

# Integrating Generative AI into Modern Software Solutions

Abhishek Kumar

Director Data Science, Publicis Sapient

publicis  
sapient



**Director Data Science, Publicis Sapient**  
**Masters from University of California, Berkeley**  
**Google Developer Expert – ML**  
**Pluralsight Author**  
**Speaker**  
**Featured in Top 40 under 40 Data Scientist list**  
**Instructor at Berkeley Haas Exec-Ed**

## KEY TAKEAWAYS

1. Generative AI in Software Development
2. Architectural Considerations for Generative AI Integration
3. Optimizing and Fine-tuning Generative Models for Software Solutions
4. Challenges and Solutions in Deploying Generative Models
5. Future Trajectory of Generative AI in software development and architecture

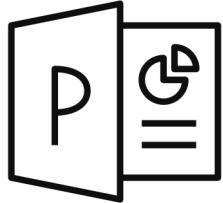
01

# GENERATIVE AI IN SOFTWARE DEVELOPMENT

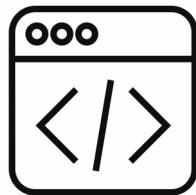


# Generative AI is transforming enterprises through

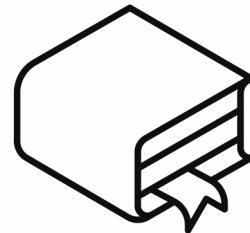
GENERATIVE  
CONTENT



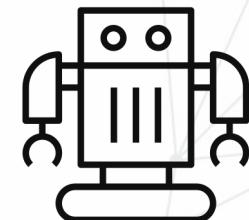
GENERATIVE  
SOFTWARE



GENERATIVE  
INTERFACES



GENERATIVE  
ROBOTS



**Generative AI is rapidly changing the way we work and create, allowing us to produce more in less time.**



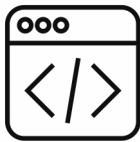
**Enterprises need to adapt and  
embrace AI to stay relevant and  
thrive.**

# We have to also focus on what is important for Tomorrow

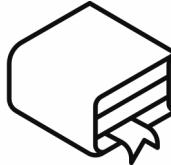
## GENERATIVE CONTENT



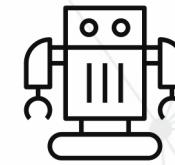
## GENERATIVE SOFTWARE



## GENERATIVE INTERFACES



## GENERATIVE ROBOTS



Today

Reduction in cost & time to develop software

Software volume increases

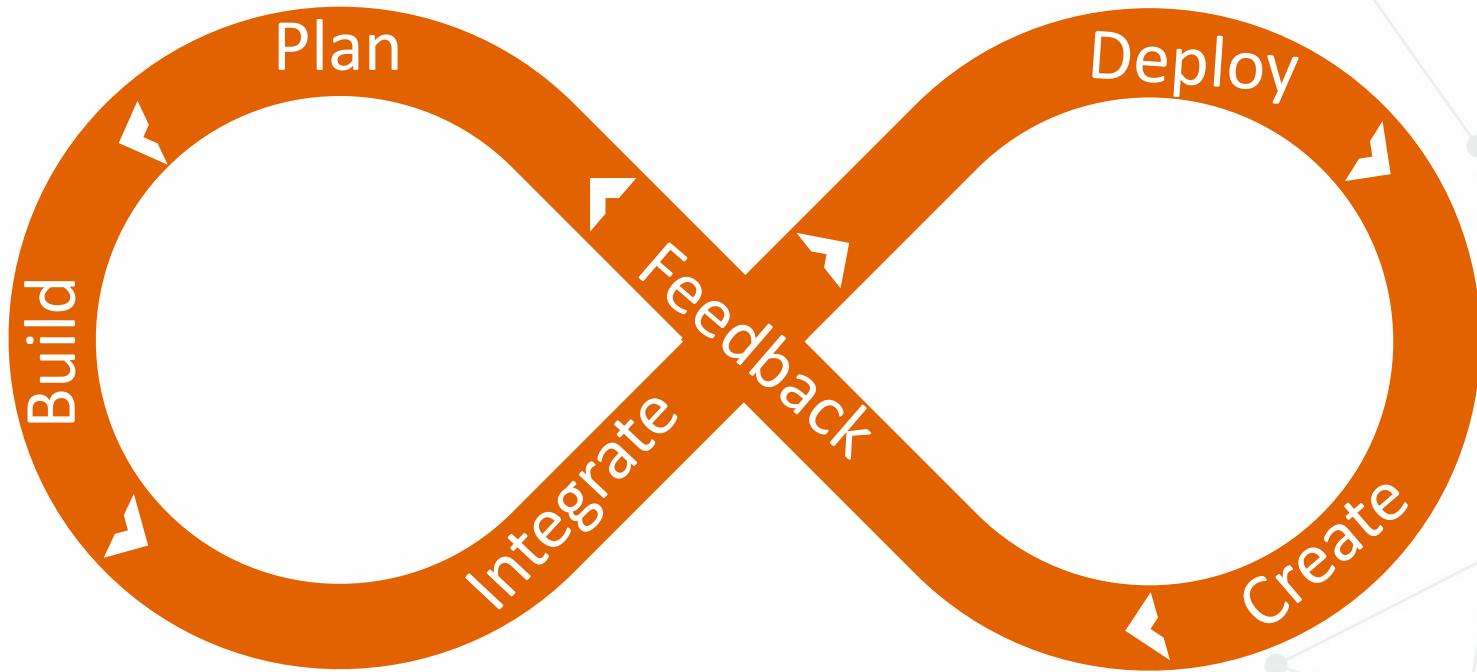
Hard-to-find skills become less of a roadblock

Economics of total cost of software ownership are disrupted

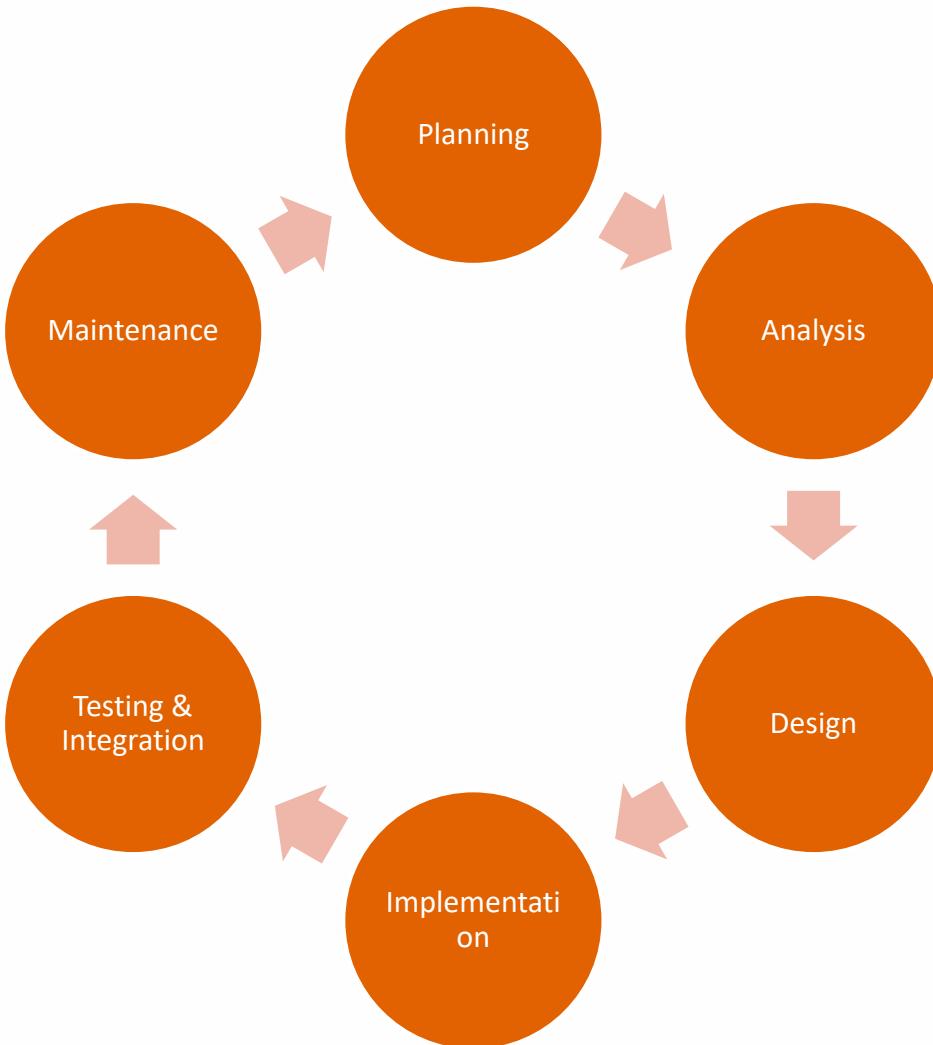
Tomorrow

Software development becomes on-demand

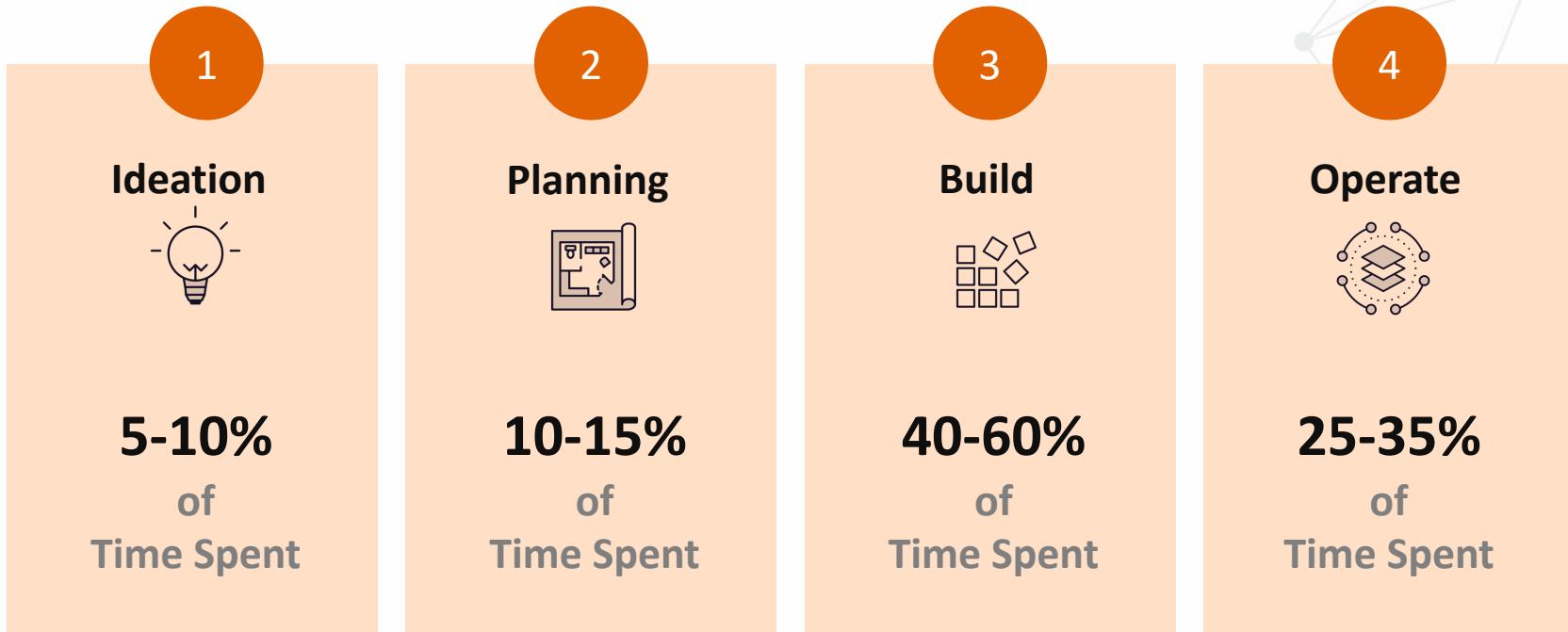
# SDLC



# SDLC

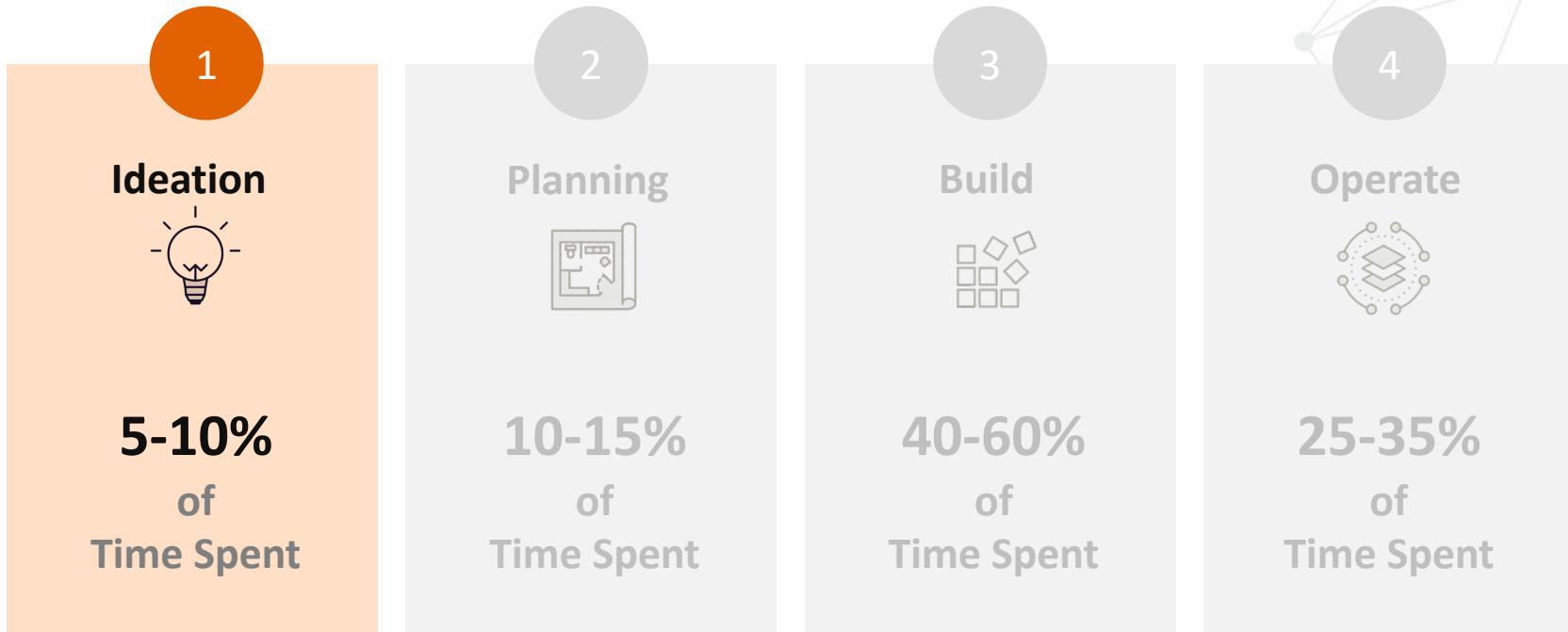


# Opportunities to optimize SDLC



Source: AWS Re:Invent 2023

# Generative AI in SDLC Ideation



Source: AWS Re:Invent 2023

1

## Ideation

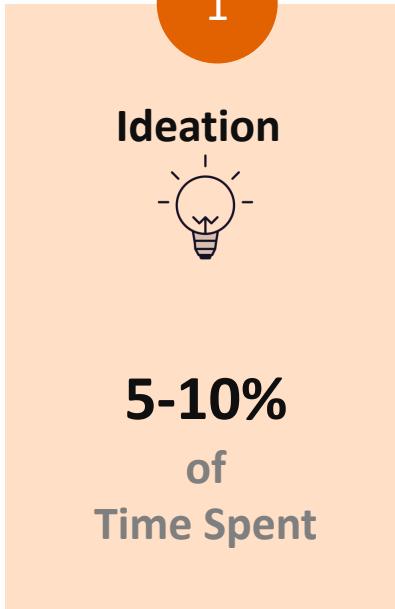


5-10%  
of  
Time Spent

ChatGPT		
💡 Examples	⚡ Capabilities	⚠ Limitations
"Explain quantum computing in simple terms" →	Remembers what user said earlier in the conversation	May occasionally generate incorrect information
"Got any creative ideas for a 10 year old's birthday?" →	Allows user to provide follow-up corrections	May occasionally produce harmful instructions or biased content
"How do I make an HTTP request in Javascript?" →	Trained to decline inappropriate requests	Limited knowledge of world and events after 2021

