

**A**  
**PROJECT REPORT ON**  
**GAME DEVELOPER AI**  
**Designed and Developed By**  
**PIYUSH CHALL**  
**Seat No:-**  
**UNDER THE GUIDANCE OF**  
**ASS. PROF. GANESHKUMAR R. NARAYANKAR**  
**SUBMITTED IN PARTIAL FULFILMENT OF ACADEMIC**  
**PROJECT**  
**BACHELOR OF SCIENCE IN COMPUTER SCIENCE**



**UNIVERSITY OF MUMBAI**  
  
**SHREE SHANKAR NARAYAN COLLEGE OF ARTS AND**  
**COMMERCE**  
  
**BACHELOR OF SCIENCE IN COMPUTER SCIENCE**  
  
**ACADEMIC YEAR 2024-25**



Shree Shankar Narayan Education Trust's

# SHANKAR NARAYAN COLLEGE OF ARTS & COMMERCE

B.M.S., B.Sc.I.T., B.C.S., B.B.I., B.A.F., B.F.M., M.Sc.I.T., M.Com.

Navghar, Mahavidyalaya Marg, Bhayandar (E), Thane - 401 105. (Maharashtra State),

(Affiliated to the University of Mumbai)

**NAAC Accredited 'B++'**

Prin. Dr. V. N. Yadav M.Com., M.Phil, Ph.D.

**T-35**

Tel. : 2804 65 64, 2804 82 35

Website - www.sncollege.com

E-mail : info@sncollege.com • Fax : 2804 0966

Ref. No.

Date \_\_\_\_\_

## CERTIFICATE

This is to Certify that Mr./Miss \_\_\_\_\_

\_\_\_\_\_ Studying in T.Y. \_\_\_\_\_

(Semester ) in our college having Roll No. \_\_\_\_\_ and Exam

Seat No. \_\_\_\_\_ has successfully completed the project on

\_\_\_\_\_ in the academic Year 20 - 20

Date : / /

Project Guide

Co-ordinator

Principal

Internal Examiner

External Examiner

## **ACKNOWLEDGEMENT**

Firstly, I would like to express my sincere thanks to Principal **Dr. V. N. Yadav** and the Head of the Department **Prof. Vaishali Kadam** of Shankar Narayan college for giving us the opportunity to accomplish this project work.

I would also like to thank **Prof. Ganesh R. Narayankar** for his guidance, support, and valuable insights throughout the project. Their expertise and continuous encouragement have been instrumental in shaping the direction of my project and ensuring its successful execution.

We also thank the faculty and staff of **SHANKAR NARAYAN COLLEGE** specifically the Department of Computer Science, for providing me with the necessary resources and facilities to carry out this project. Their commitment to academic excellence and dedication to fostering a conducive learning environment have been invaluable.

Furthermore, I am grateful to my classmates and friends who provided me with their unwavering support and encouragement throughout this journey. Their discussions, feedback, and collaborative efforts have greatly enriched my understanding and perspective on the subject matter.

Finally, this project would not have been possible without the collective efforts of these individuals and institutions. I am truly grateful for their contributions and the impact they have had on my academic journey

**Project Associate**  
**“PIYUSH CHALL”**

## **ABSTRACT**

In the competitive gaming landscape, rapid and efficient game development is paramount. To stay ahead, companies must embrace innovation. AI offers a transformative solution. By deploying a team of specialized AI agents, we can streamline the entire game development process. This collaborative approach, powered by the CrewAI framework, enables rapid brainstorming, accurate coding, and efficient bug testing. Our AI game-builder revolutionizes game creation, ensuring robust, creative, and scalable production. By automating critical tasks and optimizing workflows, we significantly reduce development time and resources, setting new standards for innovation in the gaming industry.

# PLAGIARISM REPORT



## Content Checked for Plagiarism

In the fast-paced and ever-growing gaming industry, the quick and smooth release of games is essential for a company's success and sustainability. With rising competition, the ability to innovate swiftly and efficiently becomes crucial. Integrating AI heralds, a new era in gaming innovation, offering unique opportunities for creativity and productivity. The project capitalizes on this potential by deploying a team of multiple AI agents, each with specialized skills to streamline the game development process. This collaborative approach ensures rapid brainstorming, accurate coding, and thorough bug detection and resolution. This initiative focuses on a game-builder AI that leverages the CrewAI framework, representing an innovative method in game development. By utilizing the diverse capabilities of these AI agents, the project aims to transform game creation, ensuring robust, creative, and scalable production processes. The CrewAI framework promotes smooth collaboration among agents, enabling them to generate innovative game concepts, convert these ideas into executable code, and rigorously test the final product for quality and performance. The AI-driven strategy automates and optimizes critical aspects of game development, significantly cutting down the time and resources needed to take a game from concept to launch. The AI game-builder establishes new standards for innovation and efficiency in the gaming industry, ensuring robust, creative, and scalable game production processes.

