smartsdlc – ai-enhanced software development lifecycle
Maximum Marks	5 Marks

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	2
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	1
Sprint-2		USN-3	As a user, I can register for the application through Facebook	2	Low	3
Sprint-1		USN-4	As a user, I can register for the application through Gmail	2	Medium	2
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password	1	High	1

Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	1 June 2025	5 June 2025	20	5 June 2025
Sprint-2	20	6 Days	6 June 2025	11 June 2025	18	7 June 2025
Sprint-3	20	6 Days	12 June 2025	17 June 2025	20	14 June 2025
Sprint-4	20	6 Days	18 June 2025	23 June 2025	15	23 June 2025

PHASE-V

Functional & Performance Testing

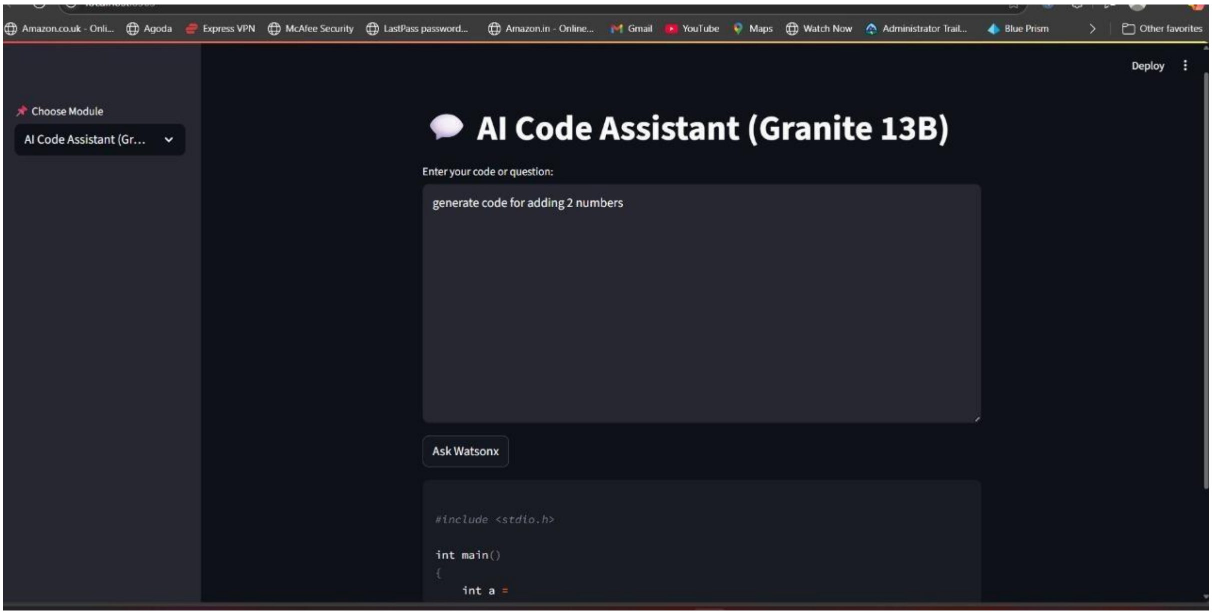
Model Performance Test

Date	25 june 2025
Team ID	LTVIP2025TMID34717
Project Name	smartsdlc – ai-enhanced software development lifecycle
Maximum Marks	

Test Scenarios & Results

Test Case ID	Scenario (What to test)	Test Steps (How to test)	Expected Result	Actual Result	Pass/Fail
FT-01	Text Input Validation (e.g., topic, job title)	Enter valid and invalid text in input fields	Valid inputs accepted, errors for invalid inputs		Pass
FT-02	Number Input Validation (e.g., word count, size, rooms)	Enter numbers within and outside the valid range	Accepts valid values, shows error for out-of-range		Pass
FT-03	Content Generation (e.g., blog, resume, design idea)	Provide complete inputs and click "Generate"	Correct content is generated based on input		Pass
FT-04	API Connection Check	Check if API key is correct and model responds	API responds successfully		Pass
PT-01	Response Time Test	Use a timer to check content generation time	Should be under 3 seconds		Pass
PT-02	API Speed Test	Send multiple API calls at the same time	API should not slow down		Pass
PT-03	File Upload Load Test (e.g., PDFs)	Upload multiple PDFs and check processing	Should work smoothly without crashing		Fail

OUTPUT&SCREEN SHOTS



📌 Choose Module

AI Code Assistant (Gr...