"Propose software product ideas tailored for *<persona>* and *<use-case>*"

"If our primary target audience suddenly became *<X>*, how would our product/service change?"



Disruptive thinking

Diversified brainstorming

Rapid idea generation

Trend synthesis

Scenario Exploration

1

## Ideation



5-10%  
of  
Time Spent

Tools

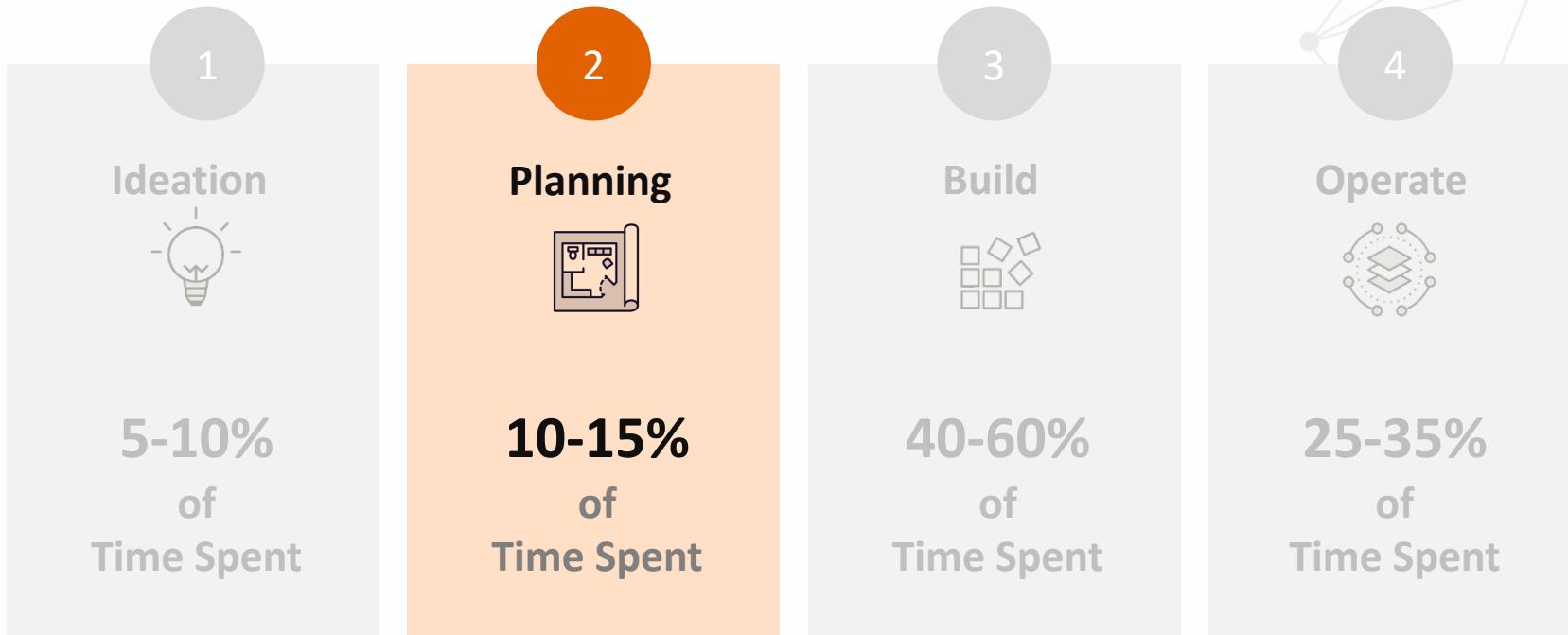


# Gemini



# Claude

# Generative AI in SDLC Planning



Source: AWS Re:Invent 2023

# Planning - User Stories



## You

Can you write the relevant user stories for audio storage as part of an app that uses genAI to create a conversation between a human and a bit?



## ChatGPT

1. As a user, I want to be able to record and store audio messages within the app so that I can easily revisit and listen to past conversations with the AI bit.
2. As a user, I want the option to organize my audio messages into folders or categories for better management and quick access.
3. As a user, I expect the app to provide a secure and private audio storage solution, ensuring that my recorded conversations with the AI bit are kept confidential.
4. As a user, I would like the ability to share specific audio messages with other users or external platforms for collaborative discussions or analysis.
5. As a user, I want to have a search functionality within the audio storage feature, allowing me to quickly find specific conversations or keywords in the recorded content.
6. As a user, I expect the app to support multiple audio formats and provide an option to export or download my audio messages for external use.

# Planning - Wireframing

## WireGen - Generate wireframes with AI

Write what you wish the AI to create and click 'Generate' →

The AI will detect and generate the Wireframe designs →

You can tell the AI to change the wireframe →

When ready, just click 'Copy to Figma'

WireGen - AI GPT wireframe generation

Beta

Generate wireframes with AI

Describe your idea

Fitness tracker app that helps users track their fitness goals, including calorie counting, exercise tracking, and progress monitoring

Example 1 Example 2 Example 3 134/500

Generate

Home - Track your fitness goals

Calorie Tracker - Log your daily calorie intake

Exercise Tracker - Track your daily exercise routine

The Exercise Tracker screen is designed to provide users with a seamless and intuitive experience for tracking their daily exercise routine. The screen features a clean and organized layout, with easy-to-use controls for adding and editing exercises. Users can choose from a variety of exercise types, including cardio, strength training, and yoga, and can track their progress over time using charts and graphs. The screen also includes helpful features like reminders and notifications to keep users on track with their fitness goals. Overall, the Exercise Tracker screen is designed to make it easy and enjoyable for users to stay motivated and achieve their fitness goals.

What would you want to change? ⓘ

Contact Help Leave Feedback \*

v0.04 Beta

Add Exercise

Exercise Type Choose...

Duration (minutes)

Calories Burned

Date

Notes

Add Exercise

Today's Exercise Summary

Total Calories Burned 300 Total Exercise Time 60 minutes

Exercise History

Date	Exercise Type	Duration	Calories Burned
10/01/2021	Cardio	30 minutes	200
10/02/2021	Strength Training	20 minutes	100
10/03/2021	Yoga	10 minutes	50

Copy to Figma

Frame

< Exercise Tracker - Track your daily exercise

Add Exercise

Exercise Type Choose...

Duration (minutes)

Calories Burned

Date

Notes

Add Exercise

Today's Exercise Summary

Total Calories Burned 350 Total Exercise Time 60 minutes

Exercise History

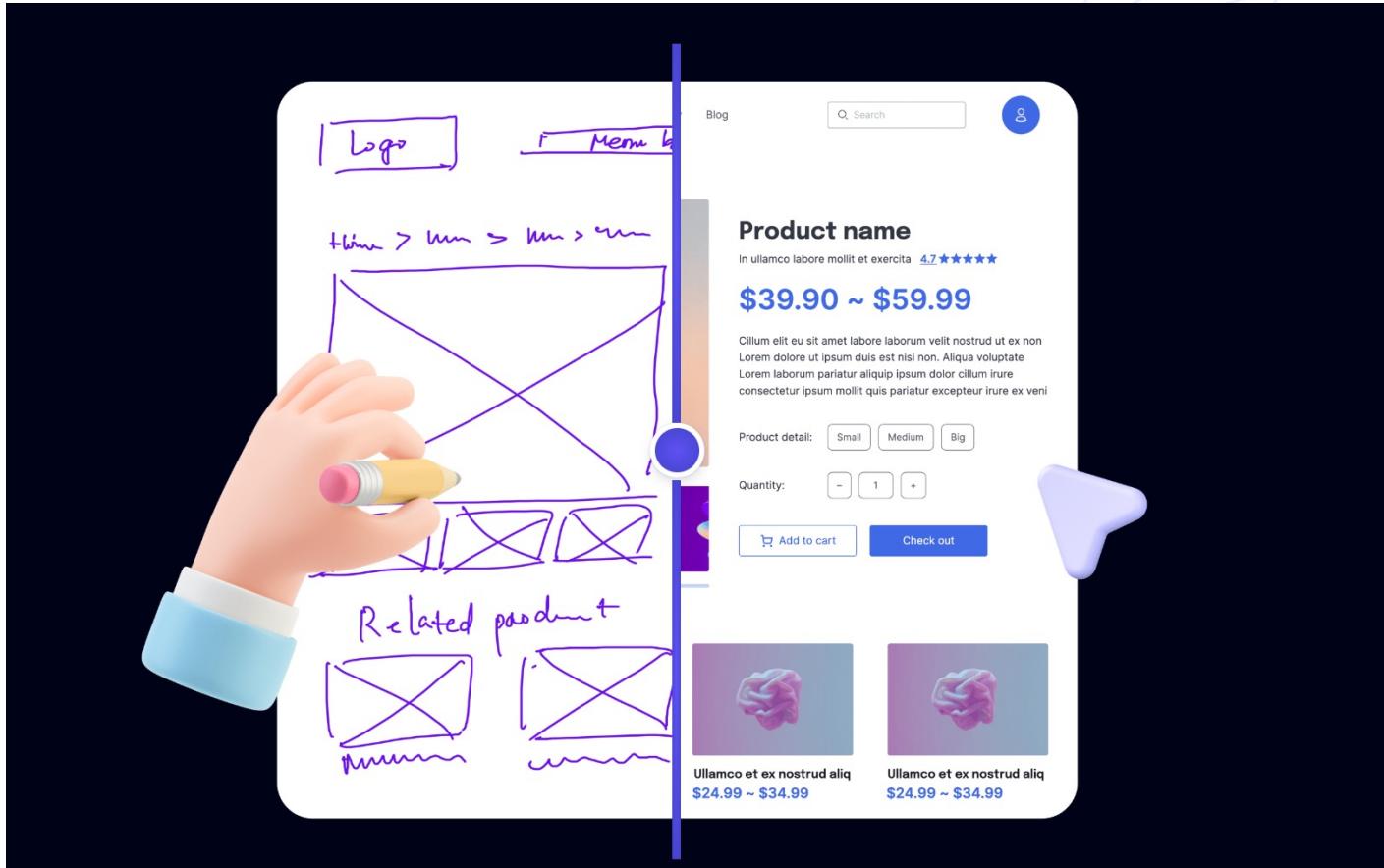
Date	Exercise Type	Duration	Calories Burned
10/01/2021	Cardio	30 minutes	200
10/02/2021	Strength Training	20 minutes	100
10/03/2021	Yoga	10 minutes	50