## **ACTIVITY SHEET**

**Name:-** Piyush Chall

**Class:-** TYCS

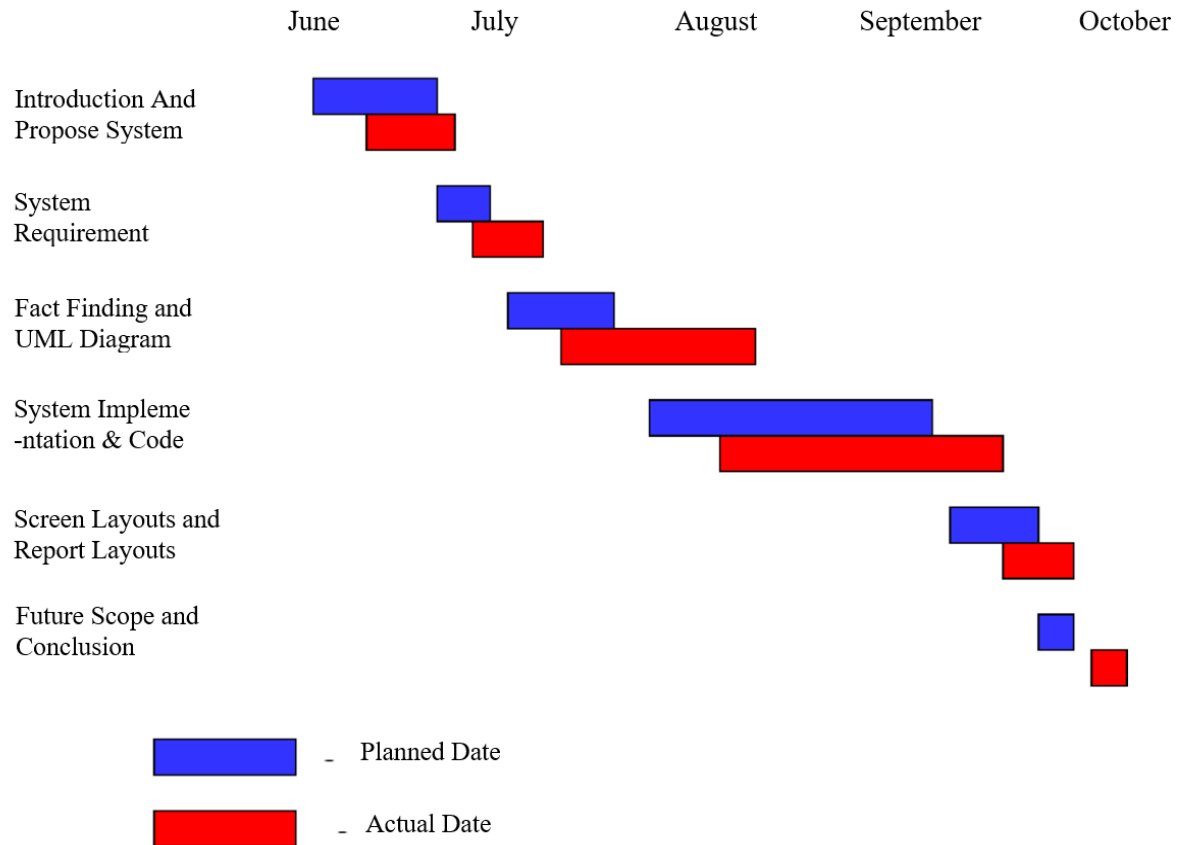
**Roll No. :-** 04

**Seat No. :-**

**Project Name:-** Game Developer AI

<b>Sr. No.</b>	<b>Content</b>	<b>Planned Date</b>		<b>Actual Date</b>		<b>Sign</b>
		<b>Start Date</b>	<b>End Date</b>	<b>Start Date</b>	<b>End Date</b>	
1	Introduction and Proposed System	20-06-24	28-06-24	21-06-24	30-06-24	
2	System Requirement	01-07-24	10-07-24	03-07-24	11-07-24	
3	Fact Finding and UML Diagrams	11-07-24	29-07-24	12-07-24	30-07-24	
4	System Implementation and Coding	01-08-24	09-09-24	02-08-24	10-09-24	
5	Screen layout and Reports	09-09-24	28-09-24	11-08-24	30-09-24	
6	Future Scope and Conclusion	01-10-24	07-10-24	03-10-24	08-10-24	

## GANTT CHART



## INDEX

<b>SR.NO.</b>	<b>CONTENTS</b>	<b>PAGE NO.</b>	<b>SIGN</b>
<b>1</b>	<b>PRELIMINARY INVESTIGATION</b>		
	1.1 Introduction	1	
	1.2 Project Objectives	2	
	1.3 Motivation	3	
	1.4 Propose System	4-5	
	1.5 Features	6	
	1.6 Advantages and Disadvantages	7	
	1.7 Fact Finding Q&A	8	
<b>2</b>	<b>REQUIREMENTS AND STAKEHOLDERS</b>		
	2.1 Hardware Requirements	9	
	2.2 Software Requirement	10-11	
<b>3</b>	<b>PROTOTYPE</b>		
	3.1 Prototype	12	
	3.2 Stakeholder	13	
	3.1 Event Table	14-15	
	3.2 Use Case Diagram	16-17	
	3.3 Class Diagram	18-19	
	3.5 Data Flow Diagram	20-21	
	3.6 Package Diagram	22-23	
	3.8 Activity Diagram	24-25	
<b>4</b>	<b>SYSTEM DESIGN AND CODING</b>		
	4.1 Design Layout	26-28	
	4.2 Main module code	29	
<b>5</b>	<b>SCREENSHOTS AND REPORT LAYOUT</b>	30-40	
<b>6</b>	<b>CONCLUSION</b>	41	
<b>7</b>	<b>FUTURE ENHANCEMENT</b>	42	
<b>8</b>	<b>REFERENCE AND BIBLIOGRAPHY</b>		
	8.1 Books	43	
	8.2 Videos	44	
	8.3 Web Pages	45	



## **PRELIMINARY INVESTIGATION**

## 1.1 Introduction

In the fast-paced and ever-growing gaming industry, the quick and smooth release of games is essential for a company's success and sustainability. With rising competition, the ability to innovate swiftly and efficiently becomes crucial. Integrating AI heralds, a new era in gaming innovation, offering unique opportunities for creativity and productivity. The project capitalizes on this potential by deploying a team of **multiple AI agents**, each with specialized skills to streamline the game development process. This collaborative approach ensures rapid brainstorming, accurate coding, and thorough bug detection and resolution.

This initiative focuses on a game-builder AI that leverages the **CrewAI** framework, representing an innovative method in game development. By utilizing the diverse capabilities of these AI agents, the project aims to transform game creation, ensuring robust, creative, and scalable production processes. The CrewAI framework promotes smooth collaboration among agents, enabling them to generate innovative game concepts, convert these ideas into executable code, and rigorously test the final product for quality and performance.

The AI-driven strategy automates and optimizes critical aspects of game development, significantly cutting down the time and resources needed to take a game from concept to launch. The AI game-builder establishes new standards for innovation and efficiency in the gaming industry, ensuring robust, creative, and scalable game production processes.

## 1.2 Project Objective

The primary goal of this project is to deploy an AI crew capable of autonomously developing a game. This crew will handle tasks such as brainstorming, programming, and code evaluation, with the **ability to autonomously fix bugs**.

This objective will be achieved through:

1. Creating multiple agents with specialized skills.
2. Efficient allocation of tasks among the agents.
3. Successful integration of the AI crew to collaborate seamlessly in game development.

## 1.3 Motivation

In today's gaming world, there is a growing demand for rapid ideation and programming. Drawing from prior experience in game development and current insights into AI, integrating AI into game programming will significantly enhance productivity for game developers globally. This integration promises to streamline game production processes, reducing development timelines. Motivated by these possibilities, I aim to assemble a team of multiple AI agents capable of efficiently creating games with minimal human intervention.

## 1.4 Propose System

### Data Acquisition:

- Input: The game's name will be provided to the AI team via text input,
- Input: The fundamental mechanics necessary for code implementation will be conveyed to the team through text input, playing a crucial role in the development process.

### Agent Implementation:

- We will implement a multimodal AI Agent system via the **CrewAI Framework** in Python.
- The Agents would be:
  1. Designer: This agent is tasked with brainstorming and researching various game mechanisms for implementation in the game.
  2. Programmer: This agent is responsible for coding and bringing the game idea to life.
  3. QA Analyst: This agent is responsible for identifying and fixing bugs in the game and evaluating the game's code.
  4. Manager: This agent would be responsible for assigning all the tasks to their respective Agent.

### Task Implementation:

- The multimodal AI Agent System necessitates the implementation of diverse tasks and goals that the agents must fulfill to achieve the outcome.
- The task would be:
  1. Brainstorming
  2. Programming
  3. Evaluating

### Crew collaboration:

- Here, we will leverage CrewAI's built-in capability to integrate all agents and their tasks into a single Python code module.

## 1.5 Features

- **Brainstorming Capabilities:** Generates and explores creative game ideas.
- **Programming Automation:** Translates game concepts into executable code.
- **Bug Detection and Fixing:** Identifies and resolves programming errors autonomously.
- **Task Allocation:** Efficiently assigns tasks to different AI agents based on their specialized skills.
- **Integration of Game Mechanics:** Incorporates various game mechanisms into the design.
- **Rapid Prototyping:** Quickly develops game prototypes to visualize ideas.
- **Scalability:** Adapts to varying project sizes and complexities.
- **Collaborative Development:** Facilitates seamless collaboration among multiple AI agents.

## 1.6 Advantages and Disadvantages

Advantages include:

- **Efficiency:** Accelerates game development processes, reducing time-to-market.
- **Accuracy:** Minimizes human error in coding and bug-fixing tasks.
- **Innovation:** Facilitates rapid prototyping and experimentation with new game ideas.
- **Cost-Effectiveness:** Lowers development costs through automation and streamlined processes.
- **Flexibility:** Adapts to changing requirements and integrates new technologies seamlessly.
- **Dynamic LLM integrity:** Multiple LLMs with callbacks can be used dynamically in the Agent's mind.

