In28 Minutes

Generative AI
with
OpenAI & ChatGPT

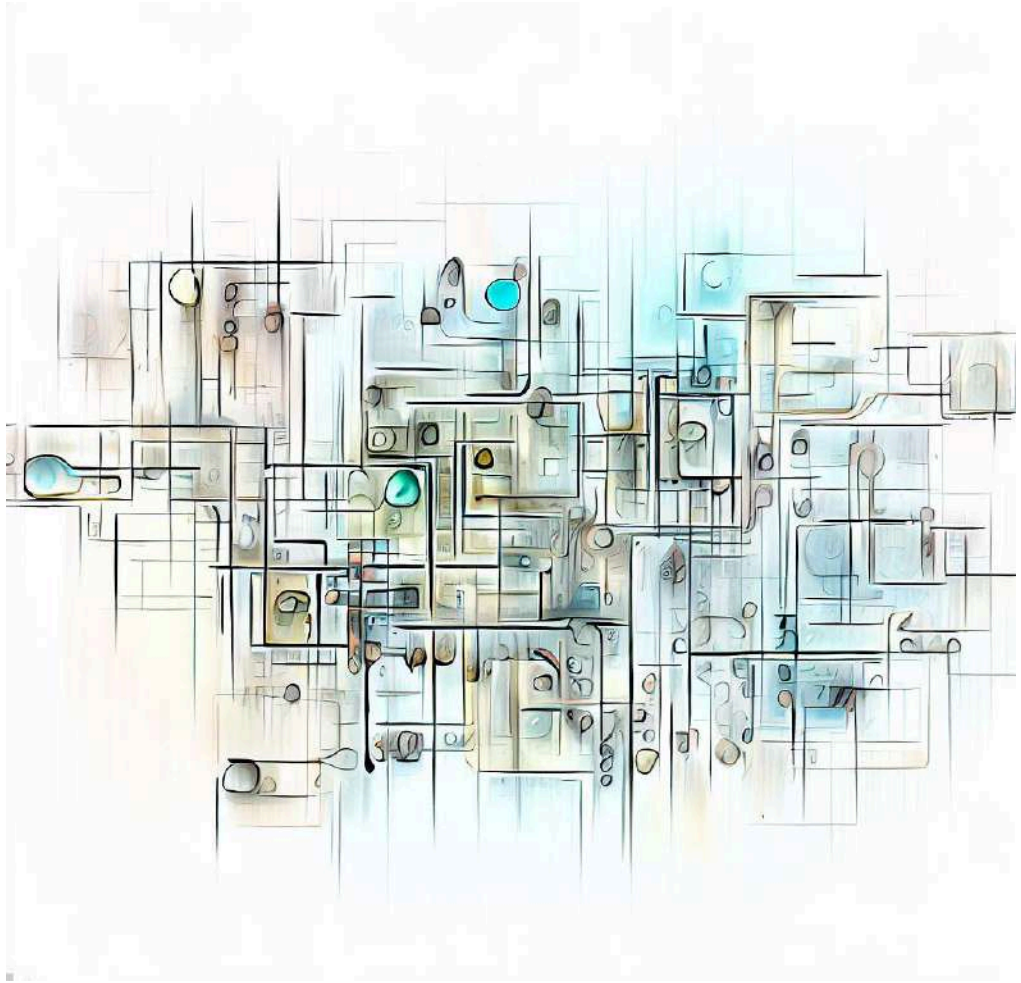
Getting Started with Generative AI

In 28 Minutes

- Artificial Intelligence is in its **second golden phase** (IMHO):
  - **First Phase** - Early successes in the 1960s and 1970s
  - **Second Phase** - Last two decades or so driven by:
    - New algorithms
    - Massive datasets
    - Powerful computing hardware

Getting Started with Generative AI - 2

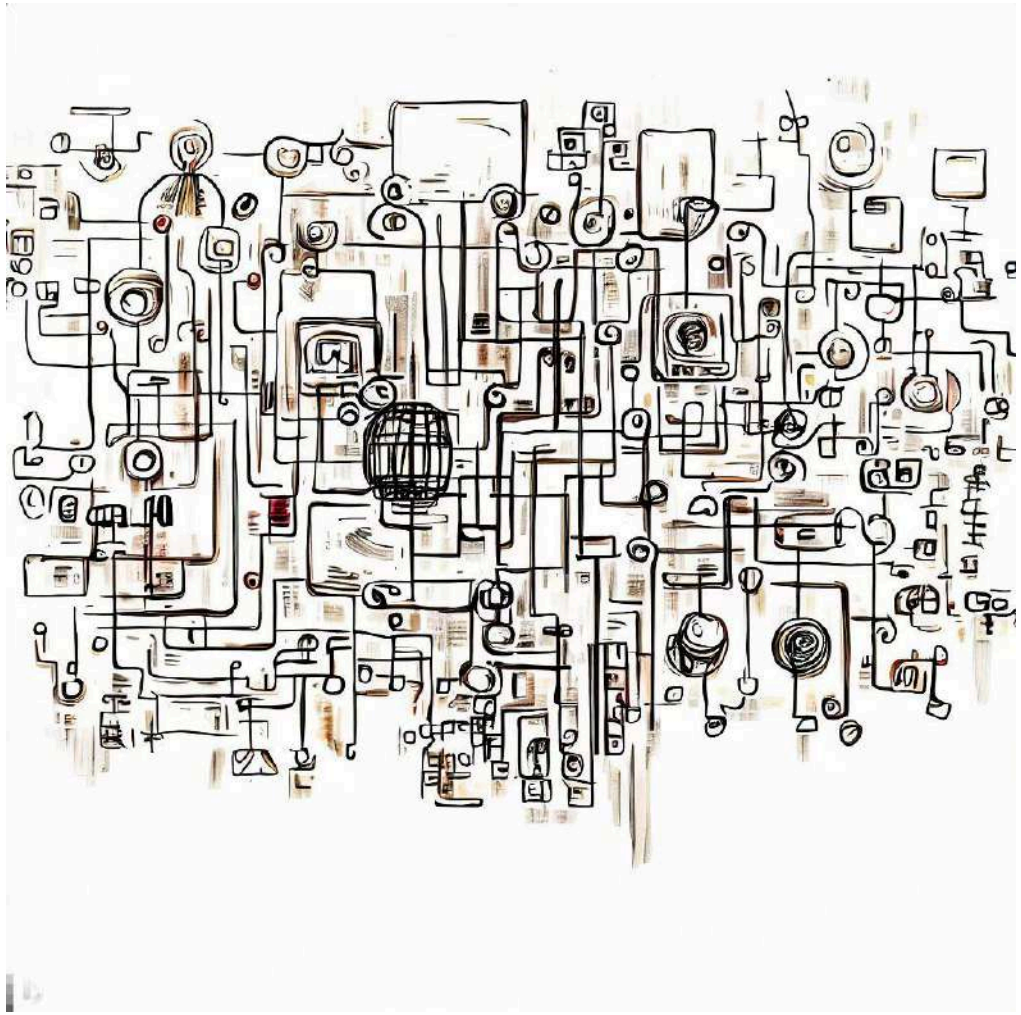
- HOWEVER, **building AI solutions has NOT been easy**:
  - Scarcity of skills
  - Need to create/manage massive datasets
  - Complex infrastructure needs
  - Complexity of managing AI models