Home Progress Tracker Profile

# Planning - Designing

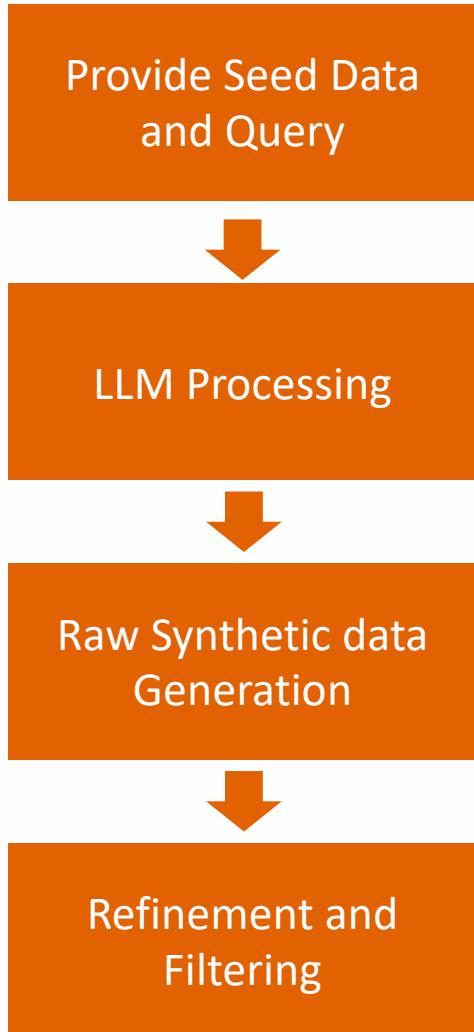
Visily AI

Sketch to  
Design



Source : <https://www.visily.ai/ai-design/#sketch-to-design>

# Planning - Synthetic Data Generation

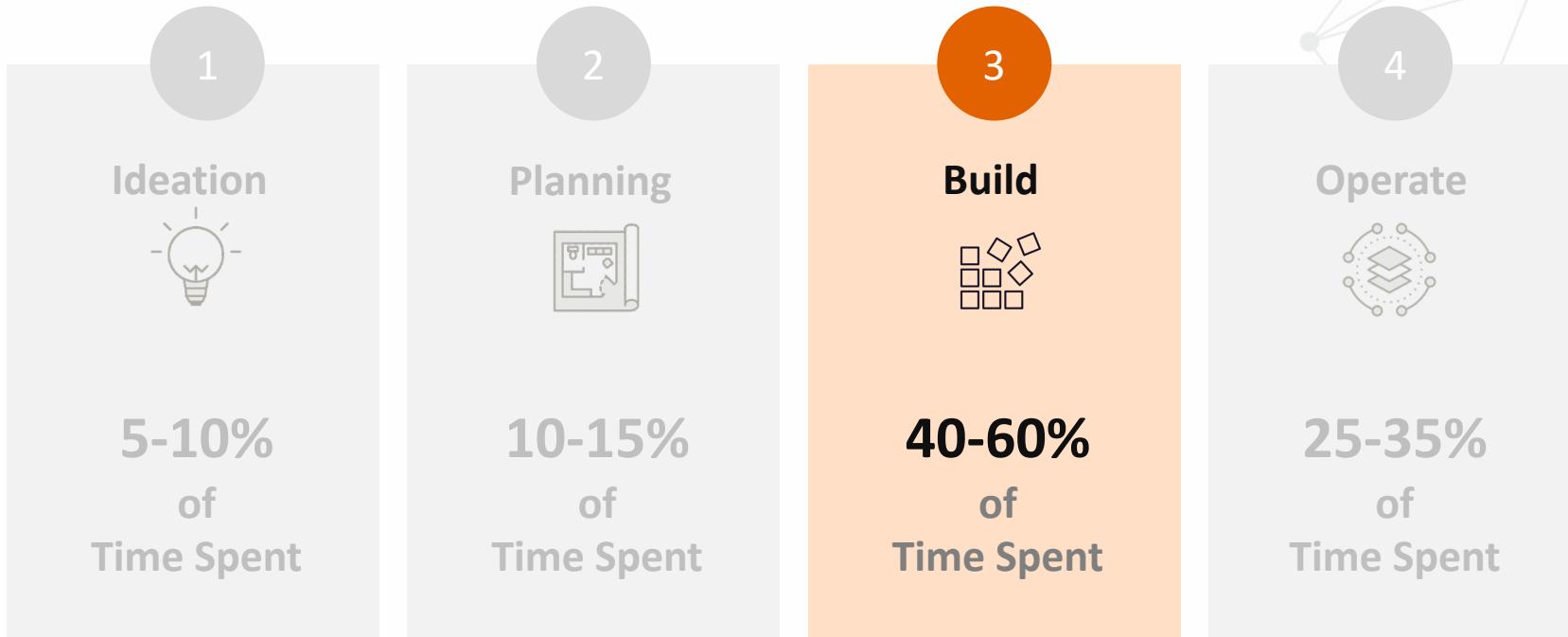


The screenshot shows the Gretel AI interface displaying two tables of synthetic data:

users_synthetic			
user_id	fname	lname	verified
IRF-uB0m6GKeV8WX9Si1NYFp6E	Jason	Alexander	true
n2XiCF3BgyNIJ9WgZ-jIn	Maya	Spivak	true
po_00157	102.68	13:39:22 GMT-0800	IRF-uB0m6GKeV8WX9Si1NYFp6E
po_00158	92.76	09:29:02 GMT-0700	n2XiCF3BgyNIJ9WgZ-jIn
po_00159	408.90	13:12:30 GMT-0700	KLeqkVQhPYvkv1hoRkBtx
po_00160	231.56	09:09:55 GMT-0800	X16Cc936qEbMRKN602oZx
po_00161	11.21	19:43:53 GMT-0700	L7NCnqtW1nZdi7TdAbaEC
po_00162	561.01	07:31:28 GMT-0700	Zz102D4o8b6WPdGvmwccgy

Gretel AI

# Generative AI in SDLC Build



Source: AWS Re:Invent 2023

# BUILD - Code Search

Retrieval  
Augmented  
Generation

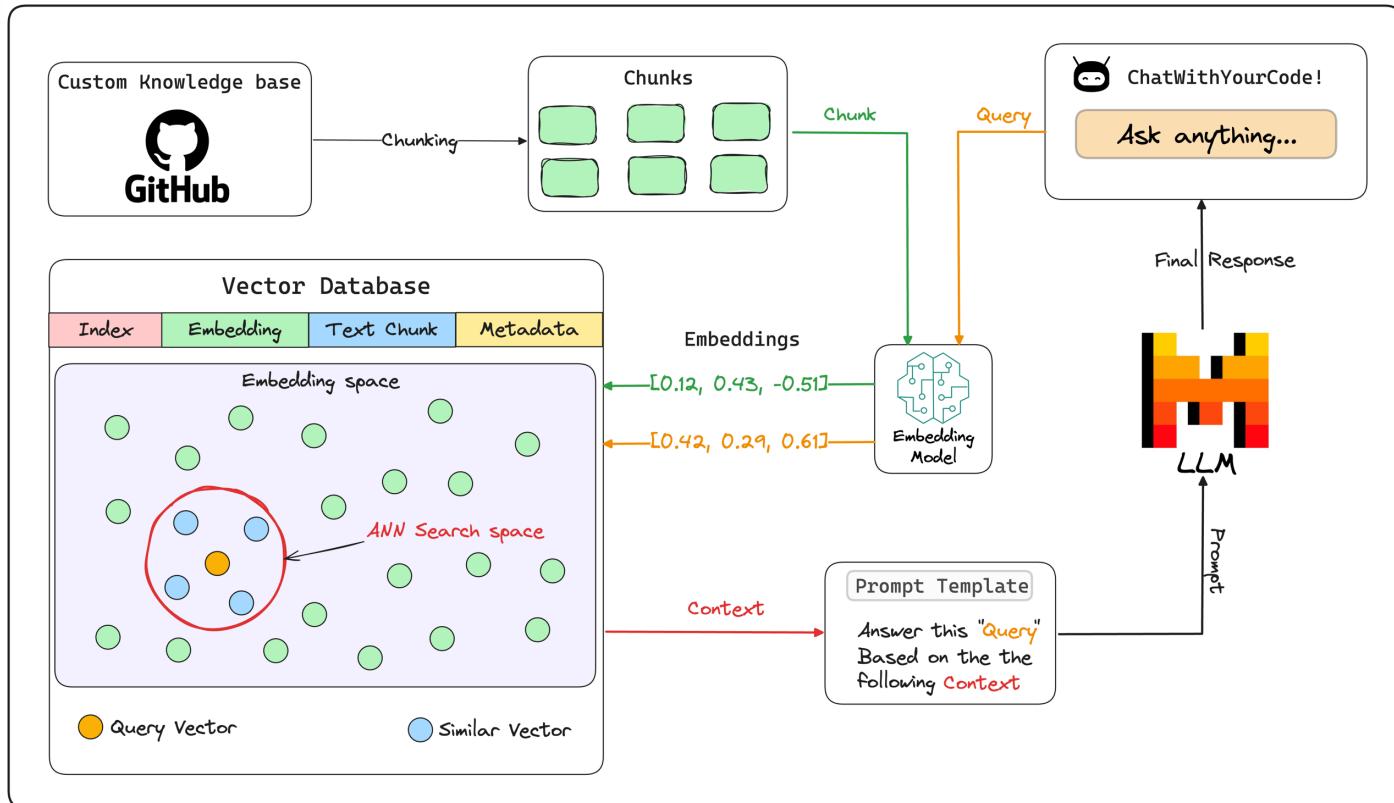


Image Source : <https://lightning.ai/lightning-ai/studios/chat-with-your-code-using-rag>

# BUILD - Code Generation

ChatGPT



Gemini



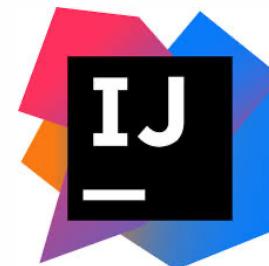
 GitHub Copilot



 aws  
CodeWhisperer



Visual Studio Code



# BUILD - Code Completion

```
def quicksort(arr):  
    if len(arr) <= 1:  
        return arr  
    pivot = arr[0]  
    less = [x for x in arr[1:] if x <= pivot]  
    greater = [x for x in arr[1:] if x > pivot]  
    return quicksort(less) + [pivot] + quicksort(greater)
```

# BUILD - Code Explanation

The screenshot shows a code editor window with a dark theme. On the left, a file named 'utils.js' is open, containing the following code:

```
104 function resetShowWordCountPill(defaultShowWordCountPill) {  
107 }  
108  
109 function countWords(str) {  
110     return str.trim().split(/\s+/).length;  
111 }  
112  
113 function calculateCharactersAndWords(str) {  
114     const characterCount = str.length;  
115     const wordCount = str === '' ? countWords(str) : 0;  
116     const wordCountText = `${characterCount} character(s), ${wordCount} word${wordCount === 1 ? '' : 's'}.  
117     return wordCountText;  
118 }  
119  
120  
121 function copyNotesToClipboard(note) {  
122     navigator.clipboard.writeText(note).then(function () {  
123         showToast('Notes copied to clipboard!')  
124     }, function () {  
125         showToast('Failure to copy. Check permissions for clipboard.')  
126     });  
127 }
```

The right side of the window displays the GitHub Copilot Labs interface. A large orange box highlights the 'EXPLAIN' section, which contains the following code snippet:

```
function calculatecharactersAndwords(str) {  
    const characterCount = str.length;  
    const wordCount = str !== '' ? countWords(str) : 0;  
    const wordCountText = `${characterCount} character(s), ${wordCount} word${wordCount === 1 ? '' : 's'}.  
    return wordCountText;  
}
```

Below this, a dropdown menu shows 'Code does following' and an 'Ask Copilot' button. The 'RESULT' section contains the following text:

/\* The code above does the following:  
1. Determines the number of characters in the string.  
2. Determines the number of words in the string.  
3. Combines the number of characters and words into a string.  
4. Returns the string. \*/

At the bottom of the Copilot interface, there are buttons for 'Incorrect', 'Unhelpful', and 'Great'. The bottom navigation bar of the code editor includes tabs for 'FILE', 'EDITOR', 'TESTS', 'LINTING', 'FORMAT', 'REFACTOR', 'REFRESH', and 'HELP'.

# BUILD - Code Refactoring

CODEGPT CHAT

Llama3-70b-8192

User  
refactor

9 CodeGPT Llama3-70b-8192

python

```
import mysql.connector
from mysql.connector import Error

# A Python script to connect to a MySQL database
try:

    connection = mysql.connector.connect(host='localhost',
                                           database='Electronics',
                                           user='pynative',
                                           password='pynative')

    if connection.is_connected():
        db_Info = connection.get_server_info()
        print("Connected to MySQL Server version ", db_Info)
        cursor = connection.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()
        print("You're connected to database: ", record)

except Error as e:
    print("Error while connecting to MySQL", e)
finally:
    if connection.is_connected():
        cursor.close()
        connection.close()
        print("MySQL connection is closed")
```

python

Copy code | Insert code

```
import mysql.connector
from mysql.connector import Error

def connect_to_database(host, database, user, password):
    """
    Establishes a connection to a MySQL database.

    Args:
    host (str): The hostname of the MySQL server.
    database (str): The name of the database to connect.
    user (str): The username to use for the connection.
    password (str): The password to use for the connection.

    Returns:
    mysql.connector.connection.MySQLConnection: A connection object.
    """
    try:
        connection = mysql.connector.connect(
            host=host,
            database=database,
            user=user,
            password=password
        )
        return connection
    except Error as e:
        print(f"Error while connecting to MySQL: {e}")
        return None

def main():
    host = 'localhost'
    database = 'Electronics'
    user = 'pynative'
    password = 'pynative@#29'
```

# BUILD - Code Documentation

```
sample.py > ...
1 import mysql.connector
2 from mysql.connector import Error
3
4 # A Python script to connect to a MySQL database
5 try:
6
7     connection = mysql.connector.connect(host='localhost',
8                                         database='Electronics',
9                                         user='pynative',
10                                        password='pynative@#29')
11
12     if connection.is_connected():
13         db_Info = connection.get_server_info()
14         print("Connected to MySQL Server version ", db_Info)
15         cursor = connection.cursor()
16         cursor.execute("select database();")
17         record = cursor.fetchone()
18         print("You're connected to database: ", record)
19
20     except Error as e:
21         print("Error while connecting to MySQL", e)
22     finally:
23         if connection.is_connected():
24             cursor.close()
25             connection.close()
26             print("MySQL connection is closed")
27
```

```
python
Copy code | Insert code
try:
    # Establish a connection to the MySQL database
    connection = mysql.connector.connect(
        host='localhost', # Hostname of the MySQL server
        database='Electronics', # Name of the database
        user='pynative', # Username to use for the connection
        password='pynative@#29' # Password to use for the connection
    )

    # Check if the connection was successful
    if connection.is_connected():
        # Get information about the MySQL server
        db_Info = connection.get_server_info()
        print("Connected to MySQL Server version ", db_Info)

        # Create a cursor object to execute SQL queries
        cursor = connection.cursor()

        # Execute a SQL query to select the current database
        cursor.execute("select database();")

        # Fetch the result of the query
        record = cursor.fetchone()

        # Print the result
        print("You're connected to database: ", record)
```

# BUILD - Code Testing

```
sample.py > ...
1 import mysql.connector
2 from mysql.connector import Error
3
4 # A Python script to connect to a MySQL database
5 try:
6
7     connection = mysql.connector.connect(host='localhost',
8                                         database='Electronics',
9                                         user='pynative',
10                                        password='pynative@#29')
11
12     if connection.is_connected():
13         db_Info = connection.get_server_info()
14         print("Connected to MySQL Server version ", db_Info)
15         cursor = connection.cursor()
16         cursor.execute("select database();")
17         record = cursor.fetchone()
18         print("You're connected to database: ", record)
19
20     except Error as e:
21         print("Error while connecting to MySQL", e)
22     finally:
23         if connection.is_connected():
24             cursor.close()
25             connection.close()
26             print("MySQL connection is closed")
```

```
python
Copy code | Insert code
import unittest
from unittest.mock import patch, MagicMock
import mysql.connector
from mysql.connector import Error

class TestMySQLConnection(unittest.TestCase):

    @patch('mysql.connector.connect')
    def test_successful_connection(self, mock_connect):
        # Arrange
        mock_connection = MagicMock()
        mock_cursor = MagicMock()
        mock_connection.cursor.return_value = mock_cursor
        mock_connect.return_value = mock_connection
        mock_cursor.fetchone.return_value = ('Electroni')
        mock_connection.get_server_info.return_value = ('Electroni')

        # Act
        try:
            connection = mysql.connector.connect(host='localhost',
                                                 database='Electronics',
                                                 user='pynative',
                                                 password='pynative@#29')
            if connection.is_connected():
                db_Info = connection.get_server_info()
                print("Connected to MySQL Server version ", db_Info)
                cursor = connection.cursor()
                cursor.execute("select database();")
                record = cursor.fetchone()
                print("You're connected to database: ", record)

        # Assert
        self.assertEqual(mock_connect.called, True)
        self.assertEqual(mock_connection.called, True)
        self.assertEqual(mock_cursor.called, True)
        self.assertEqual(record, ('Electronics'))
```

# BUILD - Code Review and Bugfix

```
sample.py > ...
1 import mysql.connector
2 from mysql.connector import Error
3
4 # A Python script to connect to a MySQL database
5 try:
6
7     connection = mysql.connector.connect(host='localhost',
8                                         database='Electronics',
9                                         user='pynative',
10                                        password='pynative@#29')
11
12     if connection.is_connected():
13         db_Info = connection.get_server_info()
14         print("Connected to MySQL Server version ", db_Info)
15         cursor = connection.cursor()
16         cursor.execute("select database();")
17         record = cursor.fetchone()
18         print("You're connected to database: ", record)
19
20     except Error as e:
21         print("Error while connecting to MySQL", e)
22     finally:
23         if connection.is_connected():
24             cursor.close()
25             connection.close()
26             print("MySQL connection is closed")
27
```

## Code Review and Correction

### Issues Identified:

- Unprotected Cursor Closure:** The `cursor.close()` statement is not protected from potential `AttributeError` exceptions, which can occur if the `cursor` object is not properly initialized.
- Unprotected Connection Closure:** The `connection.close()` statement is not protected from potential `AttributeError` exceptions, which can occur if the `connection` object is not properly initialized.