Disadvantages could include:

- **Dependency on Data Quality:** Relies heavily on the quality and relevance of input data for effective decision-making and performance.
- **Limited Creativity:** May need help with generating truly innovative or out-of-the-box game ideas compared to human creativity.
- **LLM API cost:** Some LLM integrations like the openAI's GPT-4o, and GPT-4 can increase the cost of the AI.
- **Performance Limitations:** Depending on the AI's capabilities, there may be limitations in handling complex game mechanics or real-time interactions effectively.
- **Risk of Over-Automation:** This could lead to a reduction in human involvement and creativity in game development processes.



## 1.7 Fact-finding and Question-Answers

Game Builder AI involves systematically integrating multiple AI agents to rapidly develop games effectively. Here are the key areas for fact-finding in the project:

- **How is AI useful in the gaming industry?**

Ans: Facilitates rapid prototyping and experimentation with new game ideas. Lowers development costs through automation and streamlined processes. Accelerates game development processes, reducing time-to-market.

- **What potential challenges would the end-user face while using this AI?**

Ans: This AI can only offer a limited extent of creativity and innovation. The AI requires a lot of computational power for its execution of tasks which can limit the production phase of the game if a high-end device is not used.

- **Is this AI game-building Agent system easy to use?**

Ans: This AI can easily create very understandable code for the game. It has very low human intervention. Comprehensive documentation and support resources are available to assist users, while the system's flexible input options make it adaptable to various needs and preferences.

- **What game development tools and techniques will the Agent system use?**

Ans: The AI system is capable of researching any game-related topic. Its dynamic and vast knowledge of the gaming field equips it with the intelligence to accurately determine which tools and techniques to use for game production.

- **How will the performance of the AI be evaluated?**

Ans: The skills and performance of this AI Agent can be evaluated through verifying the creative decisions made by the AI. Additionally, by using and examining the AgentOP dashboard, which I have already integrated into the main Crew file, users can gain insights into the Agent System's operations and identify any errors made by the AI.

## **REQUIREMENTS**

## 2.1 Hardware Requirements

1. Processor:

- Multi-core processor (Quad-core or higher recommended)

2. RAM:

- Minimum 4GB RAM

3. SSD:

- SSD with at least 2GB of storage for fast data access and read/write speeds

4. GPU:

- Dedicated GPU with at least 8GB VRAM

5. Network:

- High-speed internet connection for data transfer and cloud-based operations

## 2.2 Software Requirements

### 1. Operating System:

- Windows 10/11 (64-bit)
- macOS (latest version)
- Linux (Ubuntu 20.04 LTS or later)

### 2. Development Environment:

- Integrated Development Environment (IDE) such as PyCharm, Visual Studio Code, and Replit.

### 3. Programming Language:

- Python 3.11 or later as per the requirements of CrewAI.

### 4. Framework:

- CrewAI Framework for Agent Integration.

### 5. Additional Packages:

- Python's Textwrap package for customizing strings is used in the prompts of the agents. (Optional)
- PyGame library required for the Agent's tasks.
- Agentops library for verification and validation of Agents. (Optional)
- Langchain\_google\_genai for using gemini model.

6. Version Control:

- GitHub

7. Additional tools:

- SerperDevTool for rapid researching of agents.
- WebsiteSearchTool for detailed searching over the internet.
- Prettier extension for easy code visualization. (Optional)

8. Required APIs:

- GOOGLE\_API\_KEY required for gemini model integration.
- SERPER\_API\_KEY required for SerperDevTool integration.

**PROTOTYPE**

## 3.1 Prototype

- Python

Python is a high-level, general-purpose programming language known for its readability and ease of use. It was created by Guido van Rossum in the late 1980s and released in 1991. Python's design philosophy emphasizes code readability and simplicity, making it a popular choice for both beginners and experienced programmers.

- CrewAI

CrewAI is a powerful framework designed to facilitate collaboration between multiple AI agents. It provides a structured environment where agents can interact, share information, and work together towards common goals. By leveraging the collective intelligence of multiple agents, CrewAI can tackle complex tasks that would be challenging or impossible for a single agent to accomplish.

CrewAI also provides mechanisms for agents to communicate and share information. This enables them to coordinate their actions, avoid conflicts, and make informed decisions. Through effective communication, agents can collaborate on tasks, such as planning, problem-solving, and decision-making.

- Gemini

**Gemini** is a large language model developed by Google AI. It's designed to be able to communicate and generate human-like text in response to a wide range of prompts and questions. Gemini is trained on a massive dataset of text and code, allowing it to perform tasks such as summarizing text, translating languages, writing different kinds of creative content, and answering your questions in an informative way.

Gemini is still under development, but it has already shown impressive capabilities. In the future, Gemini could be used for a variety of applications, such as customer service, education, and research.

## 3.2 Stakeholder

- Stakeholder

Stakeholder refers to any individual, group, or organization that has a vested interest in a project, business, or other endeavor. They can be directly or indirectly involved, and their interests can range from financial gain to social impact.

Common types of stakeholders include:

- **Internal stakeholders:** Employees, managers, shareholders, and board members.
- **External stakeholders:** Customers, suppliers, investors, government agencies, and the community.

- User/Customer

An individual interacts with the AI system to create Games.