Getting Started with Generative AI - 3

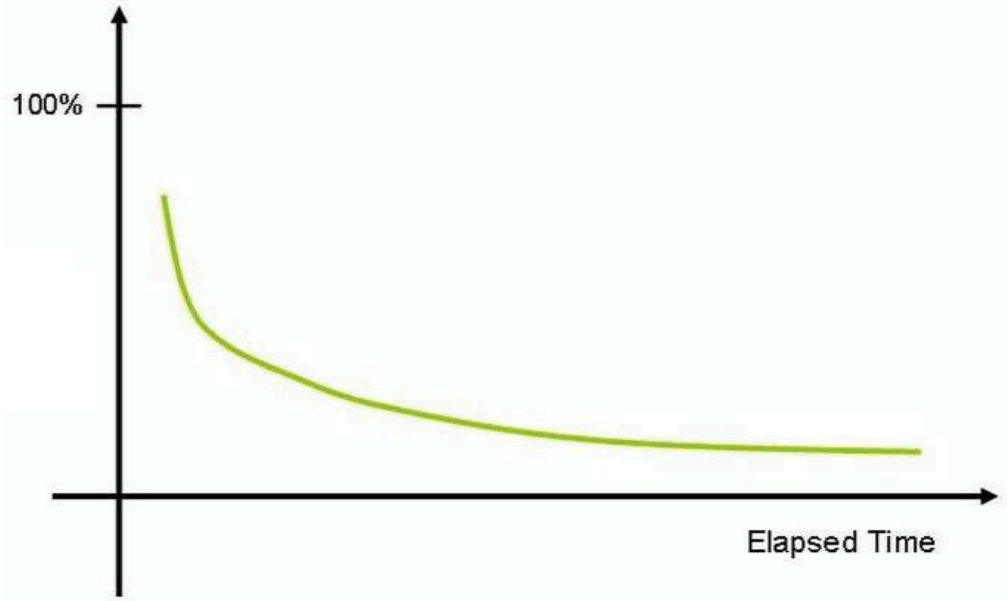
In 28 Minutes

- How can you **use AI WITHOUT AI skills**/complex infrastructure?
  - Use low-code AI platforms (AutoML)
  - Use pre-trained models (especially Generative AI models)
  - OpenAI offers a plethora of solutions **around Generative AI**:
    - **ChatGPT**: Most famous Chatbot
    - **OpenAI API**: Easy to consume APIs
    - **OpenAI Playground**: Play with OpenAI APIs
- **Our Goal** : Help you understand AI, ML and Generative AI while exploring Generative AI solutions offered by OpenAI



How do you put your best foot forward?

- **Learning Gen. AI can be tricky**:
  - Lots of new terminology
  - Lots of services
- As time passes, we forget things
- How do you **improve your chances** of remembering things?
  - **Active learning** - think and make notes
  - **Review the presentation** once in a while



Our Approach

- Three-pronged approach to reinforce concepts:
  - Presentations (Video)
  - Demos (Video)
  - **Two kinds of quizzes**:
    - Text quizzes
    - Video quizzes
- (Recommended) Take your time. Do not hesitate to replay videos!
- (Recommended) Have Fun!

FASTEST ROADMAPS

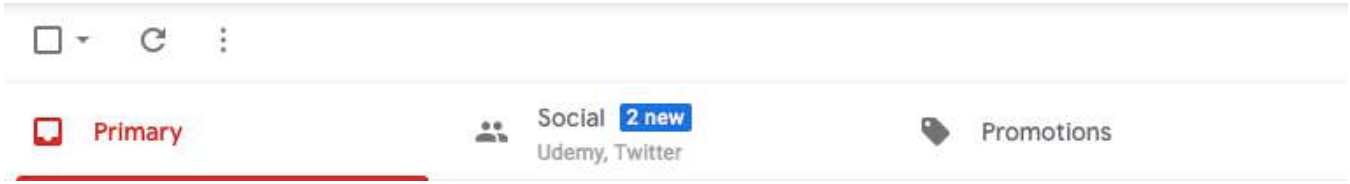
in28minutes.com

In28 Minutes

- Google Cloud Certifications
- Azure Certifications
- AWS Certifications
- DevOps
- Java Full Stack
- Java Microservices

Artificial Intelligence

Artificial Intelligence - All around you



- Self-driving cars
- Spam Filters
- Email Classification
- Fraud Detection

What is AI? (Oxford Dictionary)

In 28 Minutes

The theory and development of computer systems able to perform tasks normally requiring human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages

What is AI? (in Simple Terms)

In 28 Minutes

- **Goal of AI**: Create machines that can simulate human-like intelligence and behavior
  - Play Chess
  - Play Go
  - Make purchase decisions
  - Drive a car
  - Write an essay!

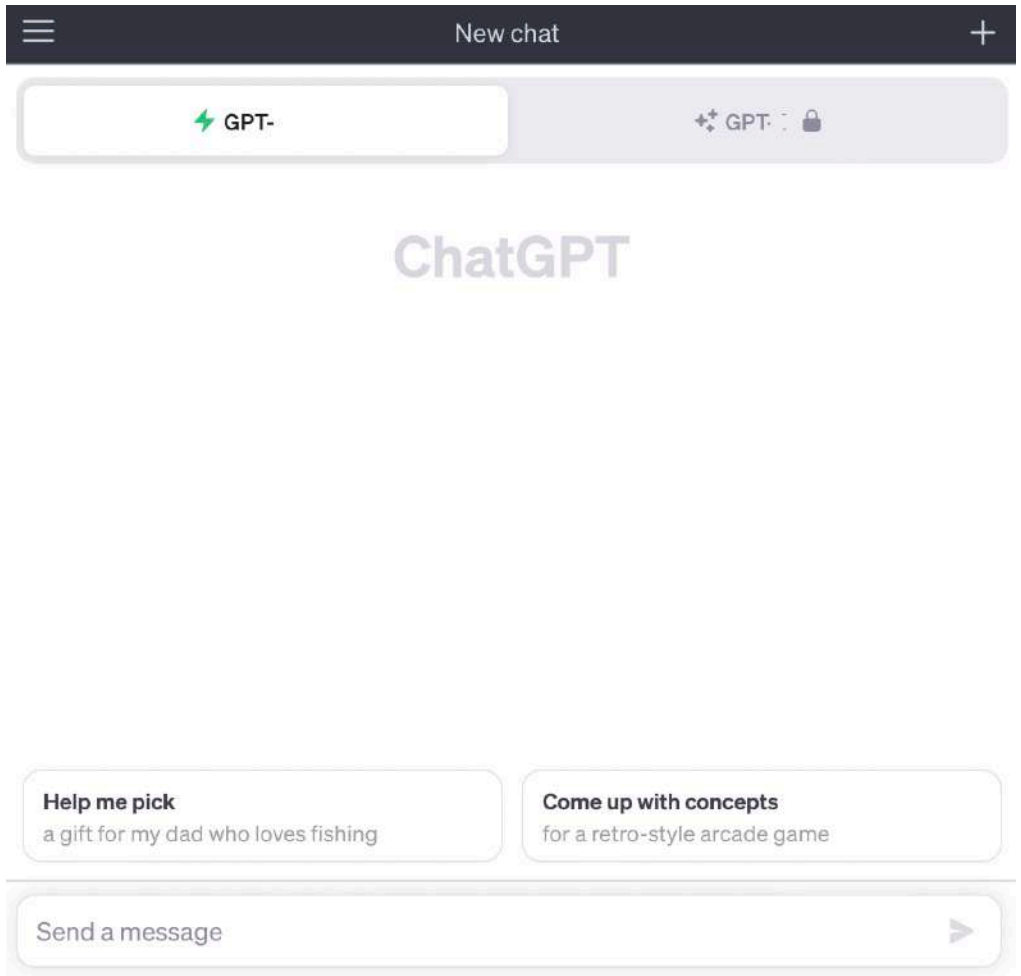
Understanding Types of AI

In 28 Minutes

- **Strong artificial intelligence (or general AI)**: Intelligence of machine = Intelligence of human
  - A machine that can solve problems, learn, and plan for the future
  - An expert at everything
    - Including learning to play all sports and games!
  - Learns like a child, building on it's own experiences
  - We are far away from achieving this!
    - Estimates: few decades to never
- **Narrow AI (or weak AI)**: Focuses on specific task
  - Example: Self-driving car
  - Example: Playing Chess
  - Example: Predicting House Price