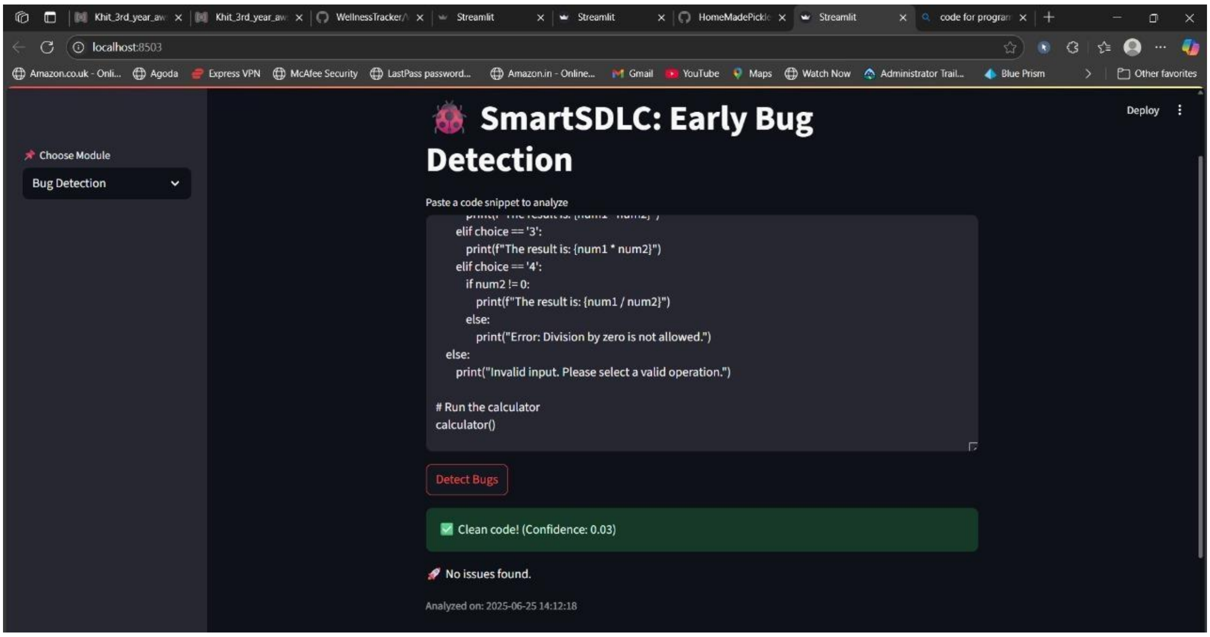
💬 AI Code Assistant (Granite 13B)

Enter your code or question:

generate code for adding 2 numbers

```
#include <stdio.h>

int main()
{
    int a =
```