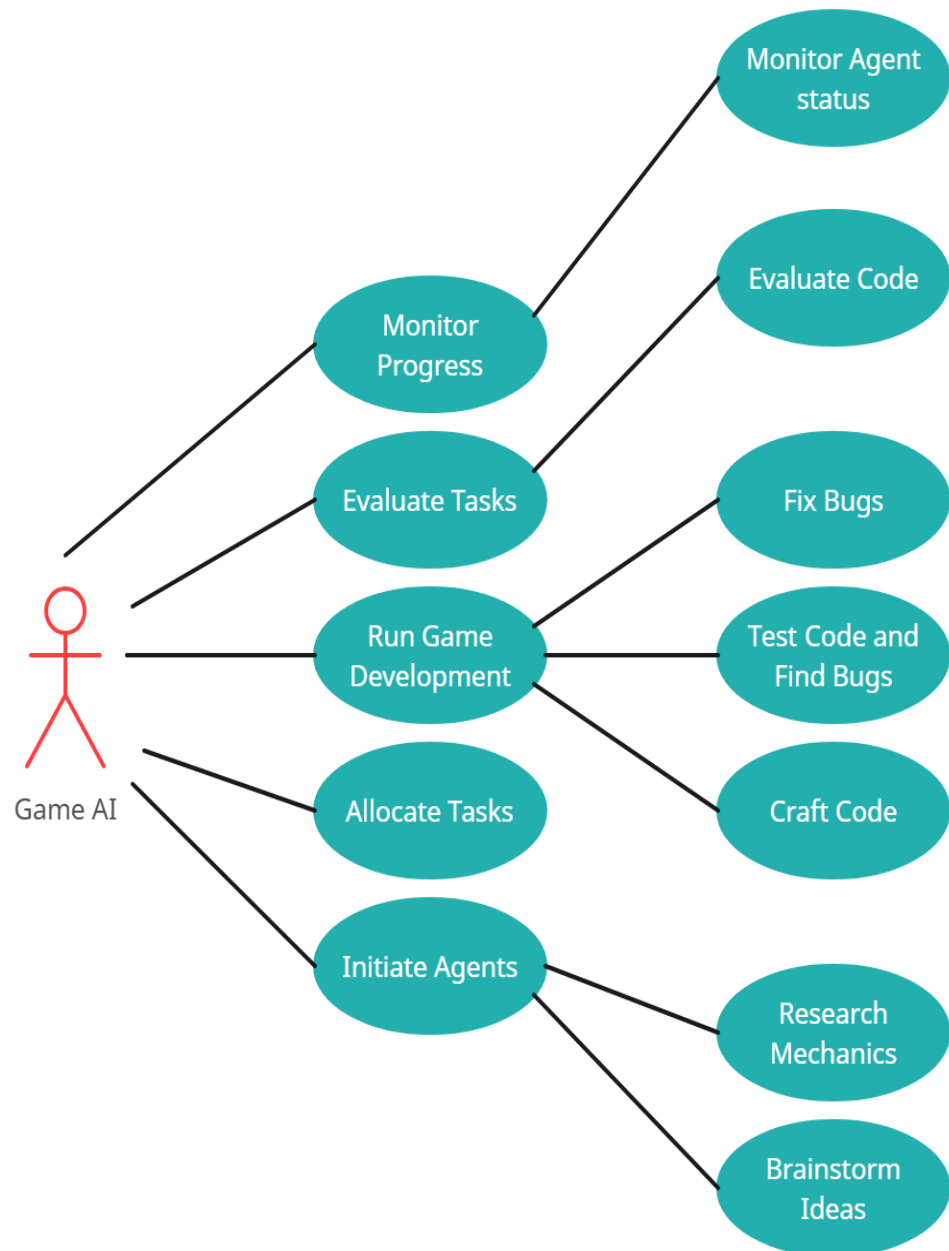
### 3.4 Event Table

Event	Trigger	Source	Description	Response	Destination
Project execution	API is called.	User	End user the AI's API is called directly or indirectly.	The AI is initiated.	User
Name Input	The user submits the name of the game.	User	The end user enters the name or description of the game.	The Name of the game is submitted to the AI.	User
Task Allocation	Allocates tasks to each AI agent.	System	The AI dedicates tasks to its co-worker AI agents.	All development tasks have been allocated.	System
Brainstorming	AI starts Brainstorming ideas.	System	Based on the given name of the game the AI crafts several ideas for the game.	A list of Ideas is generated for the game, and one is selected.	System
Programming	AI generates code.	System	A raw code is crafted by the AI for the selected Idea of the game.	A raw code for the game.	System
Bug hunt and Fixes	AI searches for bugs and fixes them.	System	The AI reviews and hunts for bugs in the developed game and tries to fix them.	Validated game code.	System
Running the Game.	The user runs the game.	User	The user runs the game developed by the AI.	Ussr plates the generated game.	User

Agent Monitoring	AI checks Agent status	System	Manager AI monitors the task execution of other AI agents.	System issues are identified and resolved.	System
------------------	------------------------	--------	--	--	--------

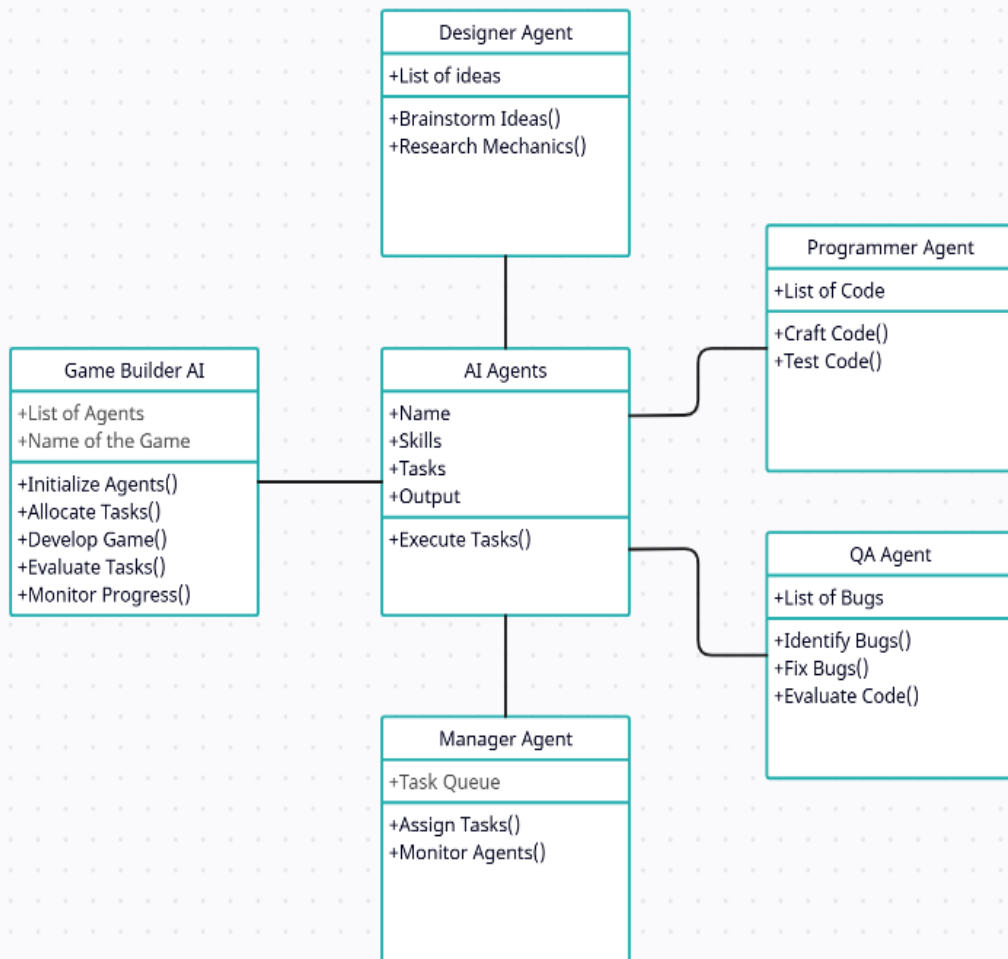
### 3.5 Use case Diagram

Use case diagrams are a valuable tool in software engineering and system analysis. They visually depict the interactions between a system and its external actors (users or other systems). These diagrams are particularly useful for understanding the functional requirements of a system before development begins.