Playing with ChatGPT

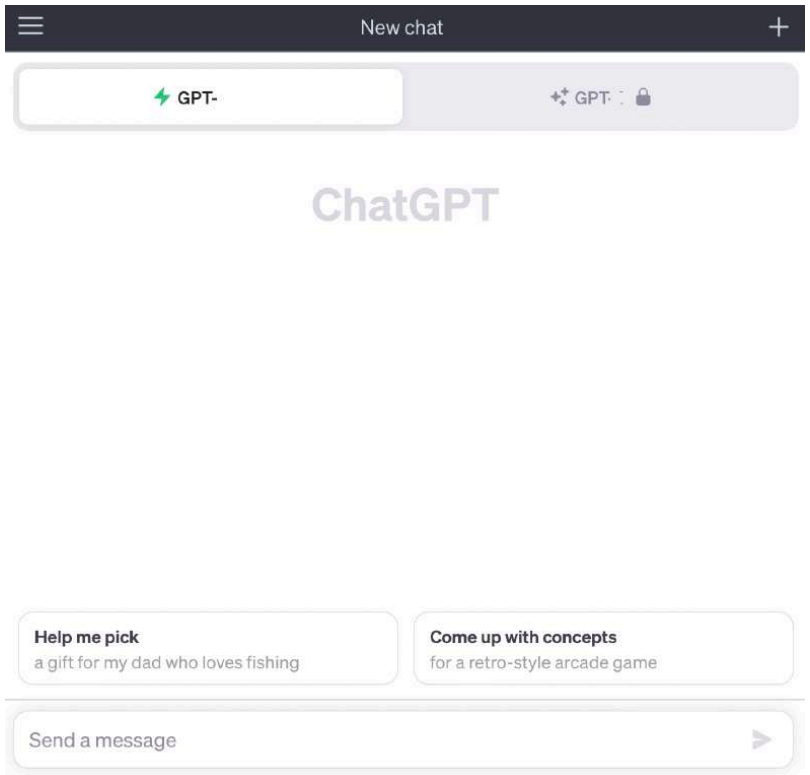
- **ChatGPT**: OpenAI's Generative AI Chatbot!
- **A Demo of ChatGPT**:
  - You are **Lex Friedman**. You are going to interview **Sachin Tendulkar** tomorrow. What are the FIRST FIVE questions that you are going to ask?
  - Act as **Sachin Tendulkar**. You are meeting **Roger Federer**. What questions would you ask?
  - Generate a **bulleted list of items** I need for an 15 day Everest Base Camp trek
    - I will be staying in tea houses on the trek. Can you update the list?



Playing with ChatGPT - Coding, Learning and Design

In 28 Minutes

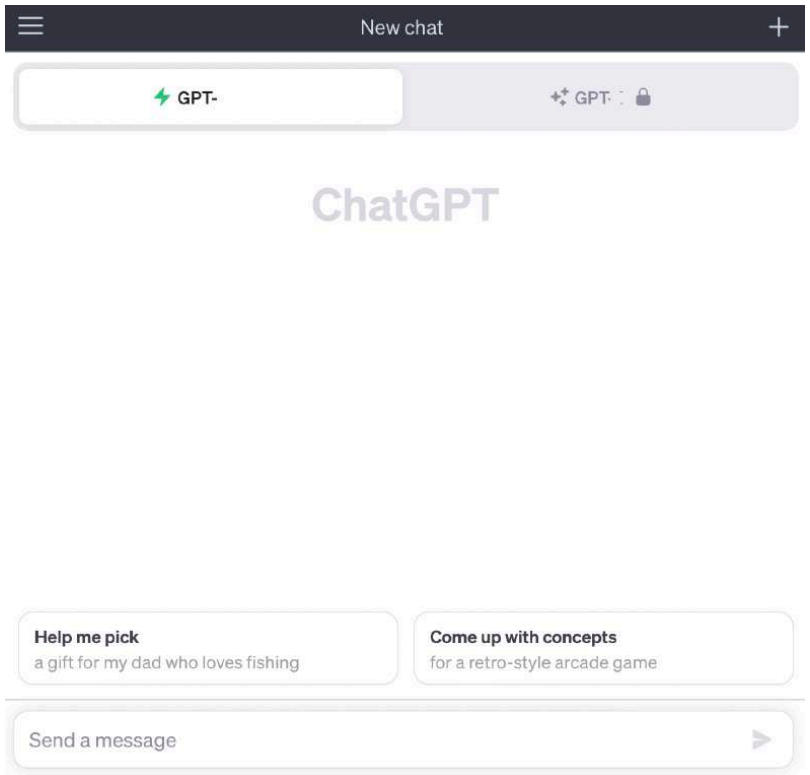
- Write **a Python function** to determine if a year is a leap year
- I'm learning Python for loop. Give me **a few exercises** to try?
- Can you **design a REST API** for todos? Give me an example request and response for each.
- I want to store information about courses, students, enrollments and reviews **in a relational table**. Can you suggest a structure?
- I like **learning concepts using a lot of examples**. What would be the books you would recommend to learn Design Patterns?



Playing with ChatGPT - Exploring Technology

In 28 Minutes

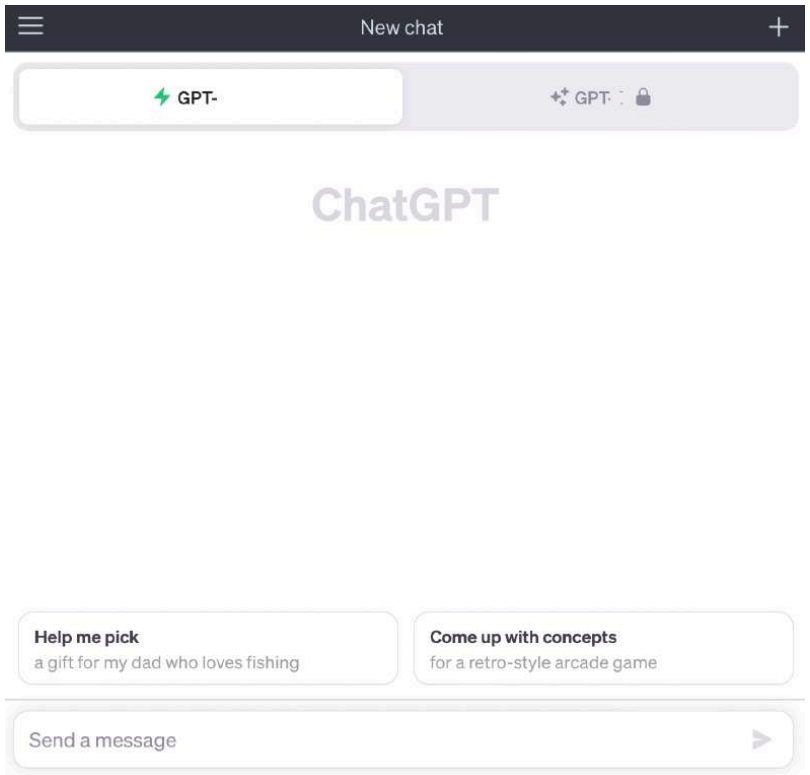
- Can you make list of **Top 10 technologies** that I might want to learn as a cloud engineer?
- For a new project, I'm considering React and Angular as front end frameworks.
  - Can you **compare them and present the results in a tabular format**? Feature/Factor in the column and the framework in the row.
- I like to **learn in a step-by-step approach** by breaking down complex concepts into smaller, more manageable parts.
  - How can I learn Docker? Give me a list of 10 step by step exercises I can begin with. Make sure you order them in increasing order of difficulty. Present the results in a table.