### Corrected Code:

```
python          Copy code | Insert code
import mysql.connector
from mysql.connector import Error

try:
    connection = mysql.connector.connect(host='localhost',
                                           database='Electronics',
                                           user='pynative',
                                           password='pynative@#29')

    if connection.is_connected():
        db_Info = connection.get_server_info()
        print("Connected to MySQL Server version ", db_Info)
        cursor = connection.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()
        print("You're connected to database: ", record)

    # Protect cursor closure
    if 'cursor' in locals() and cursor:
        cursor.close()

finally:
    connection.close()
    print("MySQL connection is closed")
```

# BUILD - Code Optimization

## Code Polish

### Code to be edited:

```
public ListNode swapPairs(ListNode head) {  
    if (head == null || head.next == null){  
        return head;  
    }  
    ListNode second = head.next;  
    head.next = swapPairs(second.next);  
    second.next = head;  
    return second;  
}
```

*Optimize execution  
efficiency and  
memory usage*

### Difficulty:

Medium

### Average Running Time:

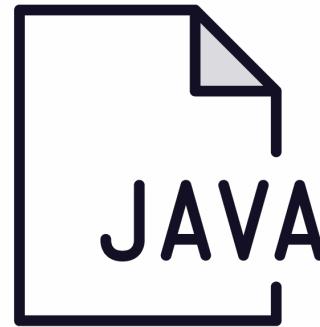
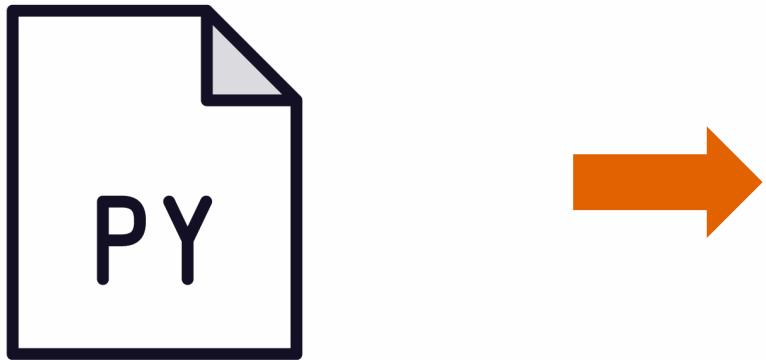
14546ms

### Average Memory:

29811kb

**Optimized for execution time, memory**

# BUILD - Code Translation

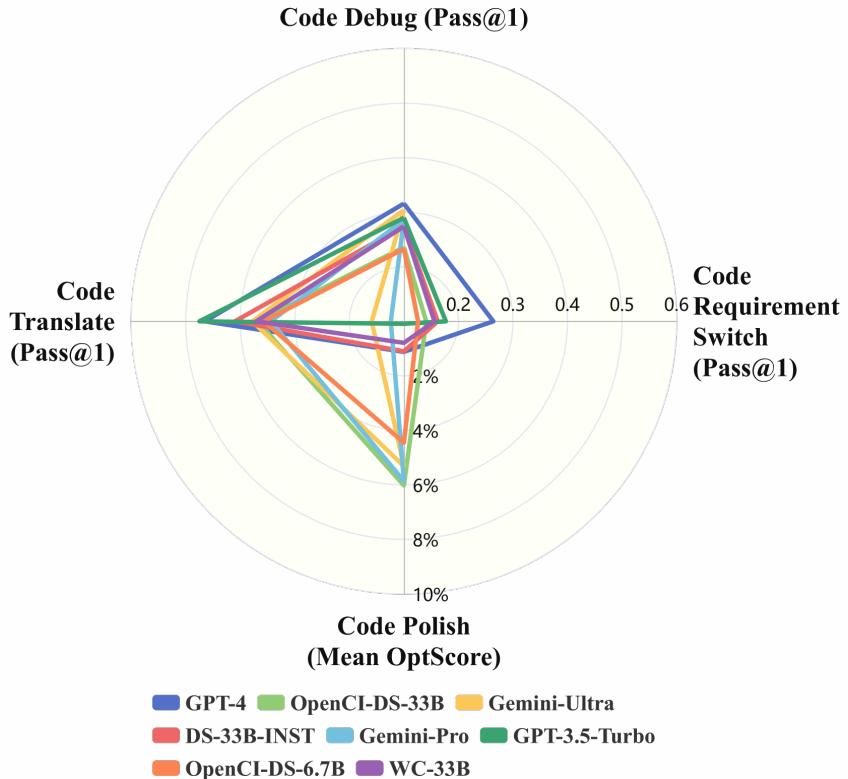


# BUILD - Code Translation

Dataset	Source Language	Source Samples	#Tests	Target Language	#Translations	% Unsuccessful Translations						
						CodeGen	CodeGeeX	StarCoder	GPT-4	Llama 2	TB-Airoboros	TB-Vicuna
CodeNet [Puri et al.(2021)]	C	200	200	C++, Go, Java, Python	800	76.6%	85.1%	<b>58.0%</b>	<b>17.0%</b>	85.1%	81.2%	95.6%
	C++	200	200	C, Go, Java, Python	800	86.0%	96.4%	<b>60.9%</b>	<b>20.0%</b>	90.5%	91.7%	96.6%
	Go	200	200	C, C++, Java, Python	800	85.7%	94.1%	<b>58.0%</b>	<b>14.5%</b>	83.1%	93.4%	99.1%
	Java	200	200	C, C++, Go, Python	800	78.7%	89.7%	<b>69.7%</b>	<b>18.7%</b>	86.1%	93.5%	99.9%
	Python	200	200	C, C++, Go, Java	800	82.5%	92.7%	<b>66.7%</b>	<b>20.1%</b>	89.0%	93.5%	99.0%
Total/Average (CodeNet)	-	1,000	1,000	-	4,000	81.9%	91.6%	<b>62.7%</b>	<b>18.0%</b>	86.8%	90.7%	98.0%
AVATAR [Ahmad et al.(2021)]	Java	249	6,255	C, C++, Go, Python	996	91.9%	98.2%	<b>88.1%</b>	<b>29.2%</b>	98.2%	94.9%	100%
	Python	250		C, C++, Go, Java	1,000	96.2%	98.4%	<b>85.8%</b>	<b>47.8%</b>	95.3%	99.1%	99.1%
Total/Average (AVATAR)	-	499	6,255	-	1,996	94.1%	98.3%	<b>87.0%</b>	<b>38.5%</b>	96.8%	97.0%	99.6%
EvalPlus [Liu et al.(2023b)]	Python	164	2,682	Java	164	83.5%	96.3%	<b>78.0%</b>	<b>20.7%</b>	98.8%	86.0%	92.1%
Commons CLI [apa(2023)]	Java	22	310	Python	22	100%	100%	100%	<b>86.4%</b>	100%	100%	100%
Click [cli(2023)]	Python	15	611	Java	15	100%	100%	100%	100%	100%	100%	100%
Total/Average (All)	-	1,700	10,858	-	6,197	91.9%	97.2%	<b>85.5%</b>	<b>52.7%</b>	96.5%	94.7%	97.9%

GPT-4 managed a success rate of **8.1%** in real-world projects

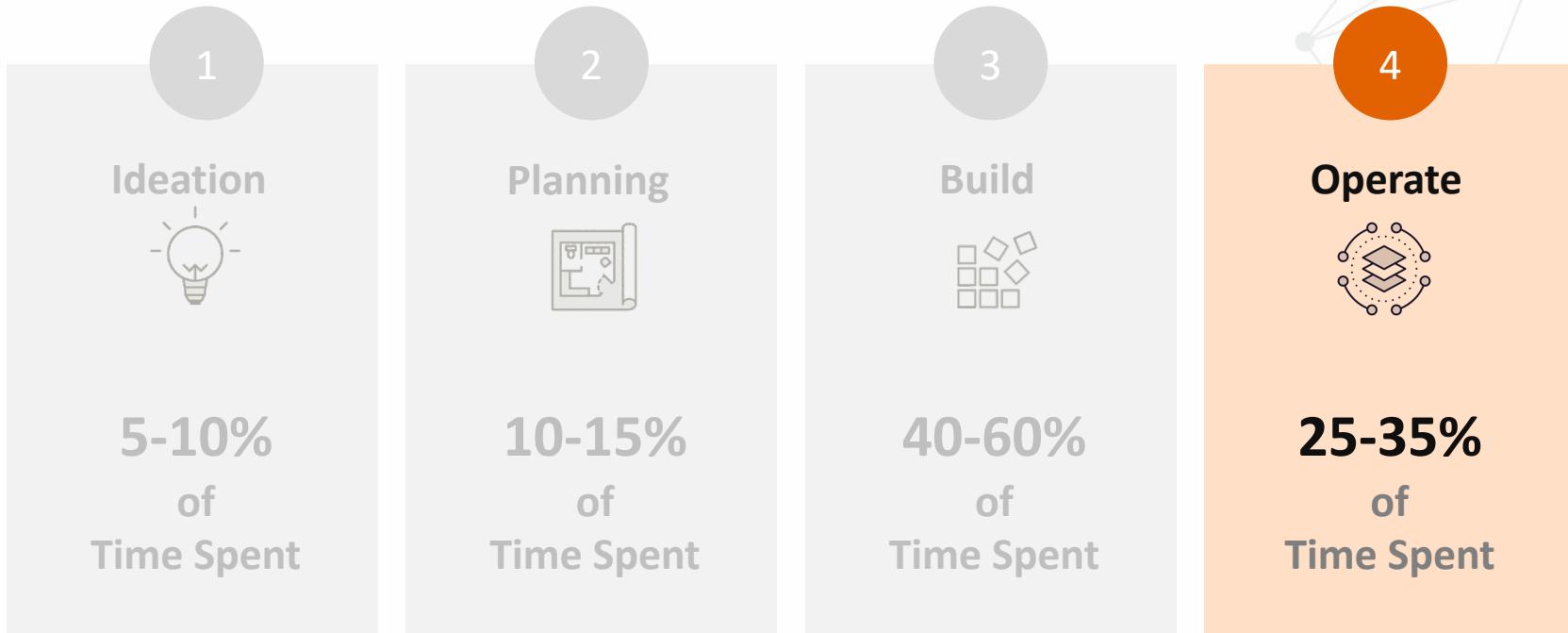
# BUILD - Code Editor Bench



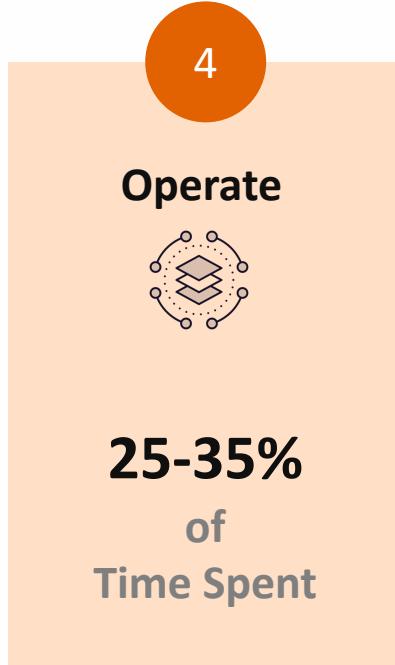
Model	Size	Open	Debug	Translate	Switch	Polish	Win Rate	Zero-shot	
								0.1%	0.2%
GPT-4	-	✗	0.316(0.493)	0.465(0.503)	0.264	1.12%(1.33%)	0.855(0.868)		
OpenCI-DS-33B	33B	✓	0.236(0.429)	0.368(0.428)	0.141	6.02%(6.49%)	0.776(0.816)		
Gemini-Ultra	-	✗	0.304(0.459)	0.378(0.278)	0.041	5.31%(3.77%)	0.750(0.579)		
DS-33B-INST	33B	✓	0.275(0.487)	0.410(0.451)	0.162	1.10%(1.14%)	0.737(0.757)		
Gemini-Pro	-	✗	0.286(0.423)	0.344(0.344)	0.076	5.86%(6.65%)	0.737(0.711)		
GPT-3.5-Turbo	-	✗	0.290(0.494)	0.475(0.480)	0.177	0.09%(0.84%)	0.724(0.776)		
OpenCI-DS-6.7B	6.7B	✓	0.233(0.402)	0.357(0.384)	0.126	4.45%(4.28%)	0.671(0.697)		
WC-33B	33B	✓	0.274(0.487)	0.371(0.438)	0.156	0.79%(0.90%)	0.632(0.704)		
GLM-4	-	✗	0.220(0.271)	0.278(0.365)	0.085	5.17%(6.46%)	0.526(0.592)		
MC-DS-6.7B	6.7B	✓	0.242(0.406)	0.343(0.401)	0.130	0.21%(1.99%)	0.513(0.697)		
PH-34B	34B	✓	0.230(0.369)	0.279(0.331)	0.074	2.84%(1.78%)	0.500(0.539)		
Octocoder	15.5B	✓	0.042(0.145)	0.392(0.223)	0.030	1.39%(2.70%)	0.434(0.289)		
CL-13B-INST	13B	✓	0.176(0.368)	0.333(0.275)	0.021	2.31%(1.82%)	0.421(0.368)		
CL-34B	34B	✓	0.163(0.250)	0.310(0.240)	0.052	1.10%(0.84%)	0.382(0.171)		
MC-CL-7B	7B	✓	0.174(0.317)	0.272(0.276)	0.039	1.31%(1.31%)	0.329(0.342)		
WC-15B	15B	✓	0.159(0.354)	0.309(0.278)	0.067	0.91%(0.96%)	0.329(0.408)		
CL-7b-INST	7B	✓	0.155(0.336)	0.289(0.231)	0.017	1.47%(1.17%)	0.289(0.250)		
CL-34B-INST	34B	✓	0.131(0.250)	0.287(0.240)	0.027	1.02%(0.84%)	0.211(0.171)		
CF-34B	34B	✓	0.166(0.223)	0.218(0.177)	0.028	0.33%(0.70%)	0.184(0.105)		