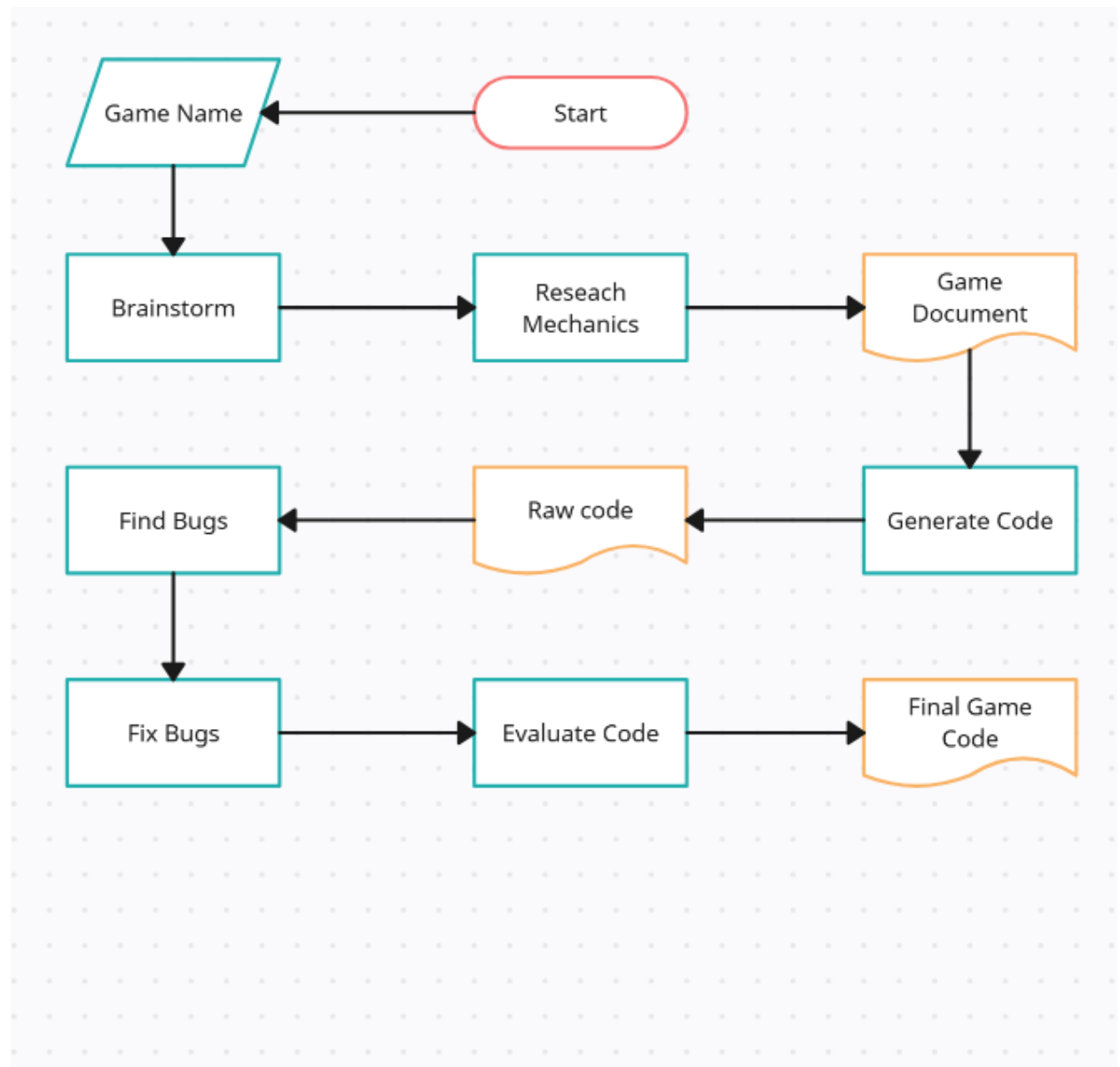
### 3.6 Class diagram

Class diagrams are a fundamental tool in object-oriented analysis and design (OOAD). They provide a graphical representation of the classes, attributes, operations, and relationships between them in a system.



### 3.7 Dataflow diagram

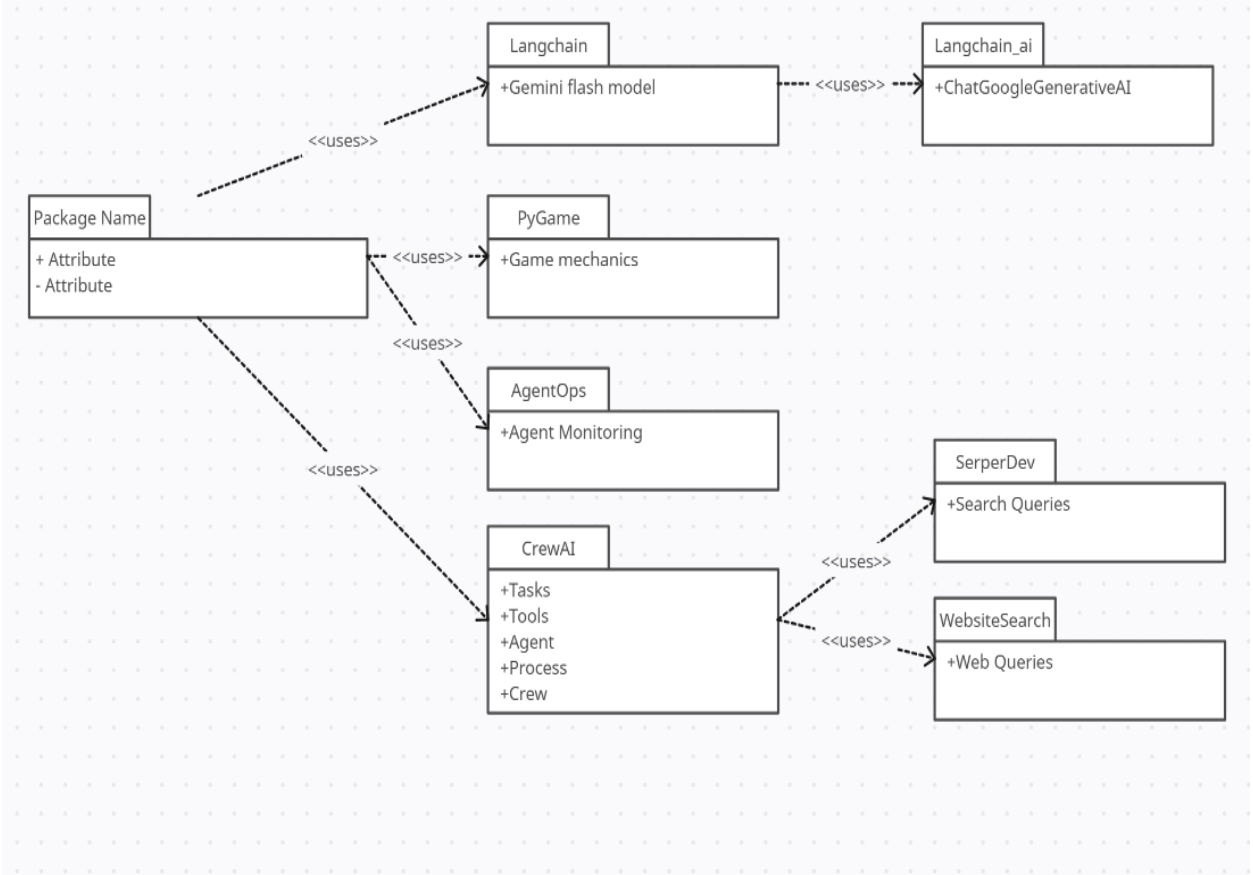
Dataflow diagrams (DFDs) are graphical tools used to represent the flow of data through a system. They are particularly useful for understanding the functional requirements of a system, especially in systems analysis and design.