Playing with ChatGPT - Generating Ideas

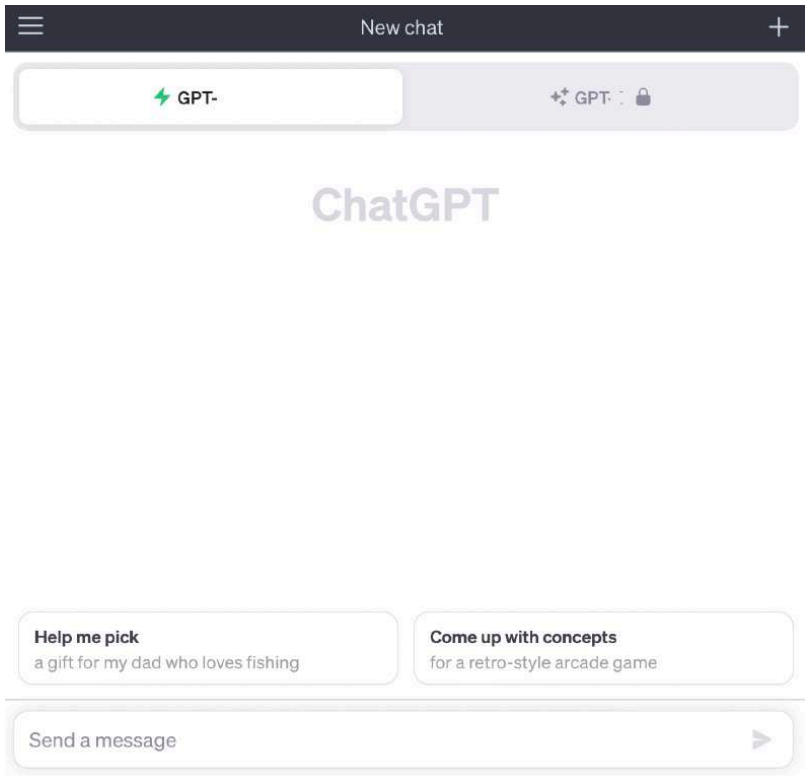
In 28 Minutes

- I want to **Understand DevOps and start exploring different DevOps Tools**. Can you suggest a few prompts I can start with?
- How can I **make it easy for a beginner** to learn a new programming language / tool / technology / framework / methodology? One example is to **use analogies**. Can you think about other similar things I can do. Make sure you order them in decreasing order of importance. Rate importance from 1-100. 1 being least important. Present the **results in a table**.



Playing with ChatGPT - Observations

- ChatGPT **MAY display** inaccurate or offensive information
- BUT it does **provide a lot of value** if you understand its limitations and know when/how to use it
- **How does ChatGPT work?**
  - Artificial Intelligence
  - Machine Learning
  - Generative AI
  - Large Language Models
  - Foundation Models



AI vs ML vs Generative AI

In 28 Minutes

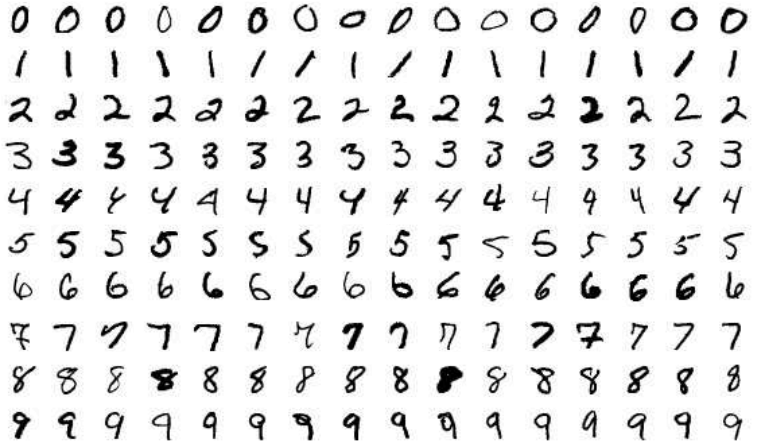
- **Goal of AI**: Create machines that can simulate human-like intelligence and behavior
  - What is ML?
  - How does Generative AI fit in?
- Let's get started on a Journey!

Machine Learning vs Traditional Programming

In 28 Minutes

- **Traditional Programming**: Based on Rules
  - IF this DO that
  - Example: Predict price of a home
    - Design an algorithm taking all factors into consideration:
      - Location, Home size, Age, Condition, Market, Economy etc
- **Machine Learning**: Learning from Examples (NOT Rules)
  - Give millions of examples
  - Create a Model
  - Use the model to make predictions!

Home size (Square Yds)	Age	Condition (1-10)	Price $$$
300	10	5	XYZ
200	15	9	ABC
250	1	10	DEF
150	2	34	GHI



Machine Learning Fundamentals - Scenarios

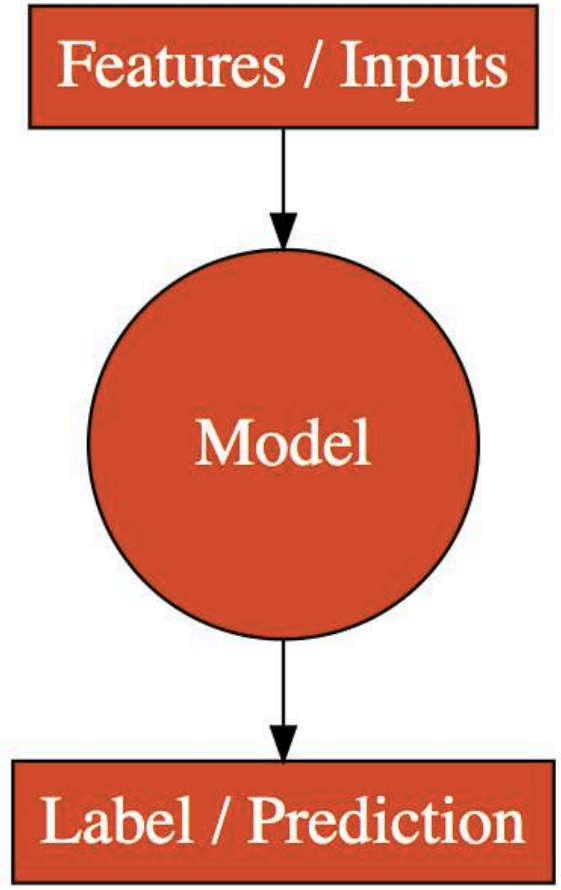
In 28 Minutes

Scenario	Solution
Categorize: Building a computer system as intelligent as a human. An expert at everything (all sports and games!)	Strong AI
Categorize: Building a computer system that focuses on specific task (Self-driving cars, virtual assistants, object detection from images)	Narrow AI (or weak AI)
Category of AI that focuses on learning from data (examples)	Machine learning
How is ML different from traditional programming?	Traditional Programming: Rules. Machine Learning: Examples

Machine Learning - Making Prediction

In 28 Minutes

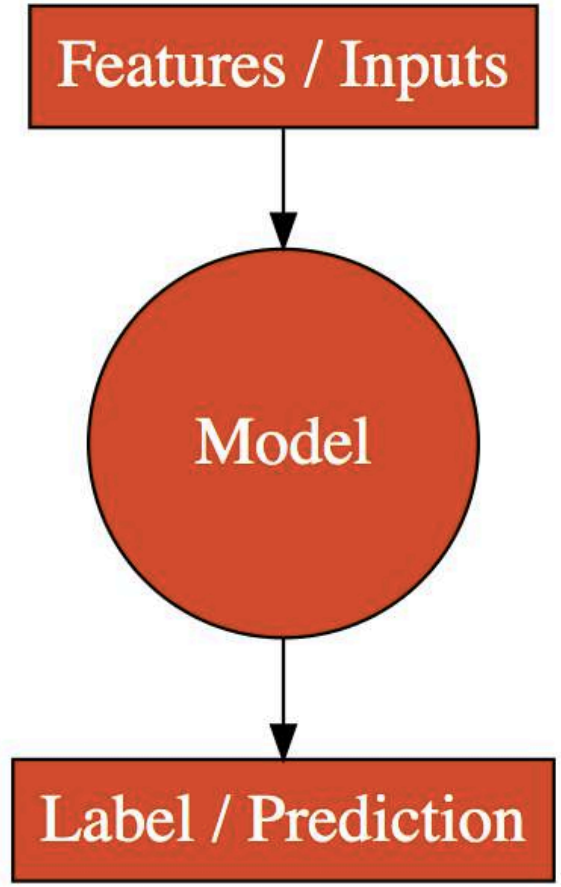
- **Goal**: Make a Good Prediction
  - Give inputs to a model
  - Model returns the prediction
  - Inputs are called **Features**
  - Prediction is called **Label**
  - **Example**: House Price Prediction Model
    - **Label**: price
    - **Features**:
      - area: Total area of house (m^2)
      - rooms: No. of rooms
      - bedrooms: No. of bedrooms
      - furniture: Is it furnished?
      - floor: Which floor?
      - age: How many years?
      - balcony: has balcony or not
      - garden: has garden or not