Source : <https://arxiv.org/html/2404.03543v1>

# Generative AI in SDLC Operate



Source: AWS Re:Invent 2023



## Code Deployment

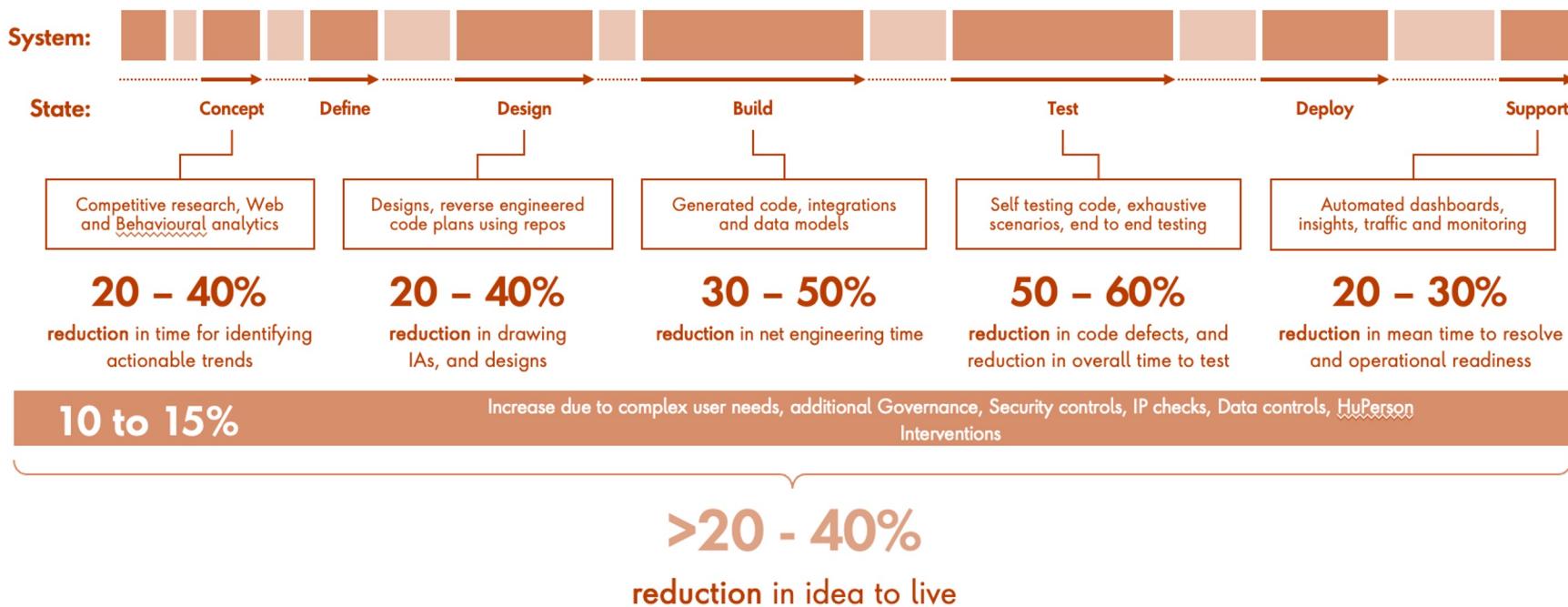
## Pipeline Automation

## Anomaly detection

## Infrastructure optimization

## Adaptive Problem resolution and incident handling

# Product Engineering - GAI in the end-to-end product lifecycle can result in significant value creation and deliver a 20-40% efficiency saving in idea to live



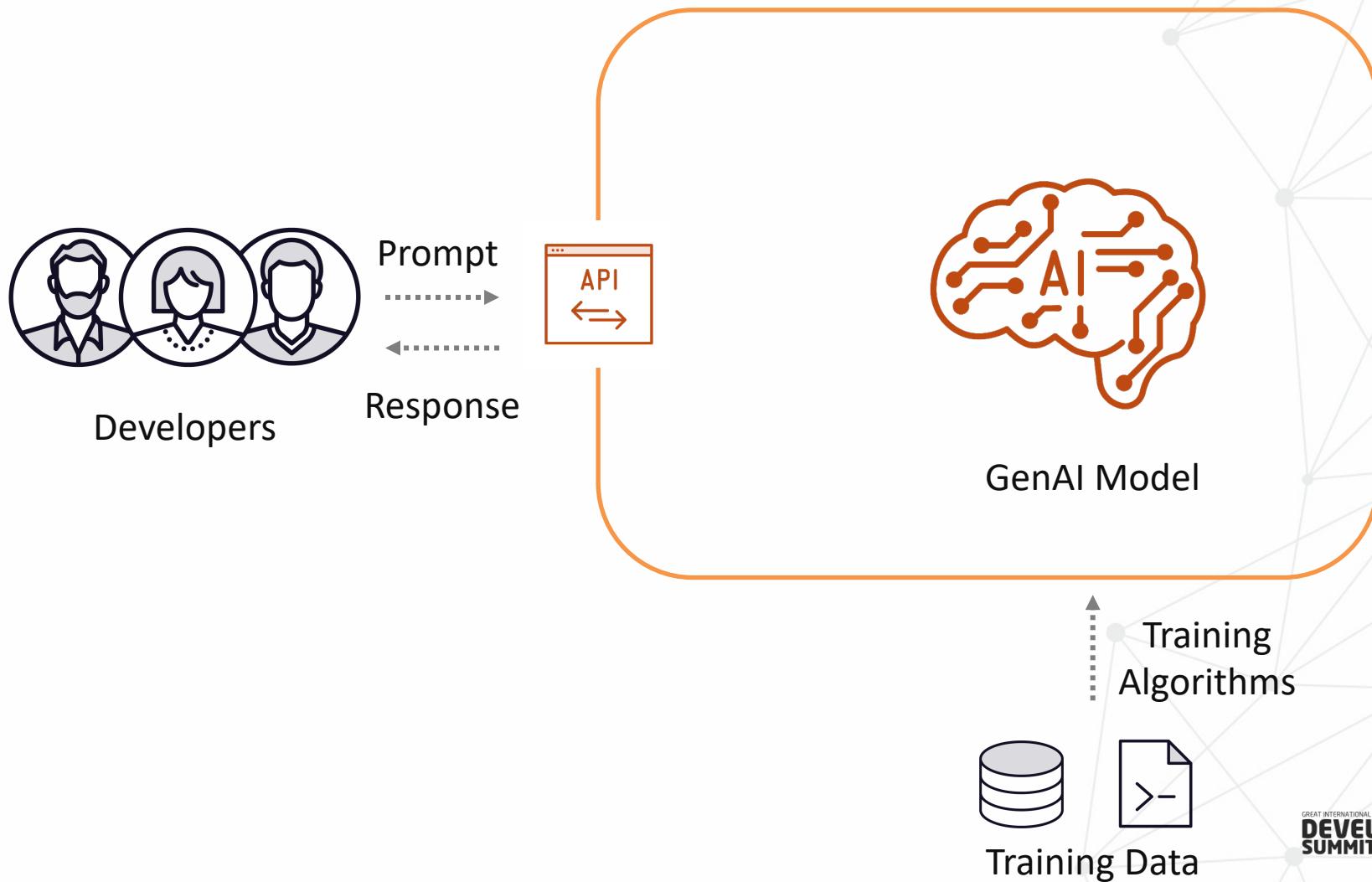
**Source:** This data is based on our analysis of over 100 end to end experiments with multiple AI and GenAI tools used internally across projects

Section 02

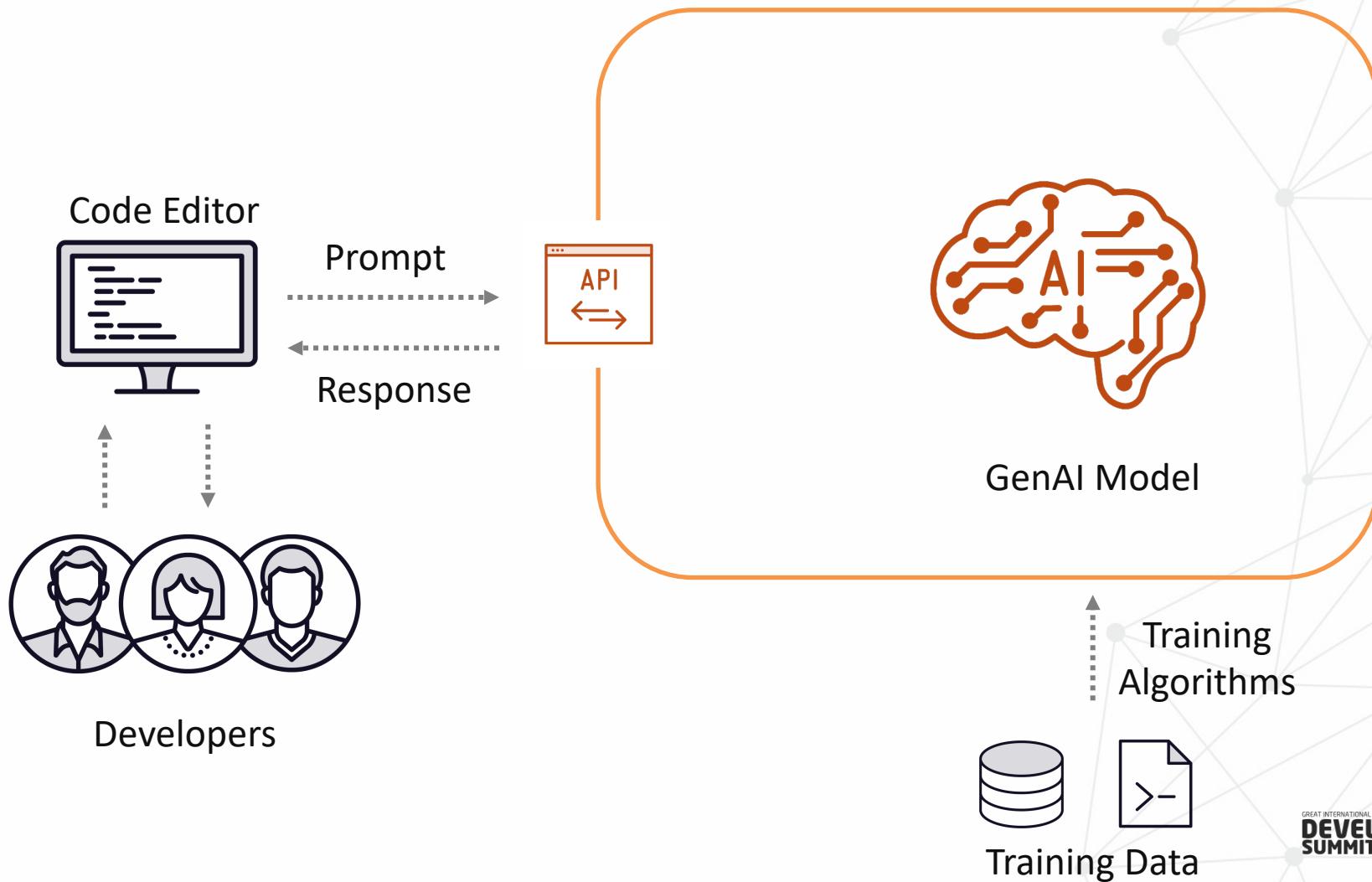
# ARCHITECTURAL CONSIDERATIONS FOR GENERATIVE AI INTEGRATION



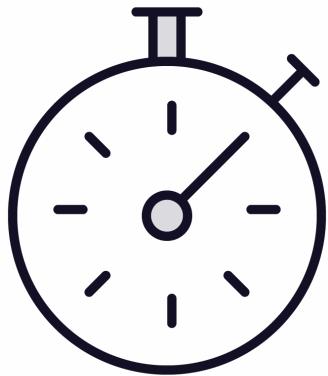
# How Developers Integrate with GenAI Model