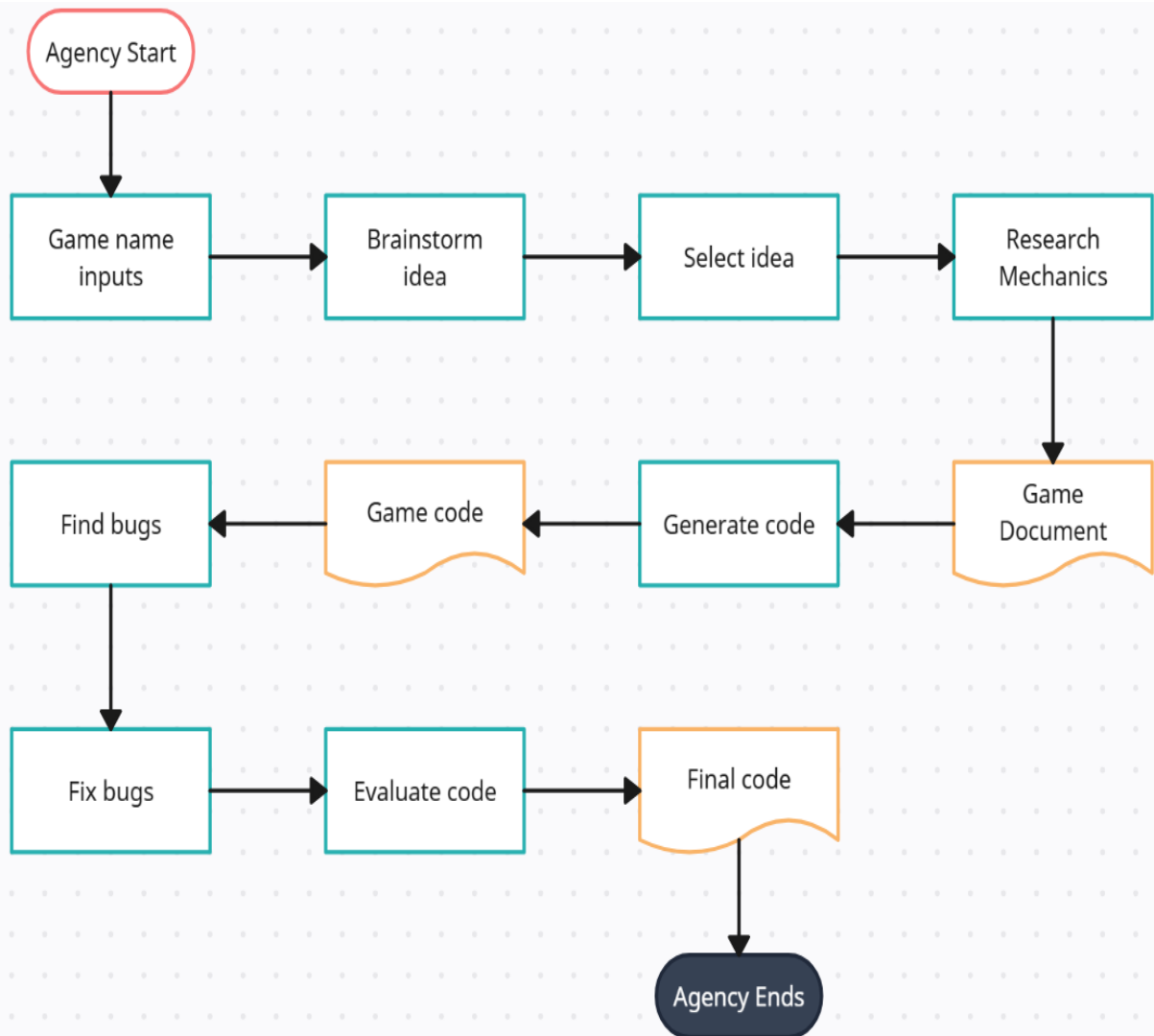
## 3.8 Package diagram

Package diagrams are used to model the high-level structure of a system by grouping related elements (such as classes, interfaces, and components) into packages. They are particularly useful for understanding the overall organization of a large-scale system.



### 3.9 Activity diagram

Activity diagrams are used to model the flow of activities within a system. They are particularly useful for understanding the dynamics of a system's processes and workflows.



## **SYSTEM DESIGN AND CODING**

## 4.1 Design Layout

My application runs entirely on the command line, thus I have used various effective layout design. While I won't be working with a traditional visual interface, I have applied many principles to create a user-friendly and efficient command-line experience.

- Task execution Highlighting
- Crew Kick off separation
- Easy to read Thought process of the AI
- Model waiting for Feedbacks
- Easy input system
- Dedicated execution behavior

For proper handling of various exceptions, the AI system has been integrated by AgentOps which helps to effectively monitor Agent execution and provide an extremely friendly and resource extensive command line environment.

Here you can see an easy input system.

```
~/AI-agent$ cd Game_Agent
~/AI-agent/Game_Agent$ python crew.py
pygame 2.5.2 (SDL 2.28.2, Python 3.10.14)
Hello from the pygame community. https://www.pygame.org/contribute.html
What is the name of the game you want to create? (e.g. Pong, 2048, Tetris, etc):
```

Here we have given “Tetris” as input

```
~/AI-agent$ cd Game_Agent
~/AI-agent/Game_Agent$ python crew.py
pygame 2.5.2 (SDL 2.28.2, Python 3.10.14)
Hello from the pygame community. https://www.pygame.org/contribute.html
What is the name of the game you want to create? (e.g. Pong, 2048, Tetris, etc):
tetris
§ AgentOps: Session Replay: https://app.agentops.ai/drilldown?session_id=9aa7818f-cc7e-4730-a684-8f9532e76546
2024-10-08 15:46:32,784 - 140534351140800 - client.py-client:366 - INFO: Session
Replay: https://app.agentops.ai/drilldown?session_id=9aa7818f-cc7e-4730-a684-8f9532e76546
```

And the AgentOps utility automatically instantiates and begins to monitor the AI Agent behavior

Here you can see the agent system asks for feed back from the user to check if the agents idea is validated with the user's requirements.

#### **\*\*7. Development Plan\*\***

- \* **\*\*Phase 1:\*\*** Prototype development focusing on core gameplay mechanics, block manipulation, and line clearing.
- \* **\*\*Phase 2:\*\*** Implementation of Fusion Blocks, visual design, and sound effects.
- \* **\*\*Phase 3:\*\*** Development of game modes, user interface, and multiplayer functionality.
- \* **\*\*Phase 4:\*\*** Testing, bug fixing, and optimization.

#### **\*\*8. Target Audience\*\***

- \* Casual gamers
- \* Tetris fans
- \* Players seeking a unique and challenging puzzle experience

#### **\*\*9. Marketing Strategy\*\***

- \* **\*\*Social media promotion:\*\*** Engage with gaming communities on platforms like Reddit, Twitter, and Discord.
- \* **\*\*Game trailers and gameplay videos:\*\*** Showcase the game's unique features and gameplay.
- \* **\*\*App store optimization:\*\*** Use relevant keywords and engaging descriptions to attract players.

This document provides a comprehensive overview of the Tetris Fusion game design. It outlines the core gameplay mechanics, unique elements, game modes, visual and audio design, user interface, technical requirements, and development plan. This information will serve as a foundation for building a captivating and engaging game experience.

captivating and engaging game experience.

Please provide a feedback: good

And after reading the game documentation the user here has provided with “Good” as feedback. Any individual using the AI agent system can provide anything as feedback, even any negative feedback is cross validated by the AI agency.



## 4.2 Main module code

```
import os

import agentops

from agents import chief_developer, qa_engineer, senior_developer
from crewai import Crew, Process
from tasks import brainstorm, evaluation, programming, reviewing, game
from tools import llm

agentop_api_key = os.environ['AGENTOP_API_KEY']
agentops.init(agentop_api_key)

crew = Crew(agents=[senior_developer, qa_engineer, chief_developer],
            tasks=[brainstorm, programming, reviewing, evaluation],
            process=Process.hierarchical,
            manager_llm=llm)

result = crew.kickoff(inputs={"game": game})
print(result)
```

## **SCREENSHOTS AND REPORT LAYOUT**

## 5.1 Screenshot and Report Layout

- Project Initiation

```
~/AI-agent$ cd Game_Agent
~/AI-agent/Game_Agent$ python crew.py
pygame 2.5.2 (SDL 2.28.2, Python 3.10.14)
Hello from the pygame community. https://www.pygame.org/contribute.html
What is the name of the game you want to create? (e.g. Pong, 2048, Tetris, etc):
```

- Inputs

```
~/AI-agent$ cd Game_Agent
~/AI-agent/Game_Agent$ python crew.py
pygame 2.5.2 (SDL 2.28.2, Python 3.10.14)
Hello from the pygame community. https://www.pygame.org/contribute.html
What is the name of the game you want to create? (e.g. Pong, 2048, Tetris, etc):
tetris
§ AgentOps: Session Replay: https://app.agentops.ai/drilldown?session_id=9aa7818f-cc7e-4730-a684-8f9532e76546
2024-10-08 15:46:32,784 - 140534351140800 - client.py-client:366 - INFO: Session
Replay: https://app.agentops.ai/drilldown?session_id=9aa7818f-cc7e-4730-a684-8f9532e76546
```

- Agent Instantiation

> Entering new CrewAgentExecutor chain...