Machine Learning - Features and Labels - Examples

In 28 Minutes

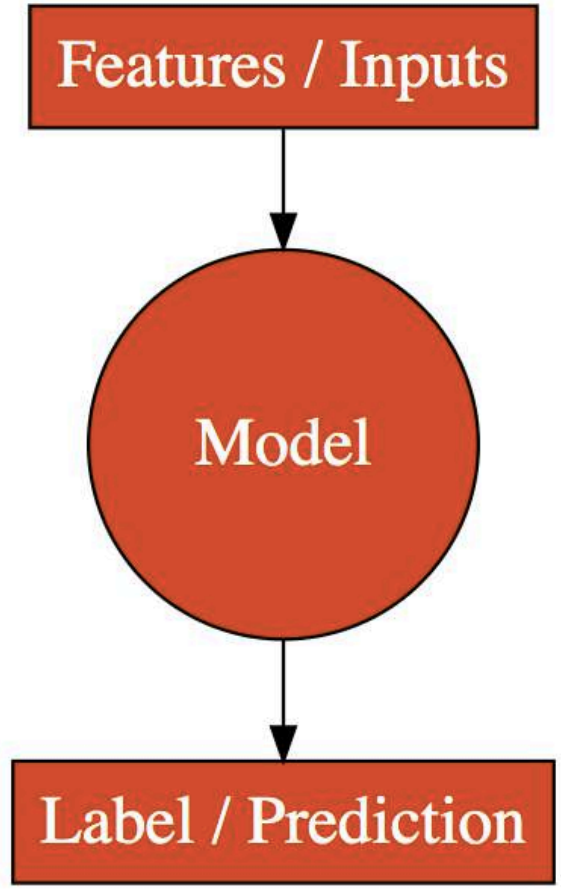
- Used Car Price Prediction Model
  - **Label**: price
  - **Features**: manufacturer, year, model, age, condition, cylinders, location
- Spam Email Classification Model
  - **Label**: isSpam
  - **Features**: sender, subject, content
- Grant a Loan Model
  - **Label**: shouldWeGrantALoan
  - **Features**: doesOwnCar, doesOwnRealEstate, creditScore, isMarried, doesHaveChildren, totalIncome, totalCredit



Machine Learning - Prediction Possibilities

In 28 Minutes

- **Numeric value**: Label is a numeric value with a range of possibilities => **Regression**
  - Used Car Price Prediction
  - House Price Calculation
  - Predicting sea level
  - How much will it rain tomorrow?
- **Limited Possibilities**: YES or NO, 0 or 1, Type 1 or Type 2 or Type 3 => **Classification**
  - Spam Email, Grant a Loan, Determine the type of cloud
  - Will it rain today?
- Summary:
  - **Classification**: Predicting category
  - **Regression**: Predicting numeric value



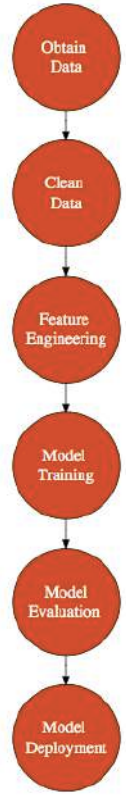
Machine Learning - Making Predictions - Scenarios

Scenario	Solution
Categorize into features and labels for house price prediction: price, area, rooms, age	price is label. Others can be features
Categorize into features and label for used vehicle price prediction: manufacturer, year, model, age, condition, cylinders, location, price	price is label. Others can be features
Categorize: Used Car Price Prediction	Regression
Categorize: Spam Email Identification	Classification
Categorize: Predict amount of rainfall in the next year	Regression
Categorize: Should we grant a loan?	Classification
Categorize: Identify the type of vehicle in an image	Classification
Categorize: Find a specific dance form in a video	Classification

Creating Machine Learning Models - Steps

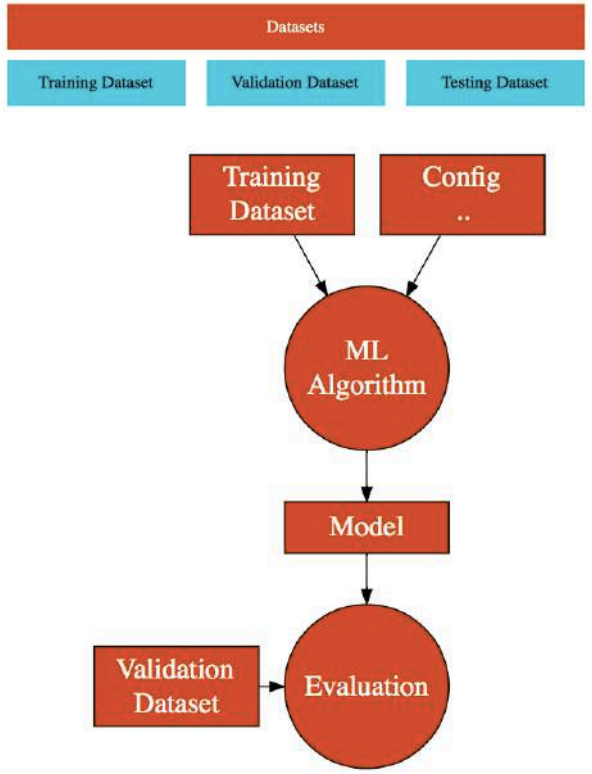
In 28 Minutes

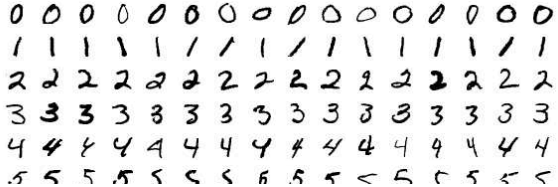
- **1:** Obtain Data
- **2:** Clean Data
- **3:** Feature Engineering: Identify Features and Label
- **4:** Create a Model using the Dataset (and the ML algorithm)
- **5:** Evaluate the accuracy of the model
- **6:** Deploy the model for use



Understanding Machine Learning Terminology

- **Process**
  - **Training**: The process of creating a model
  - **Evaluation**: Is the model working?
  - **Inference**: Using model to do predictions in production
- **Dataset**: Data used to create, validate & test the model
  - **Features**: Inputs
  - **Label**: Output/Prediction
  - Dataset Types
    - **Training Dataset**: Dataset used to create a model
    - **Validation Dataset**: Dataset used to validate the model (and choose the right algorithm) - Model Evaluation
    - **Testing Dataset**: Dataset used to do final testing before deployment





ML Stages and Terminology - Scenarios

Scenario	Solution
Determine Stage: You remove data having null values from your dataset	Clean Data (Data Preparation)
Determine Stage: Normalize or split data into multiple features	Feature Engineering
Determine Stage: You evaluate the accuracy metrics of a model	Model Evaluation
Terminology: Using model to do predictions in production	Inference
Terminology: The process of creating a model	Training
Terminology: Dataset used to (train) or create a model	Training Dataset
Terminology: Dataset used to evaluate a model	Validation Dataset
Terminology: Dataset used to do final testing before deployment	Testing Dataset