# How Developers Integrate with GenAI Model



# Considerations and Challenges



## Managing Latency Considerations

Subscription level quota

Switch to smaller model

LPU

Manage your own infra



## Managing Data Privacy Considerations

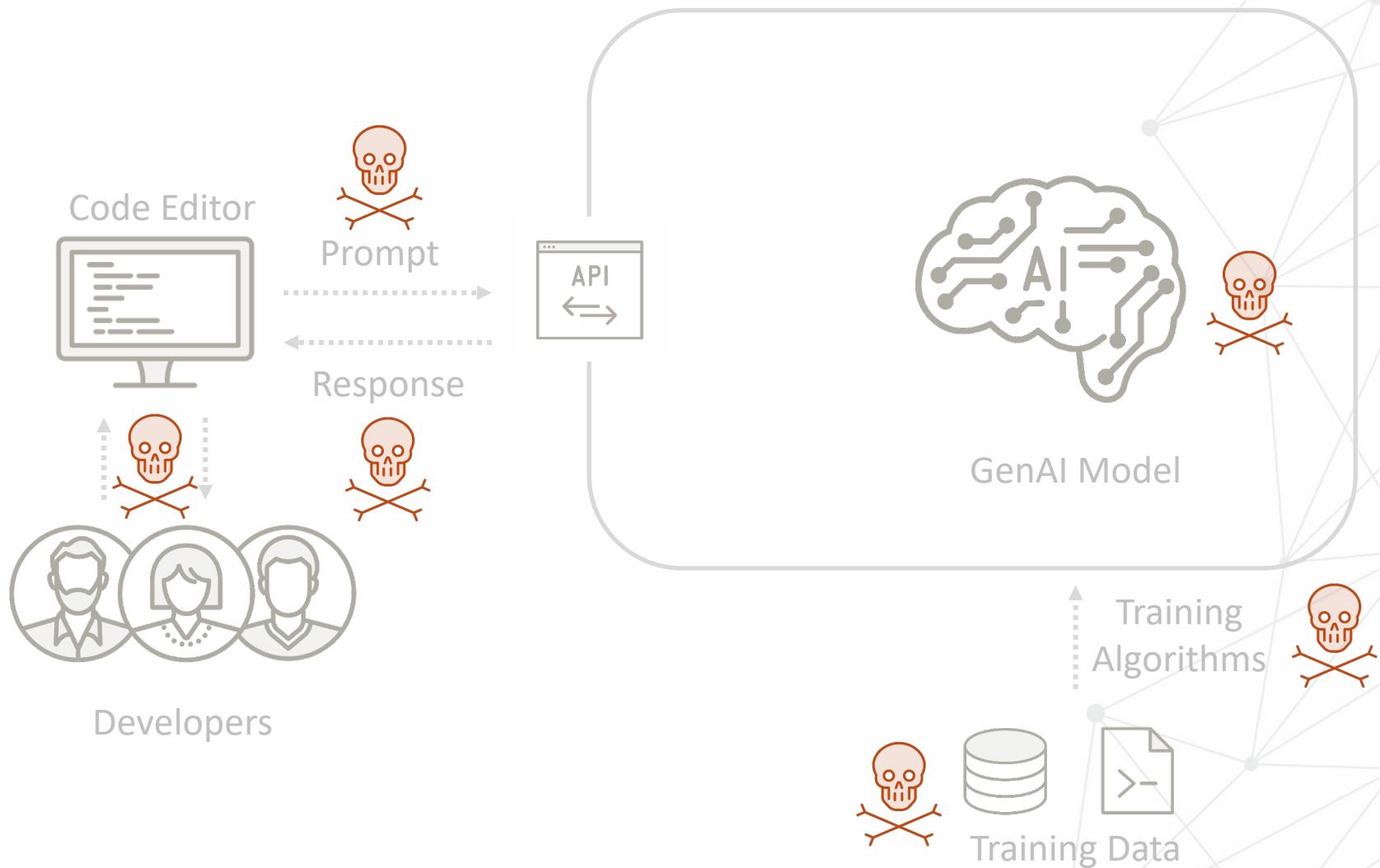
Check the fine prints

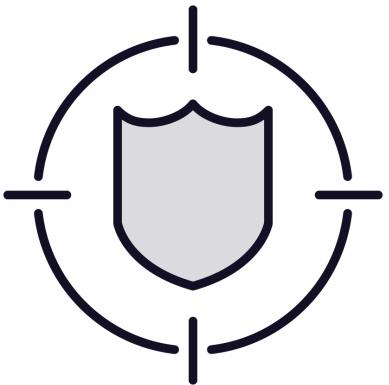
Zero data retention

Regulations and compliance

Manage your own LLMs

# Security Considerations

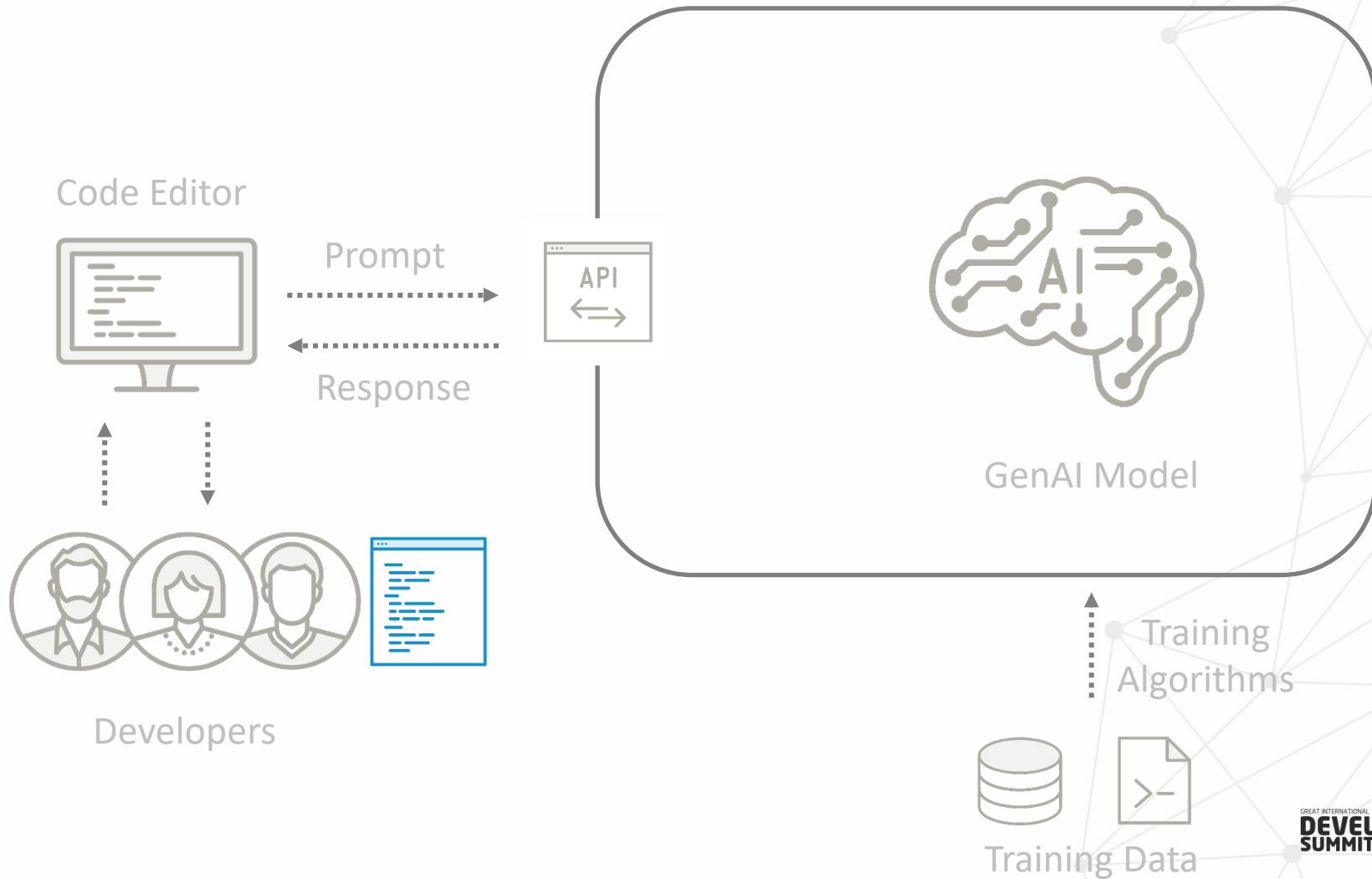




## Managing Security Considerations

- Encryption at rest
- Encryption in motion
- Access Control
- Input and Output guardrails

# Copyright Consideration



# Who Owns the Code?



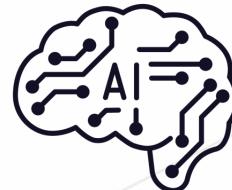
Training Data  
Owner



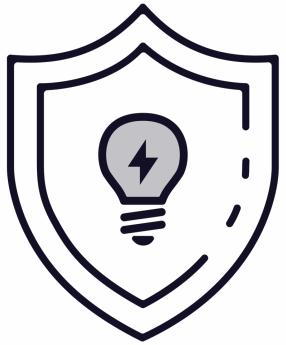
GenAI Model  
Creator



Prompt User



AI Model



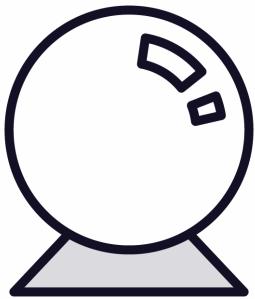
## Managing IP Considerations

Check training data for models

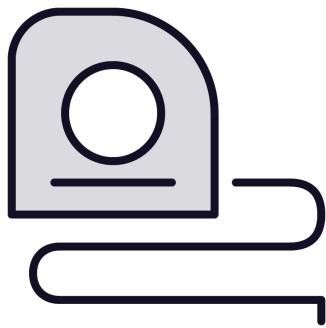
License impact

Involve legal teams

# Personalization Consideration



Results are not  
Predictable



Prompt length is  
limited



Model response  
Quality not adequate

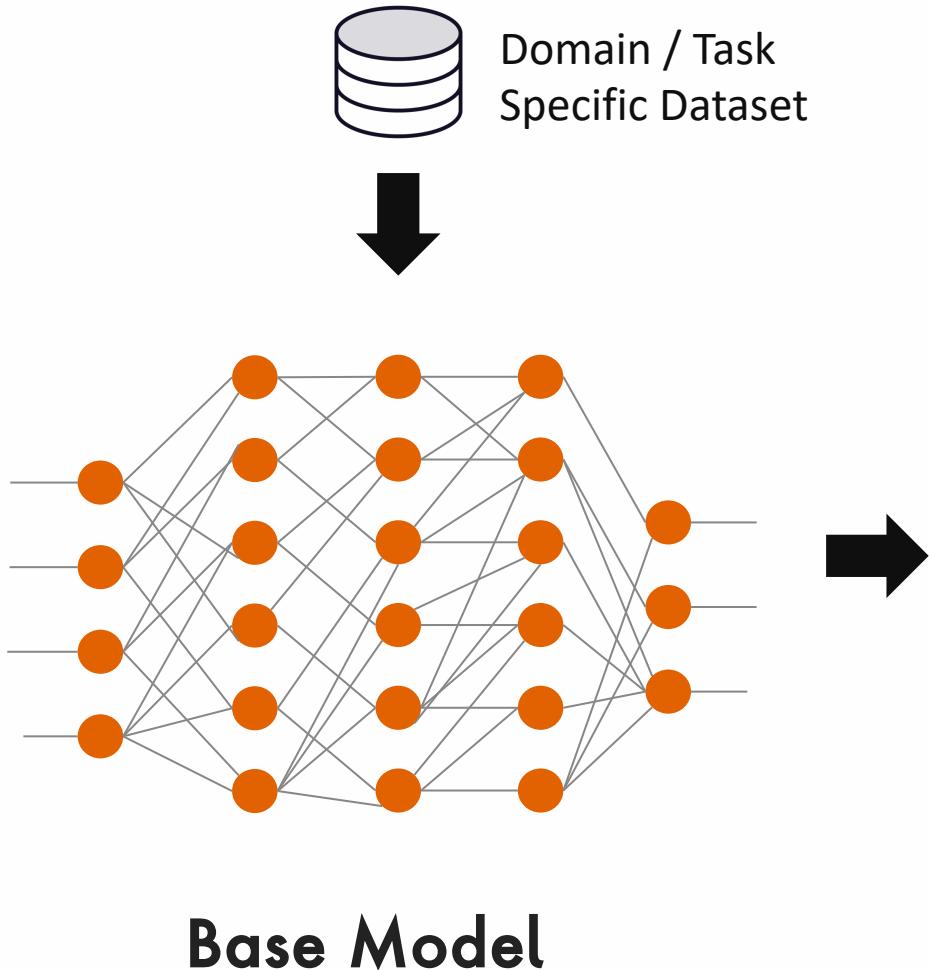


Articulating task as  
prompt not easy

Section 03

# OPTIMIZING AND FINE-TUNING GENERATIVE MODELS FOR SOFTWARE SOLUTIONS

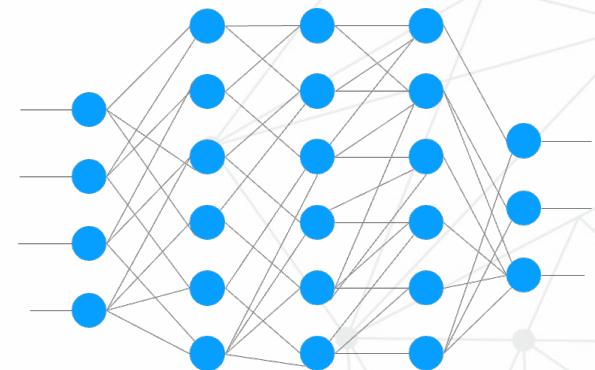
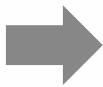
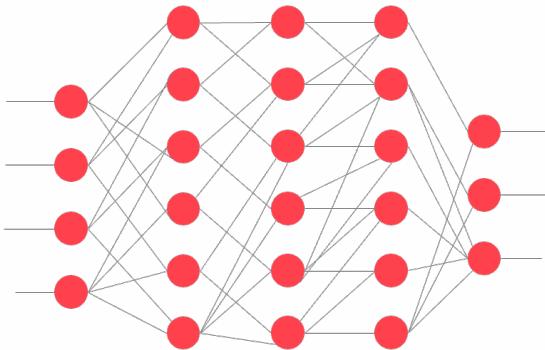
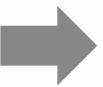
# Fine Tuning



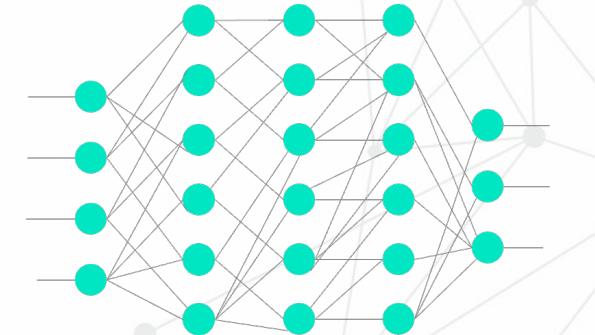
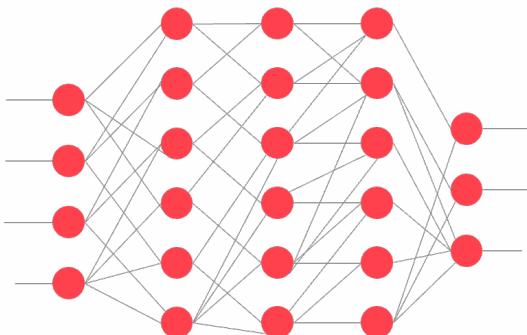
# Fine Tuning