*Thought: I need to brainstorm game ideas based on Tetris and then decide which one is best .*

*Action: Delegate work to co-worker*

*Action Input: {"task": "Brainstorm game ideas based on Tetris. Consider variations on the core gameplay, new mechanics, and different themes. Please provide at least 5 distinct ideas, each with a brief description.", "context": "We are looking to create a new game inspired by Tetris. Focus on ideas that are fresh and engaging, while still retaining the core elements of Tetris. Consider different themes, gameplay variations, and new mechanics that could be incorporated.", "coworker": "Senior Game Developer"}*

- Task allocation

> Entering new CrewAgentExecutor chain...

*Thought: I can definitely come up with some fresh Tetris-inspired game ideas! Let's brainstorm some variations that keep the core gameplay but add exciting twists.*

*Final Answer:*

*Hey team, here are some Tetris-inspired game ideas we can explore:*

*\*\*1. Tetris: Time Warp\*\**

*\* \*\*Theme:\*\* Sci-fi, time travel*

*\* \*\*Gameplay:\*\* Instead of falling blocks, players control a time-traveling device that can manipulate the past. Each level is a timeline, and players can rewind or fast-forward to strategically place blocks in the past, creating advantageous formations in the present*

- Searching over internet

Search results: Title: Tetris game in Python - GitHub Gist

Link: <https://gist.github.com/timurbakibayev/1f683d34487362b0f36280989c80960c>

Snippet: Tetris game in Python. GitHub Gist: instantly share code, notes, and snippets.

---

Title: Creating Tetris in Python with pygame - Beginner Tutorial (OOP)

Link: [https://www.youtube.com/watch?v=nF\\_crEtmpBo](https://www.youtube.com/watch?v=nF_crEtmpBo)

Snippet: Build the classic game Tetris in less than two hours with Python and Pygame! Follow along as ...

---

Title: Python Tetris Game - Develop Tetris using PyGame - DataFlair

Link: <https://data-flair.training/blogs/python-tetris-game-pygame/>

Snippet: Create Tetris Game Project in Python using pygame module to build Tetris game & random module to get shapes of blocks in random manner.

---

Title: Tetris implementation in Python - Discover gists · GitHub

Link: <https://gist.github.com/silvasur/565419/d9de6a84e7da000797ac681976442073045c74a4>

Snippet: game = Tetris(20, 10) counter = 0. pressing\_down = False. while not done: if game.figure is None: game.new\_figure() counter += 1 if counter > 100000: counter ...

---

Title: Tetris in Pygame [Python 3] - Reddit

Link: [https://www.reddit.com/r/pygame/comments/uz1319/tetris\\_in\\_pygame\\_python\\_3/](https://www.reddit.com/r/pygame/comments/uz1319/tetris_in_pygame_python_3/)

Snippet: I am doing fun "pet projects" of coding old games (that i find in tutorials) into pygame, eventually leading up to programming ai to play said games.

---

- Thought process

*Thought: I need to delegate the task of implementing the game modes to the Senior Game Developer.*

*Action: Delegate work to co-worker*

*Action Input: { "coworker": "Senior Game Developer", "task": "Implement the game modes for the Tetris Fusion game.", "context": "We have the core Tetris gameplay with Fusion Blocks implemented. Now, we need to add the different game modes as outlined in the Game Design Document. Here's a breakdown of what needs to be done:\n\n\*\*Classic Mode:\*\* This is the base mode with traditional Tetris gameplay, but with the addition of Fusion Blocks. You'll need to ensure that Fusion Blocks are randomly spawned during gameplay, and their fusion logic works as intended.\n\n\*\*Puzzle Mode:\*\* In this mode, players need to solve specific challenges using Fusion Blocks and Tetris blocks within a limited time or number of moves. You'll need to create a system to generate these challenges, track the player's progress, and provide feedback on their performance.\n\n\*\*Survival Mode:\*\* Players need to survive as long as possible against increasingly difficult waves of blocks. You'll need to implement a system to increase the difficulty over time, potentially by increasing the speed of falling blocks, the frequency of Fusion Blocks, or by introducing new challenges.\n\n\*\*Time Trial Mode:\*\* Players race against the clock to clear as many lines as possible. You'll need to implement a timer, track the player's score, and display the results at the end of the round.\n\n\*\*Fusion Challenge Mode:\*\* Players need to clear a specific number of lines using Fusion Blocks within a time limit. You'll need to implement a system to track the number of lines cleared using Fusion Blocks, a timer, and display the results at the end of the round.\n\n\*\*Key Considerations:\*\*\n\n\*\*User Interface:\*\* Each game mode should have a clear and intuitive user interface that displays relevant information like score, level, time remaining, and challenge objectives.\n\n\*\*Difficulty Scaling:\*\* The difficulty of each game mode should scale appropriately to provide a challenging but engaging experience for players.\n\n\*\*Game State Management:\*\* You'll need to manage the game state for each mode, including things like score, level, time, and challenge progress.\n\n\*\*Game Over Conditions:\*\* Define clear game over conditions for each mode, such as reaching the top of the playing field, running out of time, or failing to complete a challenge.\n\nPlease implement these game modes, ensuring that they are well-designed, balanced, and provide a fun and engaging experience for players." }*

- Exception

*I tried reusing the same input, I must stop using this action input. I'll try something else instead.*

I tried reusing the same input, I must stop using this action input. I'll try something else instead.

- Resource Exhaustion of API

```
2024-10-08 16:33:51,396 - 140376448358336 - before_sleep.py-before_sleep:65 - WARNING: Retrying langchain_google_genai.chat_models._chat_with_retry.<locals>._chat_with_retry in 2.0 seconds as it raised ResourceExhausted: 429 Resource has been exhausted (e.g. check quota)..
```

```
2024-10-08 16:33:53,711 - 140376448358336 - before_sleep.py-before_sleep:65 - WARNING: Retrying langchain_google_genai.chat_models._chat_with_retry.<locals>._chat_with_retry in 4.0 seconds as it raised ResourceExhausted: 429 Resource has been exhausted (e.g. check quota)..
```

- Game Documentation

> Finished chain.

Hey team, here are some Tetris-inspired game ideas we can explore:

**\*\*1. Tetris: Time Warp\*\***

\* **\*\*Theme:\*\*** Sci-fi, time travel

\* **\*\*Gameplay:\*\*** Instead of falling blocks, players control a time-traveling device that can manipulate the past. Each level is a timeline, and players can rewind or fast-forward to strategically place blocks in the past, creating advantageous formations in the present.

\* **\*\*Mechanics:\*\***

\* **\*\*Time Rewind/Forward:\*\*** Players can rewind or fast-forward the timeline to place blocks in different points in the past.