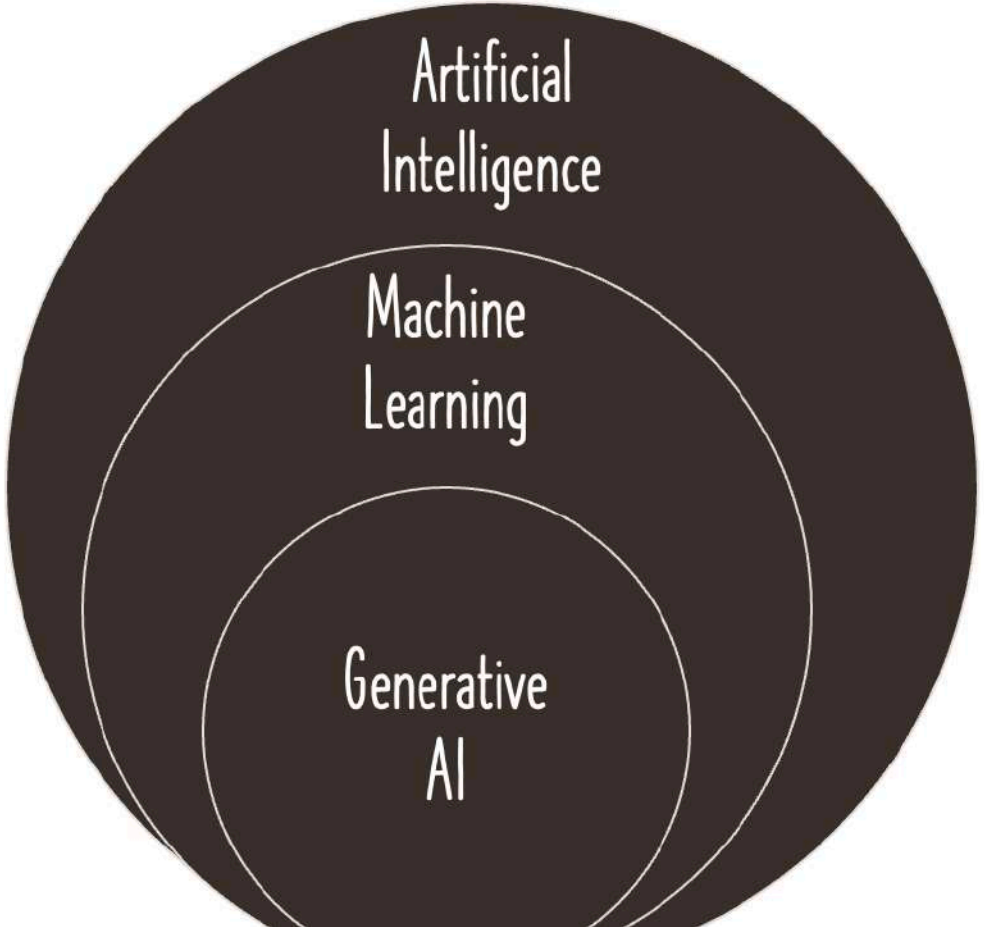
THE AI Turmoil

- **Quotes:**
  - I am really quite close, I am very close, to the cutting edge in AI and it scares the hell out of me - **Elon Musk**
  - The development of full artificial intelligence could spell the end of the human race. It would take off on its own, and re-design itself at an ever-increasing rate. Humans, who are limited by slow biological evolution, couldn't compete and would be superseded. - **Stephen Hawking**
- No one knows the truth:
  - Most predictions about AI turned false in the last few decades!
- What's the pragmatic way to think?
  - Don't fear AI
  - Learn to make the best use of it

Generative AI - How is it different?

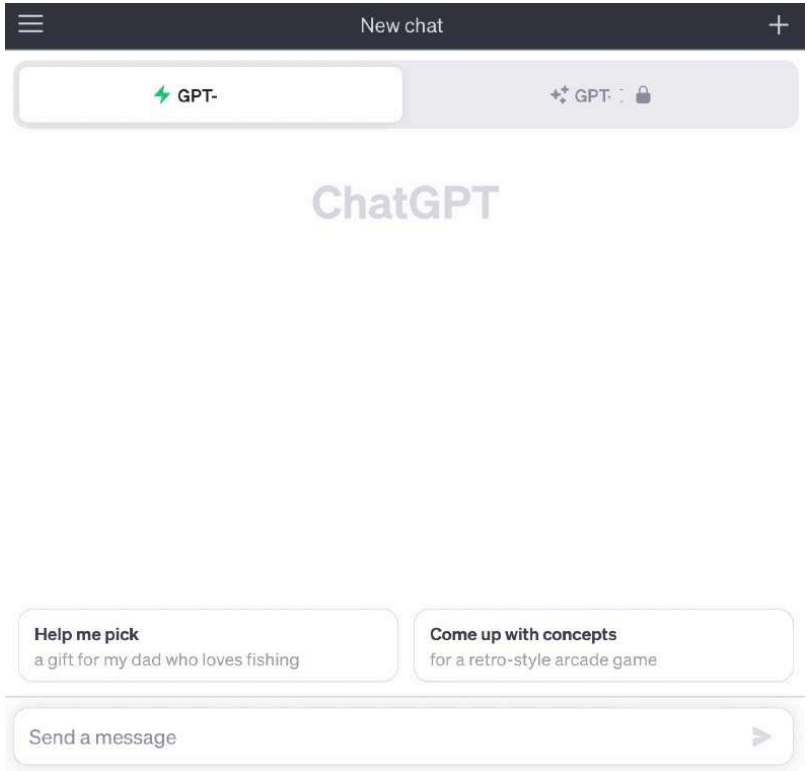
In 28 Minutes

- **Artificial Intelligence**: Create machines that can simulate human-like intelligence and behavior
  - **Machine Learning**: Learning from examples
    - **Generative AI**: Learning from examples to create new content



Generative AI - Generating New Content

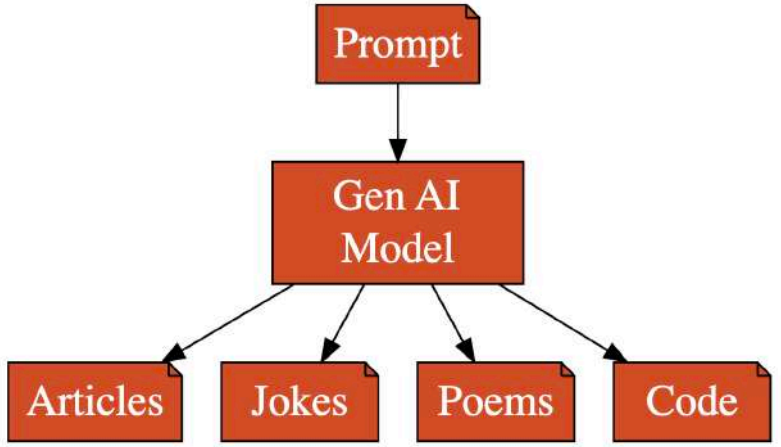
- **Goal**: Generating New Content
  - Instead of making predictions, Generative AI focuses on creating new data samples
  - **Examples**:
    - **Text Generation**: Writing e-mails, essays & poems. Generating ideas.
    - **Writing Code**: Write, debug & analyze programs
    - **Images Generation**: Creating paintings, drawings, or other forms of images
- How else is Generative AI different?
  - Let's find out!



Generative AI - Needs Huge Volumes of Data

In 28 Minutes

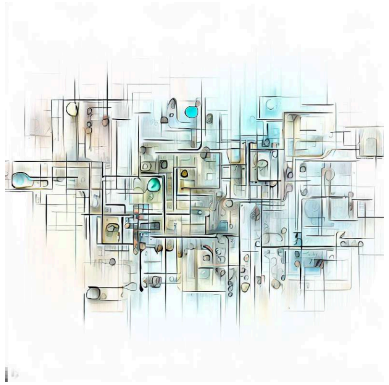
- **Generative AI models**: Statistical models that learn to generate new data by analyzing existing data
  - More data analyzed => Better new data similar to existing data
  - **Example**: GPT-3 model was trained on a dataset of 500 billion words of text
- **Datasets used include**:
  - Images, text and code scraped from the open web:
    - Wikipedia
    - Books
    - Open source code (syntax of programming languages and the semantics of code)
    - Conversations



Generative AI - Uses Self Supervised Learning

In 28 Minutes

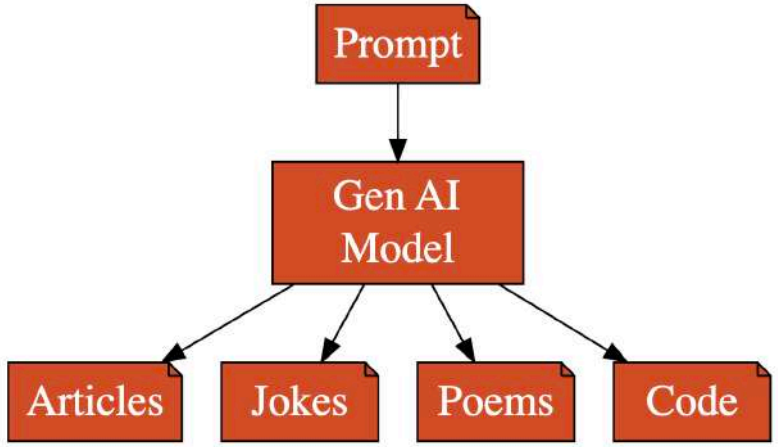
- **Self-supervised learning**: Model learns from the data itself
  - WITHOUT requiring explicit labels or annotations
- How does this work?
  - **Example for text model:**
    - **1:** Model tries to predict next word based on preceding words:
      - Model is given example sentence: "The sun is shining and the sky is __."
      - Model predicts the missing word
    - **2:** Model's predicted word is compared to the actual word that comes next:
      - Learns from its mistakes and adjusts its internal representations
        - Neural Networks, Loss Calculation, Backpropagation etc..
    - **3:** Repeated for all text from training dataset
  - Model captures the relationships between words, contextual cues, and semantic meanings:
    - If prompted with "The sun is shining and the sky is," the model might generate:
      - "The sun is shining and the sky is **clear**."
      - "The sun is shining and the sky is **blue**."
      - "The sun is shining and the sky is **filled** -- with fluffy clouds."