Task 1



Task 2



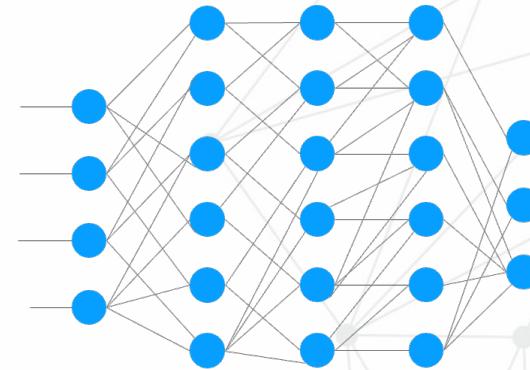
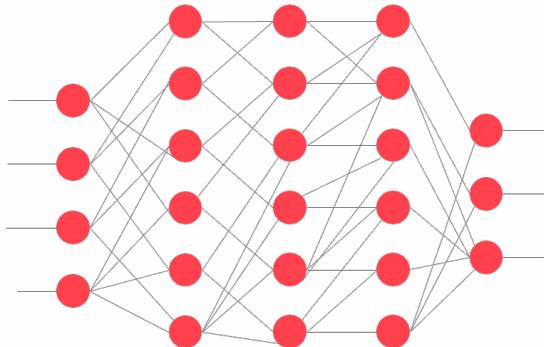
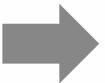
**Base  
Model**

**Fine-Tuned  
Models**

# Fine Tuning

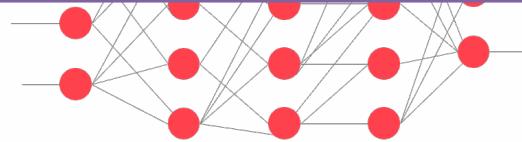


Task 1



Training - Time Consuming and Costly  
Deployment - More Space and Memory

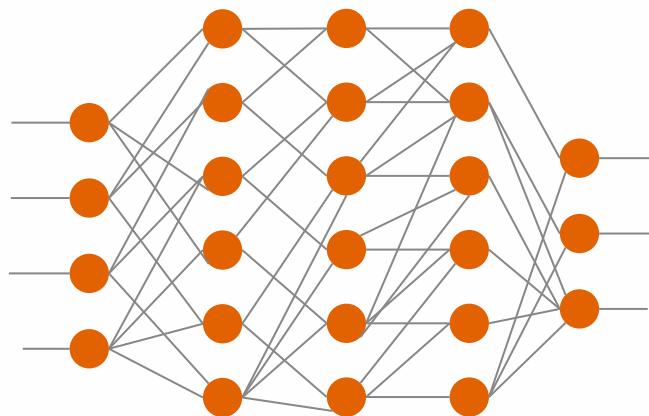
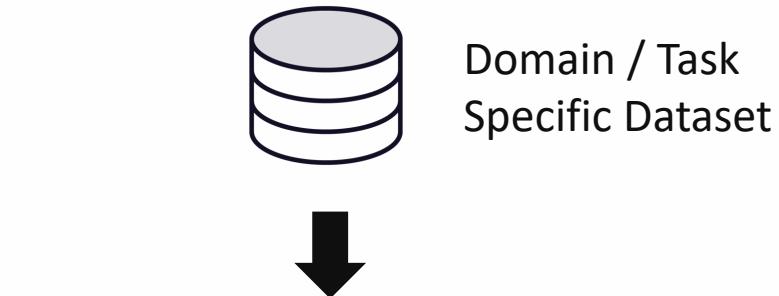
Task 2



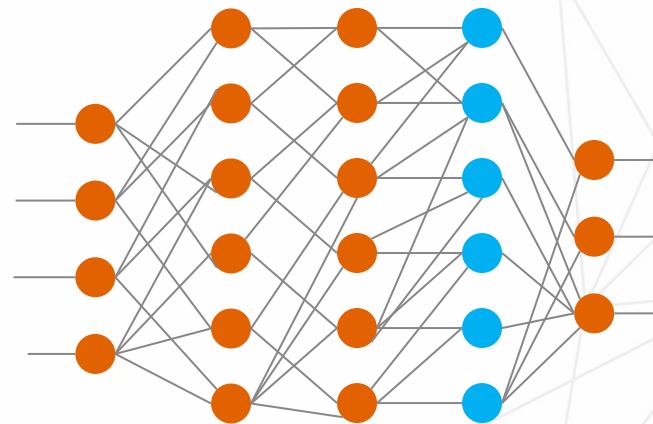
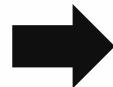
**Base  
Model**

**Fine-Tuned  
Models**

# Parameter Efficient Fine-Tuning (PEFT)

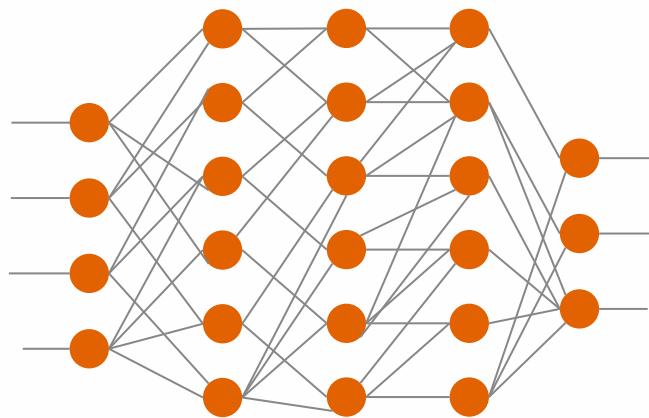


**Base Model**

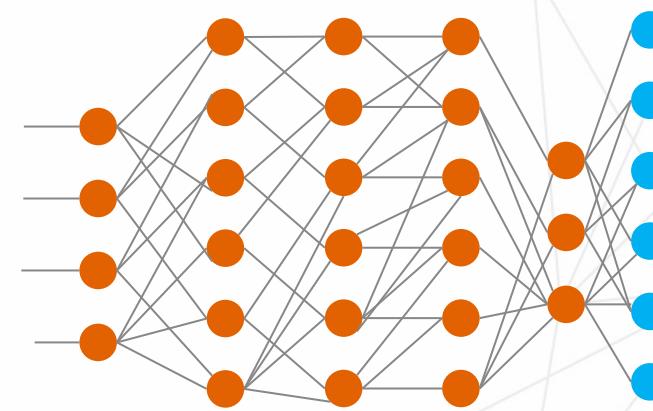
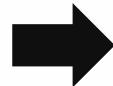