📌 Choose Module

Bug Detection

🐞 SmartSDLC: Early Bug Detection

Paste a code snippet to analyze

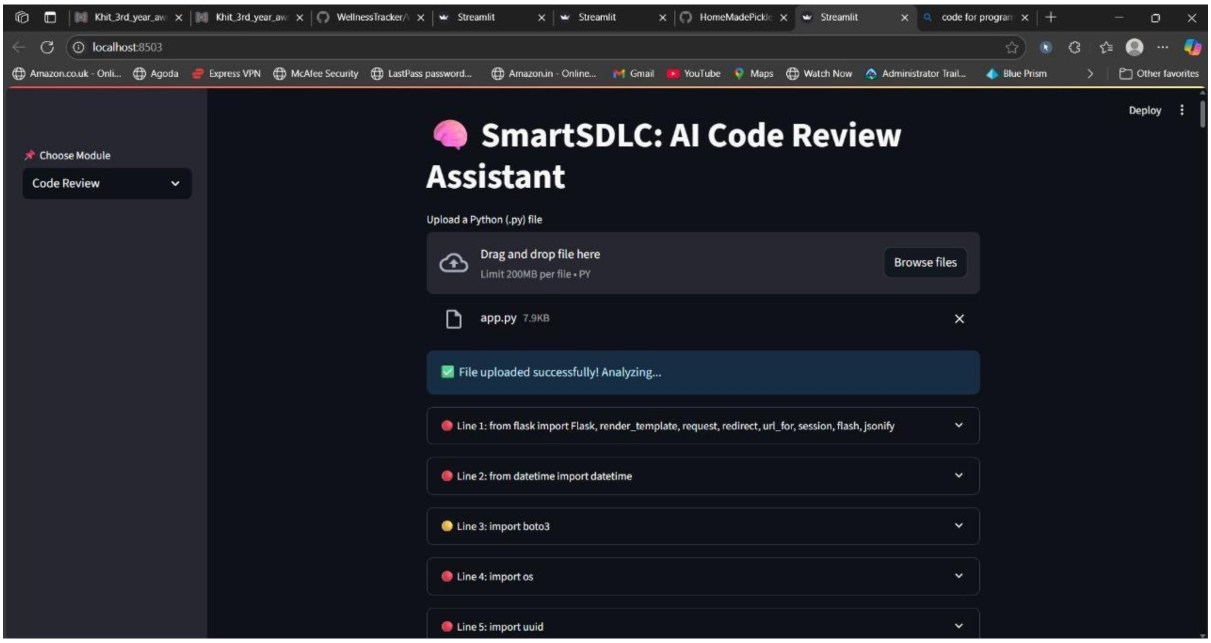
```
      print(f"The result is: {num1 - num2}")
    elif choice == '3':
      print(f"The result is: {num1 * num2}")
    elif choice == '4':
      if num2 != 0:
        print(f"The result is: {num1 / num2}")
      else:
        print("Error: Division by zero is not allowed.")
  else:
    print("Invalid input. Please select a valid operation.")

# Run the calculator
calculator()
```

✅ Clean code! (Confidence: 0.03)

🚀 No issues found.

Analyzed on: 2025-06-25 14:12:18



📌 Choose Module

Code Review

🧠 SmartSDLC: AI Code Review Assistant

Upload a Python (.py) file

Drag and drop file here
Limit 200MB per file • PY

Browse files

app.py 7.9KB

✅ File uploaded successfully! Analyzing...

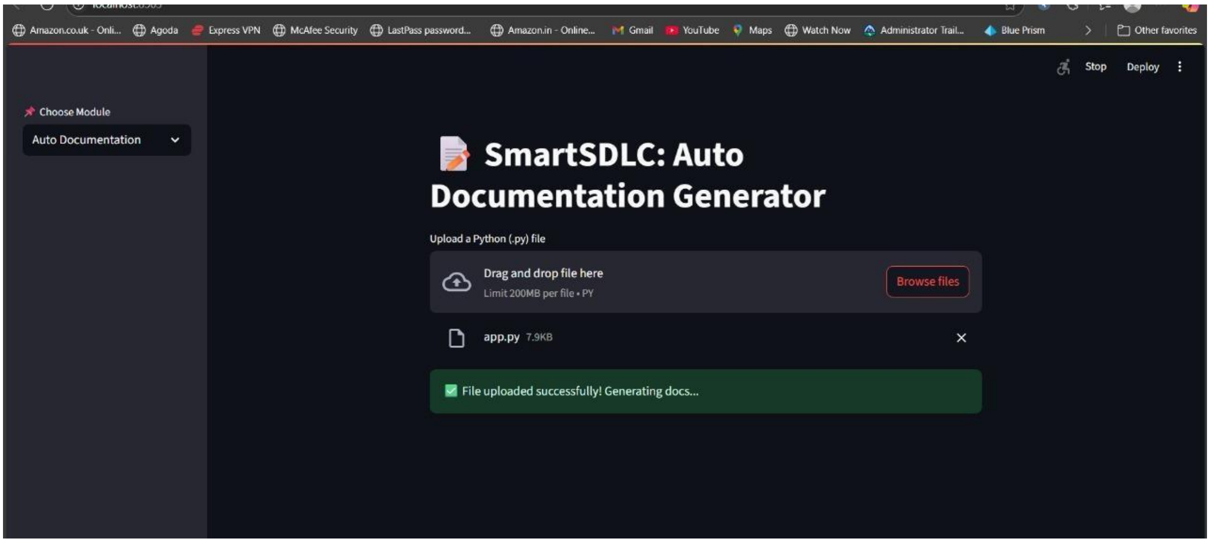
🔴 Line 1: from flask import Flask, render_template, request, redirect, url_for, session, flash, jsonify