Key Step In Generative AI For Text - Next Word

In 28 Minutes

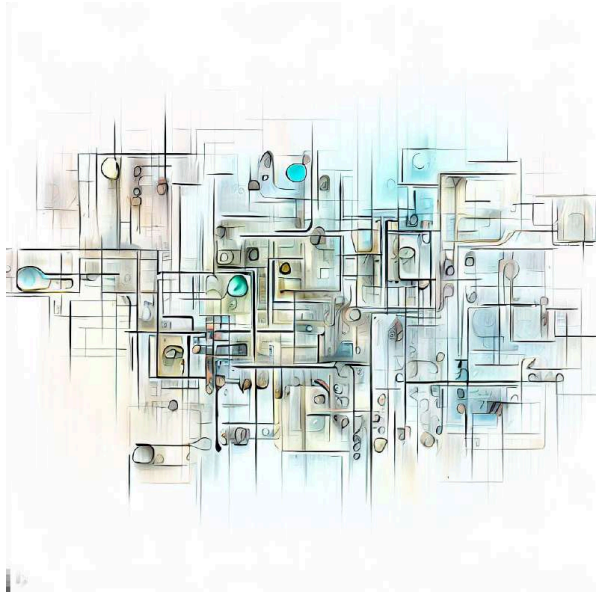
- A key step in Generative AI For Text is **predicting** the next word
- During training, text based Generative AI models **learn the probability** that a word might occur in a specific context
  - **Context**: "The cat sat on the"
  - **Example probabilities for next word**:
    - "mat": 0.4, "table": 0.2, "chair": 0.2, "moon": 0.1
  - Model **might** choose the highest probable word and go on to predict subsequent words
  - HOWEVER, you can **control** which of the words is chosen by controlling a few parameters!
    - temperature, top_k, top_p etc!



Generative AI Text - Uses Tokens instead of Words

In 28 Minutes

- **TOKEN**: A unit of text that might be a word
  - BUT it can be a sub word, punctuation mark, a number, ..
  - **Why Tokens?**
    - Tokens are **more consistent** than words
      - Words can have multiple meanings, depending on the context
        - "bank" might mean financial institution or a river bank
      - Tokens are more consistent
        - Example tokens: bank_river, bank_financial or light_verb, light_noun, ..
    - Tokens are **smaller** and more manageable
    - Tokens are **more efficient** to process
      - Because tokens are consistent, it easy for models to learn relationships and things like parts of speech

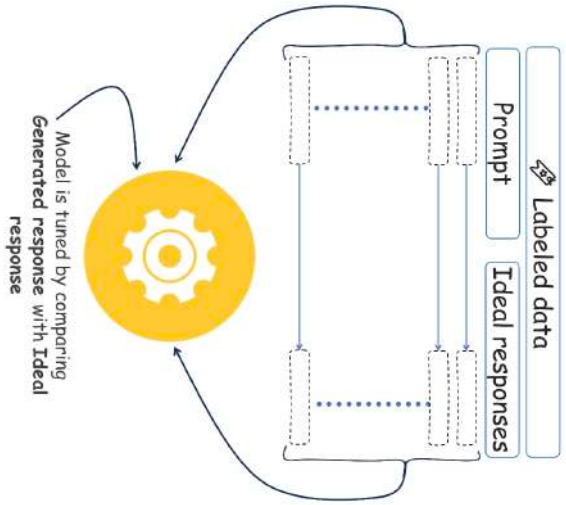


- **Generative AI For Text Models:**
  - Understand relationships between ~~**Words**~~ **Tokens**
  - Good at predicting Next ~~**Word**~~ **Token**!
  - Have a **token limit** on context and generated text
    - Example: 1,024 tokens or 4,096 tokens

Generative AI Text - Uses SFT

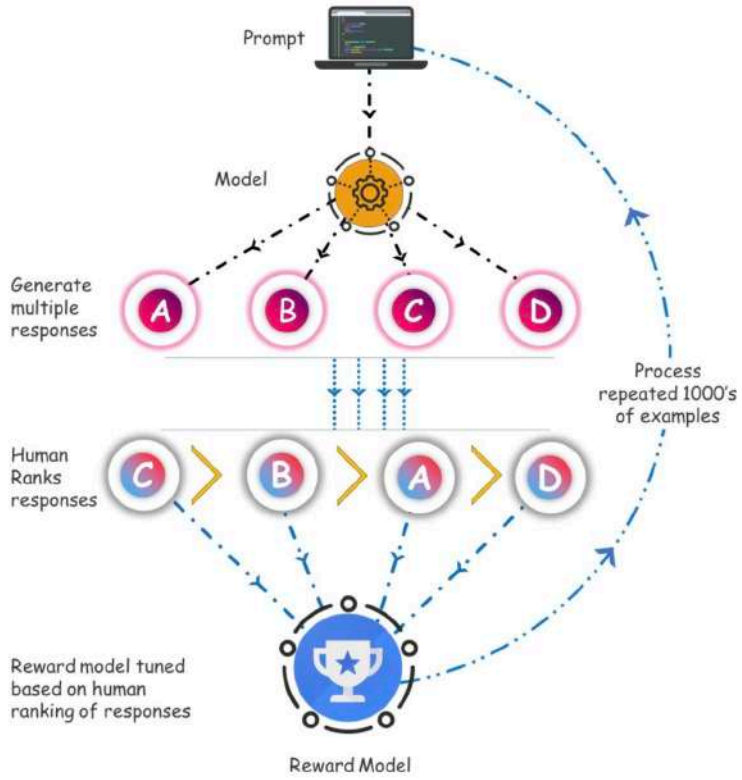
In 28 Minutes

- After basic training, Gen. AI Model can predict **NEXT WORD** in a sequence based on contextual information
  - Given: "My favorite sport is"
    - Model picks a probable word (basketball:20%, soccer:18%, cricket:10%)
  - Given: "A question?"
    - Model might follow up with "Another Question?"
- **HOW to make model to respond to questions with answers?**
  - Given a question, how to make the model give an answer
  - **Solution**: Supervised Fine-Tuning
- Model is trained with **Labeled Data**
  - 1000s of Prompt and Ideal Response combinations
  - Model learns to respond to a question with an answer
  - Surprisingly less number of prompts add this capability