**Fine-Tuned Model**

# Parameter Efficient Fine-Tuning (PEFT)

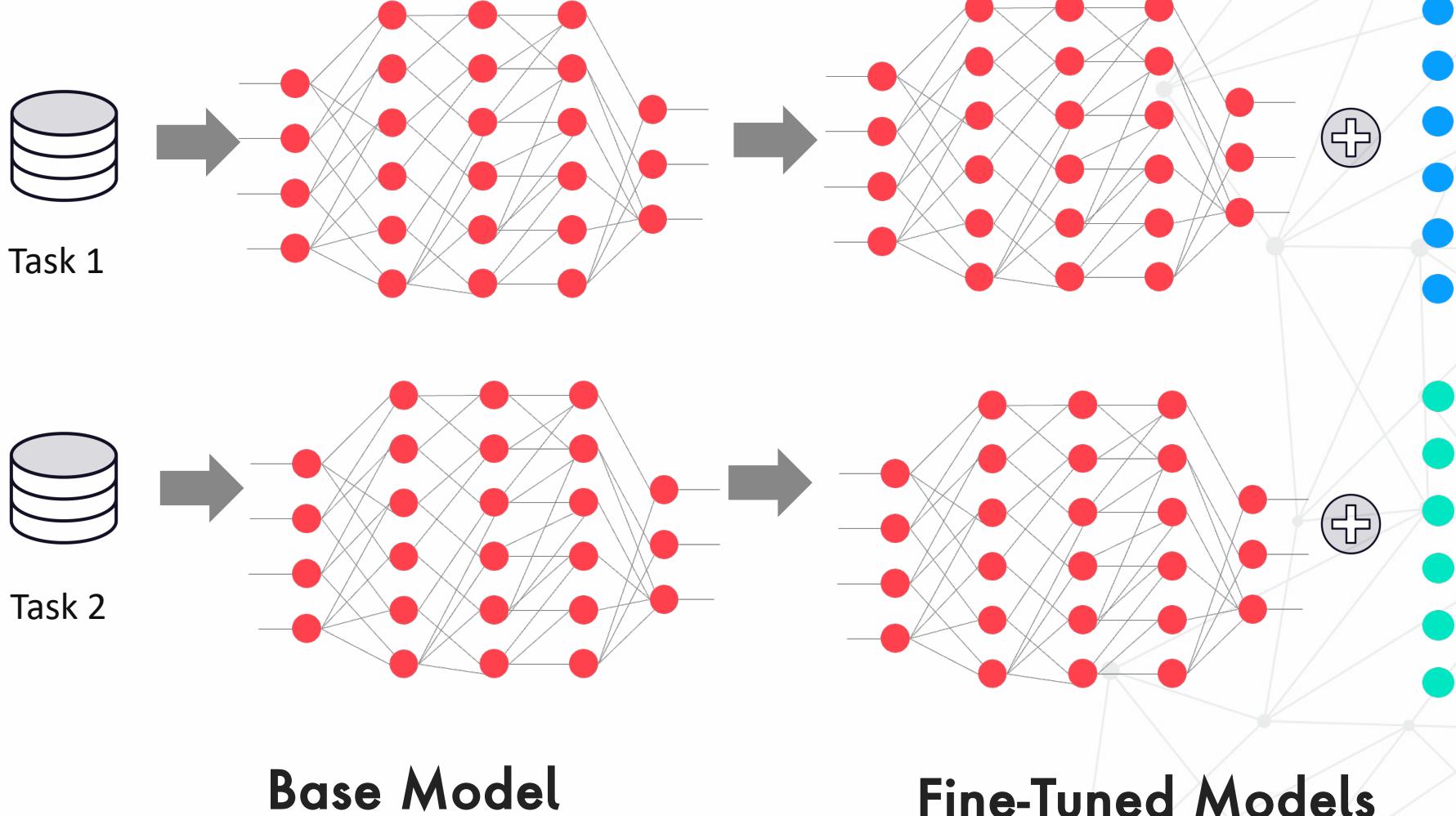


**Base Model**



**Fine-Tuned Model**

# Parameter Efficient Fine-Tuning (PEFT)



Fine-Tuning with PEFT techniques like  
LoRA, Q-LORA can **save ~98% cost**



**unsloth**

**Finetune Mistral, Gemma, Llama 2-5x faster  
with 80% less memory!**

Section 04

# CHALLENGES AND SOLUTIONS IN DEPLOYING GENERATIVE MODELS



# Local LLMs



`ollama pull llama3`

`ollama run llama3`

`ollama create <MODEL> -f  
modelfile`

# Self-Hosted LLMs

## ◆ Text Generation Inference



Predibase Lorax

High throughput

Multi GPU

Optimized batching

Streaming Support

Section 05

# FUTURE TRAJECTORY OF GENERATIVE AI IN SOFTWARE DEVELOPMENT AND ARCHITECTURE

# Trend # 1 : Open-Source Models Becoming Better every single passing day

Closed Source

GPT-4

Claude

Gemini

Open Source

Code Llama

Starcoder

Wizard coder

Llama 3

## Trend # 2 : Smaller models are becoming powerful

LLama3-8b at par

with Llama2-70b

Microsoft-Phi3

(medium - 14b) at

par with GPT 3.5 T

## Trend # 3 : Agentic world to rule the future

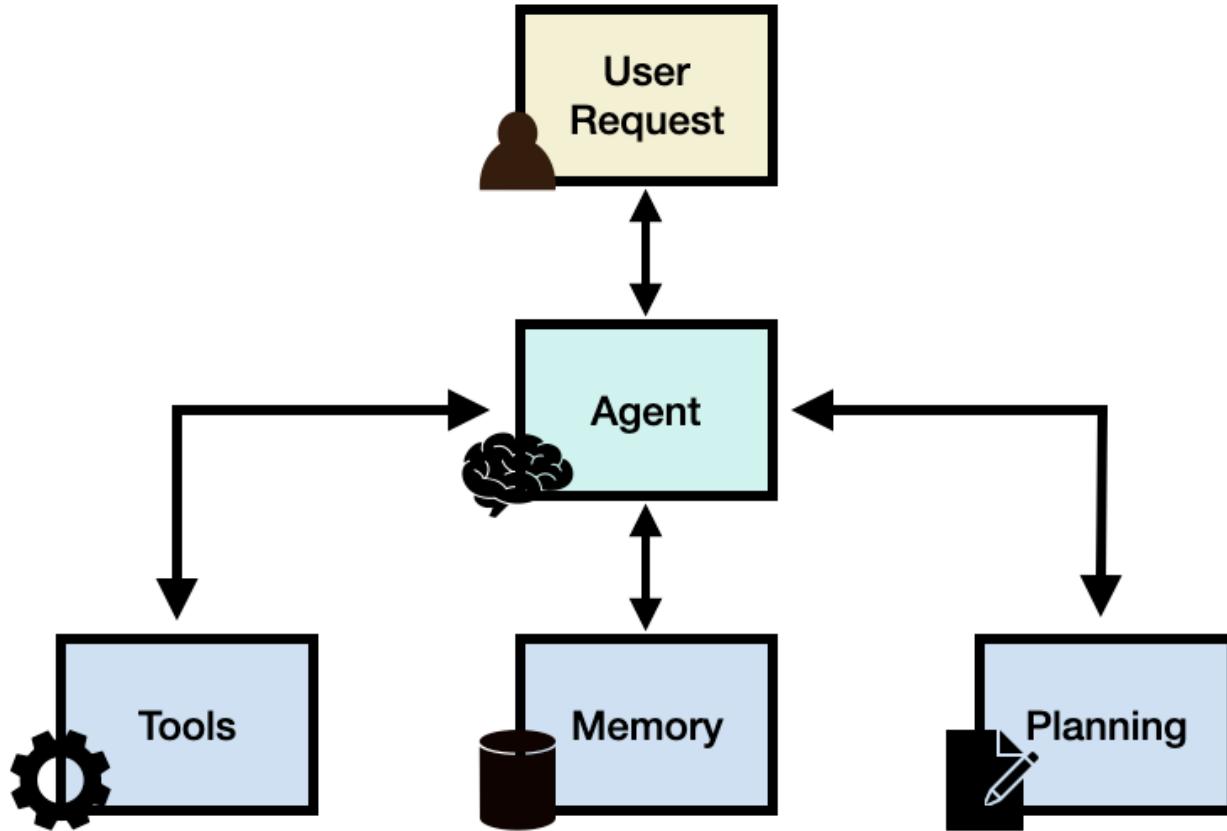
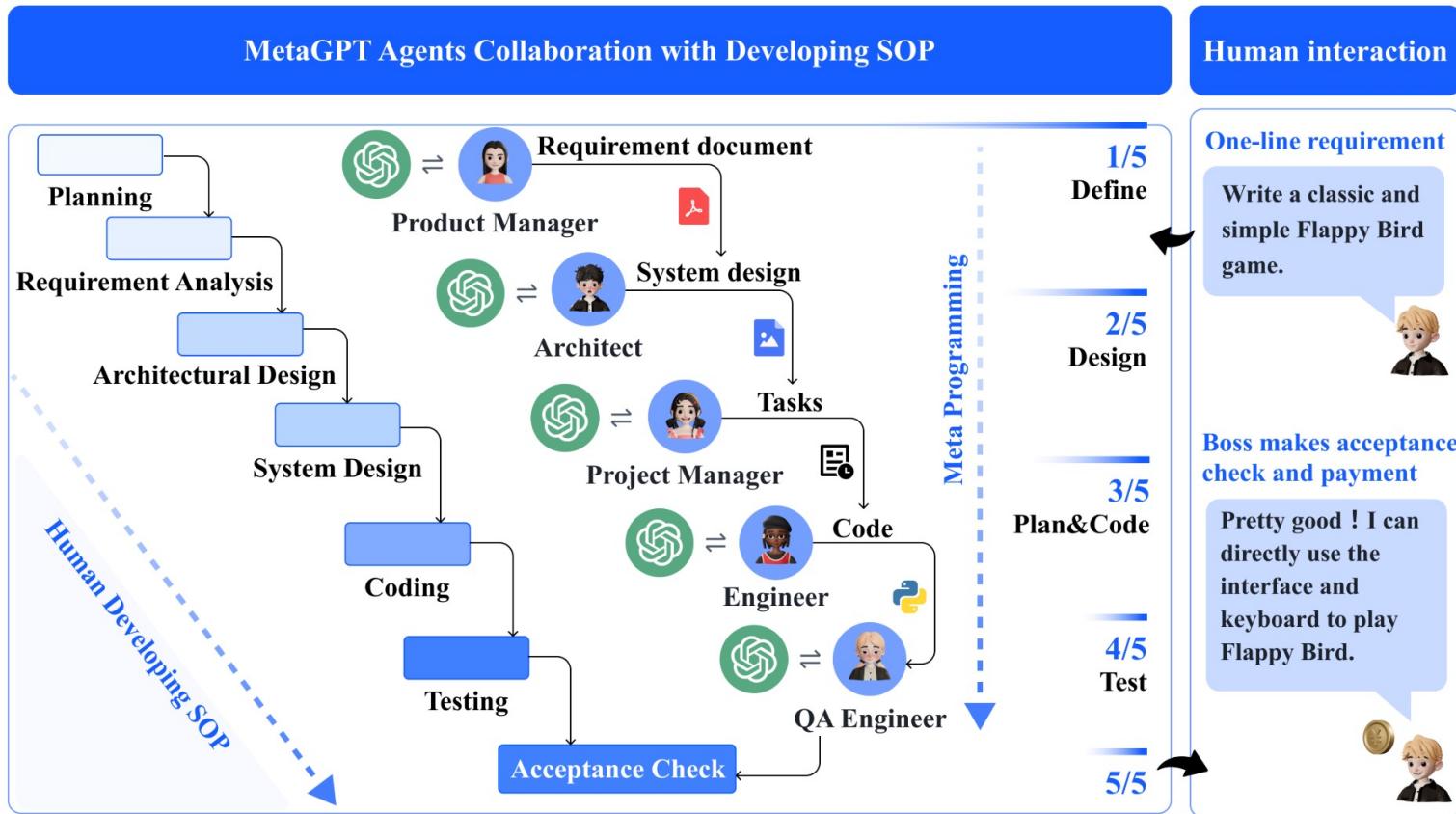


Image Source : <https://www.promptingguide.ai/research/llm-agents>

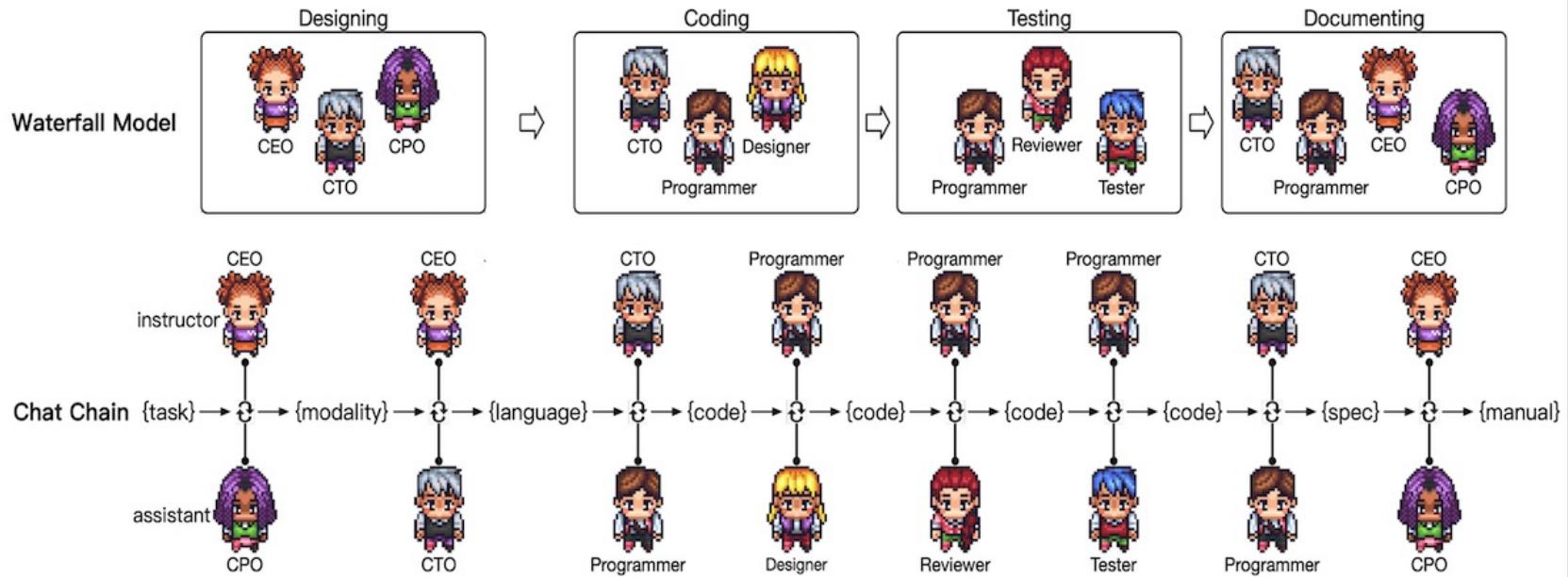
# Multi-Agent Collaboration



Paper : METAGPT: META PROGRAMMING FOR A MULTI-AGENT COLLABORATIVE FRAMEWORK

# Multi-Agent Collaboration

## Agentic Design Patterns: Multi-Agent Collaboration



*Proposed ChatDev architecture. Image adapted from “Communicative Agents for Software Development,” Qian et al. (2023).*

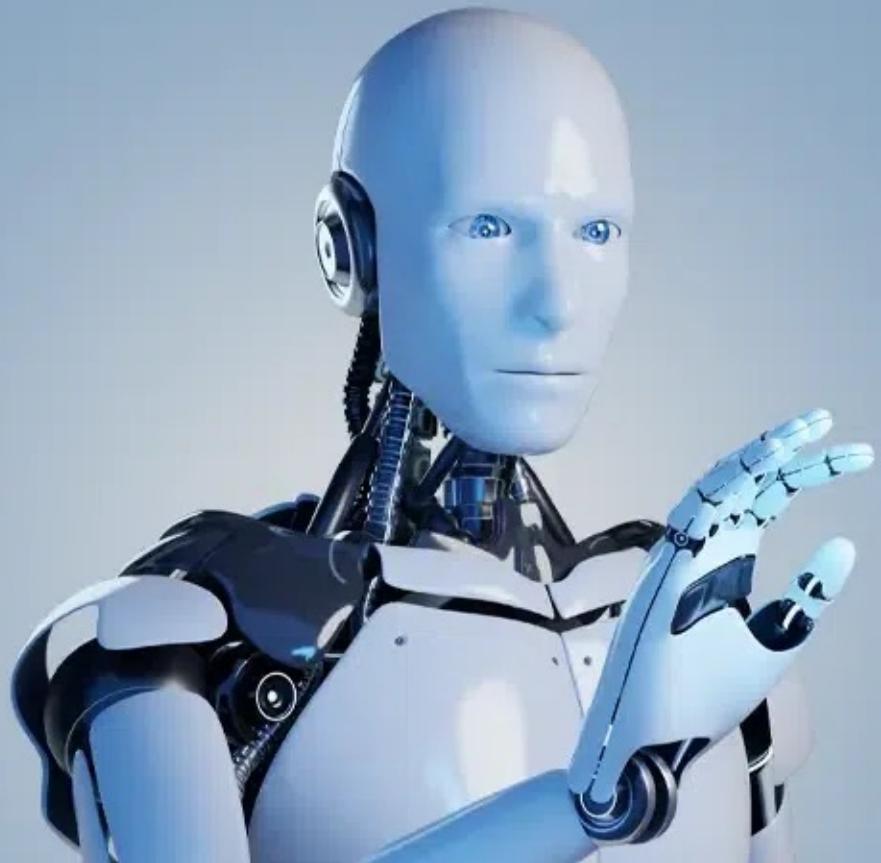
Source: <https://www.deeplearning.ai/the-batch/issue-245/>



INTRODUCING

# Devin AI

World 1st AI Software Engineer



Source : Cognition Labs



INTRODUCING

---

# DEVIKA

## AI Software Engineer



# OpenDevin



22K Stars

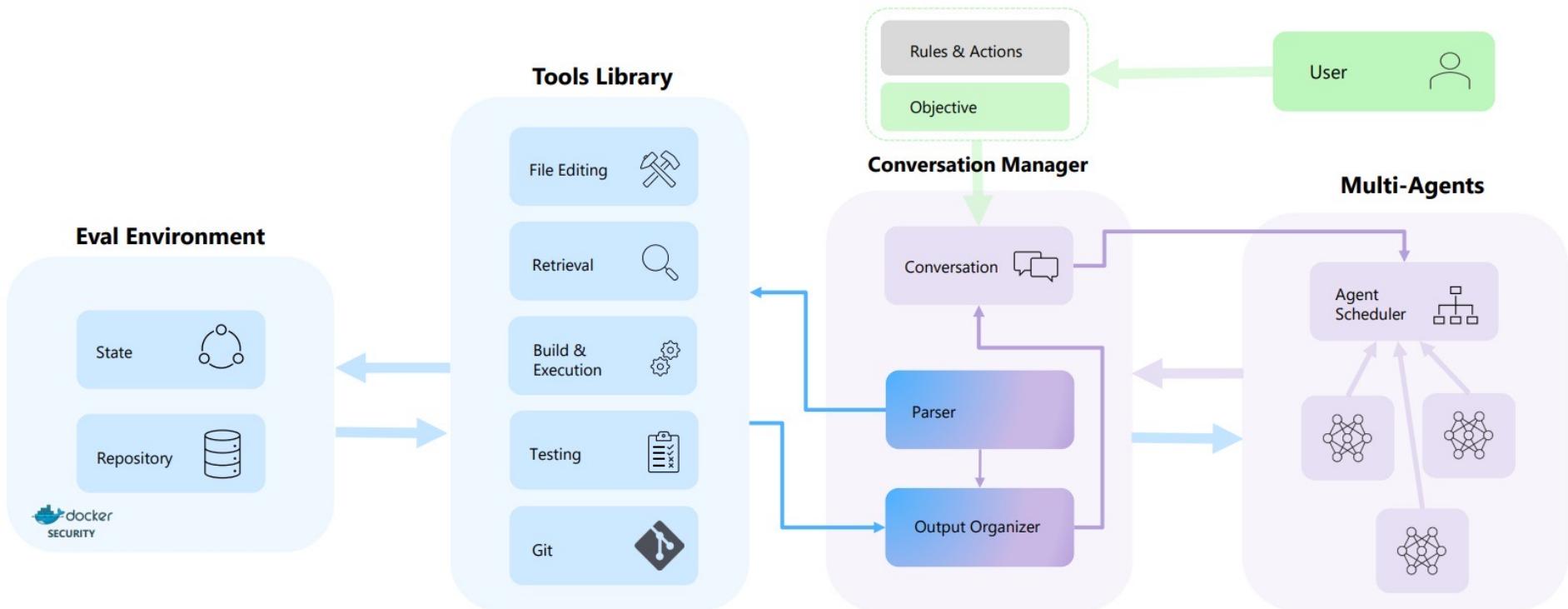
<https://github.com/OpenDevin/OpenDevin>

```
# Your OpenAI API key, or any other LLM API key
export LLM_API_KEY="sk-..."

# The directory you want OpenDevin to modify. MUST be an absolute path!
export WORKSPACE_BASE=$(pwd)/workspace

docker run \
  -e LLM_API_KEY \
  -e WORKSPACE_MOUNT_PATH=$WORKSPACE_BASE \
  -v $WORKSPACE_BASE:/opt/workspace_base \
  -v /var/run/docker.sock:/var/run/docker.sock \
  -p 3000:3000 \
  --add-host host.docker.internal=host-gateway \
  ghcr.io/opendevin/opendevin:0.3.1
```

# AutoDev - Microsoft Paper



Source : <https://visualstudiomagazine.com/Articles/2024/03/20/autodev.aspx>

# Best Practices of Leveraging GENAI in SDLC

- 1 Stay realistic and agile
- 2 Keeping Human in the loop
- 3 Policies and guidelines, Codify to the extent
- 4 Drive adoption by sharing, celebrating and creating snowball effect
- 5 Be careful when sharing data, know the fine-prints

# Remember our purpose

# Delivering faster value

**“AI will not replace  
you,**

**“AI will not replace  
you, but someone  
using AI will.”**

GREAT INTERNATIONAL  
**DEVEL  
SUMMIT**



[www.developersummit.com](http://www.developersummit.com)

Connect with me on  
LinkedIn (Add a note)



publicis  
sapient