🔴 Line 2: from datetime import datetime

🟡 Line 3: import boto3

🔴 Line 4: import os

🔴 Line 5: import uuid



📌 Choose Module

Auto Documentation

📝 SmartSDLC: Auto Documentation Generator

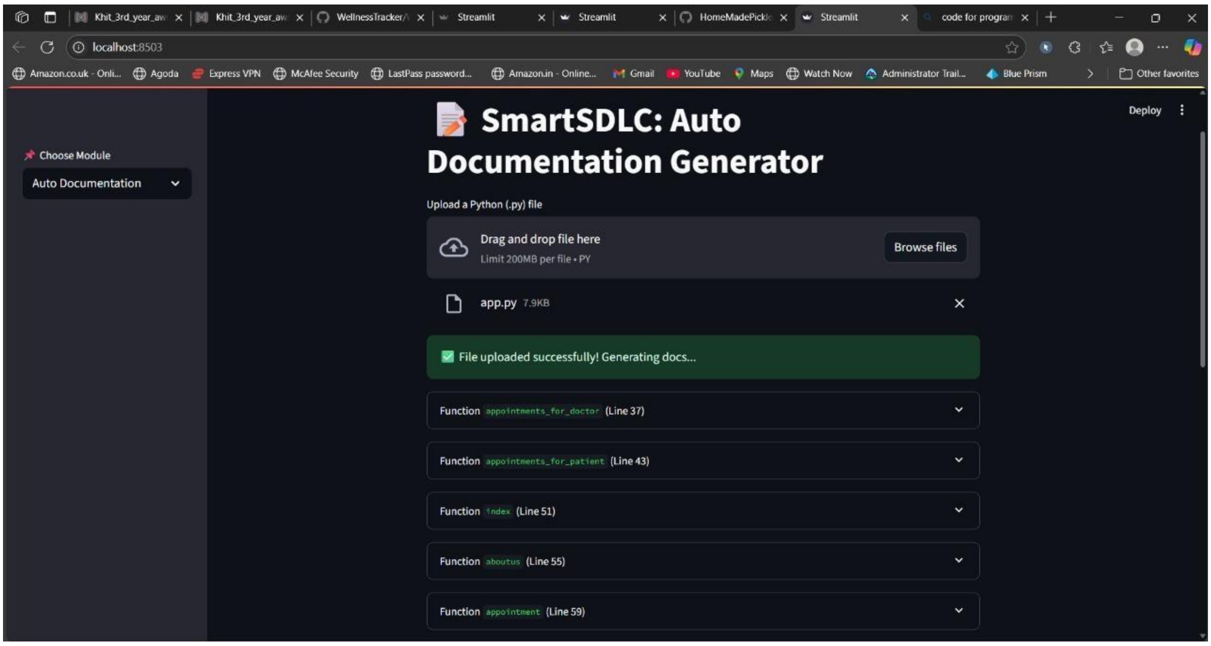
Upload a Python (.py) file

Drag and drop file here
Limit 200MB per file • PY

Browse files

app.py 7.9KB

✅ File uploaded successfully! Generating docs...



📝 SmartSDLC: Auto Documentation Generator

Upload a Python (.py) file

app.py 7.9KB

☑ File uploaded successfully! Generating docs...

Function `appointments_for_doctor` (Line 37)

Function `appointments_for_patient` (Line 43)

Function `index` (Line 51)

Function `aboutus` (Line 55)

Function `appointment` (Line 59)