\* **\*\*Time Distortion:\*\*** Special blocks can create temporary time distortions, affecting the speed of falling blocks or creating opportunities for strategic placement.

\* **\*\*Timeline Challenges:\*\*** Levels can have specific time-based challenges, like preventing a catastrophic event in the past or ensuring a specific outcome in the future.

**\*\*2. Tetris: Symphony of Shapes\*\***

\* **\*\*Theme:\*\*** Music, rhythm

- Feedback



## **\*\*7. Development Plan\*\***

- \* **Phase 1:** Prototype development focusing on core gameplay mechanics, block manipulation, and line clearing.
- \* **Phase 2:** Implementation of Fusion Blocks, visual design, and sound effects.
- \* **Phase 3:** Development of game modes, user interface, and multiplayer functionality.
- \* **Phase 4:** Testing, bug fixing, and optimization.

## **\*\*8. Target Audience\*\***

- \* Casual gamers
- \* Tetris fans
- \* Players seeking a unique and challenging puzzle experience

## **\*\*9. Marketing Strategy\*\***

- \* **Social media promotion:** Engage with gaming communities on platforms like Reddit, Twitter, and Discord.
- \* **Game trailers and gameplay videos:** Showcase the game's unique features and gameplay.
- \* **App store optimization:** Use relevant keywords and engaging descriptions to attract players.

This document provides a comprehensive overview of the Tetris Fusion game design. It outlines the core gameplay mechanics, unique elements, game modes, visual and audio design, user interface, technical requirements, and development plan. This information will serve as a foundation for building a captivating and engaging game experience. captivating and engaging game experience.

Please provide a feedback: good

- New Agent Deligation

> Finished chain.

> Entering new CrewAgentExecutor chain...

*Thought: I need to understand the code and how it works to be able to evaluate it.*

*Action: Ask question to co-worker*

*Action Input: { "coworker": "Senior Game developer", "question": "Can you explain how the game loop in this code works? I want to understand what happens in each iteration of the loop.", "context": "The code is for a Tetris game using Pygame. The game loop is the main part of the game, it controls the game's flow and updates the game state. I want to understand how the code handles player input, piece movement, line clearing, and other game logic within the loop." }*

- New Task allocation

> Entering new CrewAgentExecutor chain...

*Thought: I can definitely explain how the code handles different game modes and the logic behind each.*

*Final Answer:*

*Hey there! So, the way we handle different game modes is through a clever combination of conditional statements and dedicated functions. Let's break it down:*

- LLMs sharing their thought process with each other

1. In the `rotate\_tetromino` function, after rotating the tetromino, add a check for collision  
s:

```
```python
if check_collision(tetromino):
    # Revert the rotation if collision occurs
    rotate_tetromino(tetromino)
```
```

2. This will prevent the tetromino from rotating into invalid positions by reverting the rotation if a collision is detected.

**\*\*Overall:\*\***

The updated Tetris code is mostly functional, but it has a few bugs that need to be fixed. I have provided detailed solutions for each issue found.

```
```
```

2. This will prevent the tetromino from rotating into invalid positions by reverting the rotation if a collision is detected.

**\*\*Overall:\*\***

The updated Tetris code is mostly functional, but it has a few bugs that need to be fixed. I have provided detailed solutions for each issue found.

- Game code generation

```
# Move tetromino down only if not colliding
if not check_collision(current_tetromino):
    move_tetromino_down(current_tetromino)

# Check for collisions
if check_collision(current_tetromino):
    freeze_tetromino(current_tetromino)
    clear_lines()
    current_tetromino = create_tetromino()
    # Check for game over after creating a new tetromino
    if check_collision(current_tetromino):
        game_over = True
    # Check for game over if tetromino is above the board
    if current_tetromino['y'] < 0:
        game_over = True

# Draw the game
screen.fill(black)
draw_board()
draw_tetromino(current_tetromino)
draw_score()
if game_over:
    draw_game_over()

# Update the display
pygame.display.flip()

# Set frame rate
pygame.time.Clock().tick(60)

# Quit Pygame
pygame.quit()
```

- Python code creation

 tetris.py

A

- Game Validation and testing

**\*\*6. Drawing Tetrominoes:\*\***

- You'll need to iterate through the shapes and draw each block using ``pygame.draw.rect``. Make sure to calculate the correct position based on the current Tetromino's coordinates.

**\*\*7. Game Logic:\*\***

- You'll need to implement the following game logic:
  - Moving the Tetromino down
  - Moving the Tetromino left and right
  - Rotating the Tetromino
  - Checking for collisions with the bottom and sides of the game board
  - Clearing full lines
  - Generating new Tetrominoes

## **CONCLUSION**

## 6.1 Conclusion and Summary

The implementation of AI agent system in game development brings significance benefits to both game developers and players across the globe. This comprehensive system streamlines various aspects of AI agent system using crewai like rapid ideations, debugging and code generation.

Thus by developing this AI system we have learned various aspects of the **crewai framework**.

## **FUTURE ENHANCEMENT**



## 7.1 Future Enhancement

The future of the Game Developer AI holds exciting possibilities with the integrations of emerging technologies like GPT O1 model. As O1 model is exclusively designed for multimodal AI agent system.

Plans for the Future of Game Developer AI

- Integrating more power full AI models
- Integrating variety of programming language
- Reenforcing with more advance tools
- Adapting to upcoming advance libraries

## **REFERENCES**

## 8.1 Book Links

- Book name: **Neural Network from Scratch**  
Author: Harrison Kinsley & Daniel Kukieta  
Link: [NNFS-book-with-Implementation/Neural Networks from Scratch in Python.pdf at master · Dev-Gaju/NNFS-book-with-Implementation \(github.com\)](#)
- Book name: **Deep Learning Methods and Application**  
Author: Li Deng & Dong Yu  
Link: [Deep Learning Methods and Applications - Li Deng & Dong Yu.pdf — Yandex Disk](#)
- Book name: **Artificial Intelligence – A Modern Approach**  
Author: Stuart Russell & Peter Norvig  
Link: [Artificial Intelligence-A Modern Approach.pdf — Yandex Disk](#)

## 8.2 Video Link

- Video Title: **CrewAI Crash Course**  
Source: Youtube  
Link: <https://youtu.be/UV81LAb3x2g?si=fWoiZrMXVi-TdwK8>
- Video Title: **CrewAI Tutorial**  
Source: Youtube  
Link: <https://youtu.be/sPzc6hMg7So?si=owZEIAS0T8L3n2Tm>
- Video Title: **Right Way to build AI Agents**  
Source: Youtube  
Link: [https://youtu.be/iJjSjmZnNII?si=1guQt\\_jejC6EcqMA](https://youtu.be/iJjSjmZnNII?si=1guQt_jejC6EcqMA)
- Video Title: **Build Agentic Rag**  
Source: Youtube  
Link: <https://youtu.be/GtyiAd55XN0?si=HFi-krfC18N2Jn8->
- Video Title: **How I trained my agents**  
Source: Youtube  
Link: [https://youtu.be/TOM\\_845M6-o?si=jaIRhVnQI3JliQ50](https://youtu.be/TOM_845M6-o?si=jaIRhVnQI3JliQ50)
- Video Title: **How I made AI Agents do my work**  
Source: Youtube  
Link: <https://youtu.be/kJvXT25LkwA?si=fsESsFpM3q-8njUD>

## 8.3 Other References

- Name: CrewAI Documentation  
Link: <http://docs.crewai.com>