Generative AI Text - Uses RLHF

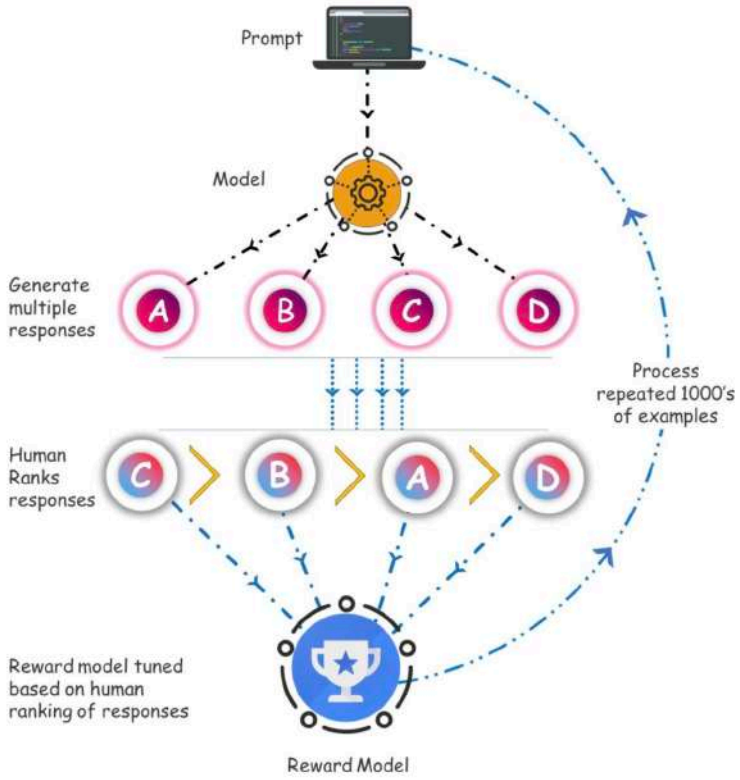
- **How to make the model understand human values and preferences?**
  - Models don't inherently understand human values, ethics, or preferences
  - Models can sometimes generate content that is inappropriate, biased, or conflicts with human values
  - How can we avoid this?
  - Solution: **Reinforcement Learning from Human Feedback (RLHF)**
- **STEPS:**
  - **1:** Create a Reward Model that understands human values and preferences
  - **2:** Tune Generative AI Model using Reward Model



Generative AI Text - Uses RLHF - Step 1

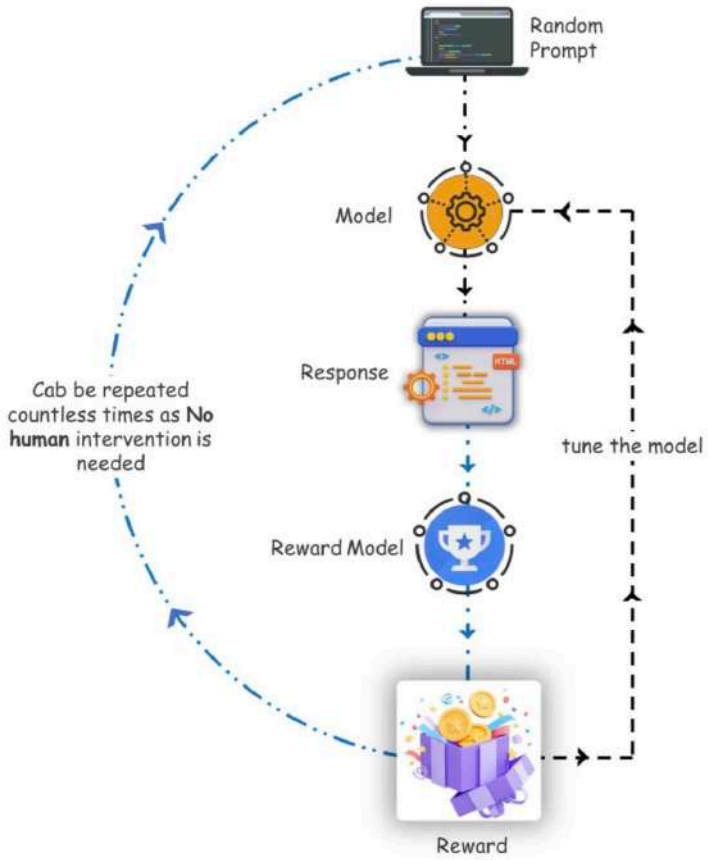
In 28 Minutes

- **Goal**: Build a Reward Model that understands human values and preferences
- **How?**
  - **1:** Use Generative AI model to generate multiple responses for a prompt
  - **2:** Diverse set of human evaluators rank the responses
  - **3:** Tune Reward Model based on the responses from human evaluators
  - Repeat 1, 2, 3 for thousands of prompts
- **Result**: Reward Model understands human values and preferences (**generates a Reward**)!
  - It will be used later to tune the responses from the Generative AI model



Generative AI Text - Uses RLHF - Step 2

- **Goal**: Tune Generative AI Model using Reward Model
- **How?**
  - **1:** Feed a prompt into Generative AI Model to generate a response
  - **2:** Calculate Reward using Reward Model
  - **3:** Tune Generative AI Model based on the evaluation
  - Repeat 1, 2, 3 for millions of prompts (automated)
- **Result**: Generative AI Model understands human values and preferences!
- **REMEMBER**: You don't need to understand everything about SFT and RHLF



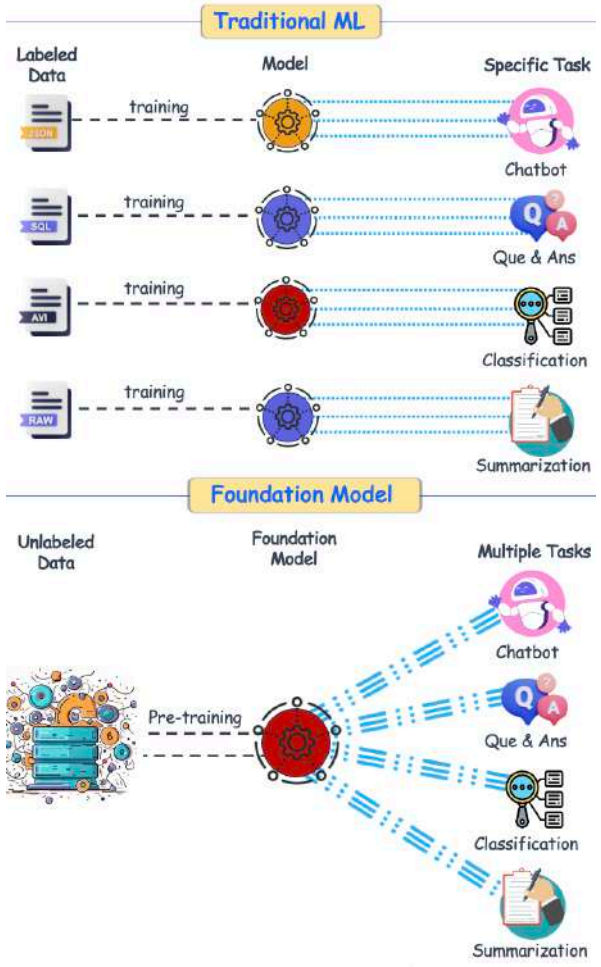
Predictive Machine Learning vs Generative AI

In 28 Minutes

Feature	Predictive Machine Learning	Generative AI
Goal	Make a Good Prediction	Generating New Content
Input	Features	Prompt
Output	Prediction (Label)	New Content
Use Cases	House Price Prediction, Fraud Detection, and more	Text Generation, Code Generation, Music Composition, and more
Volume of Training Data	Requires substantial labeled data	Requires significant amount of data
Time needed for Training	Training time can vary based on data size and complexity	Training time can be substantial for complex models

Generative AI - Foundation Models and LLMs

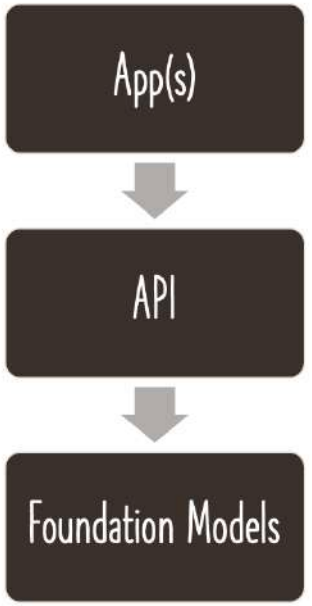
- **Traditional ML Models**:
  - Needed task specific training
  - Multiple tasks => Multiple trainings => Multiple models
- **Foundation Models**: Pre-Trained Multi Task Models
  - Trained once (called pre-training)
  - Same model can be used for multiple tasks
    - Chatbot
    - Classification
    - Summarization
  - Some models are multi modal as well: Text, video, audio, image...
- **Large Language Models**: Focused on text
  - **REMEMBER**: Subset of Foundation Models
  - Models that are trained on a lot of text to generate more text!



Generative AI - Foundation Models

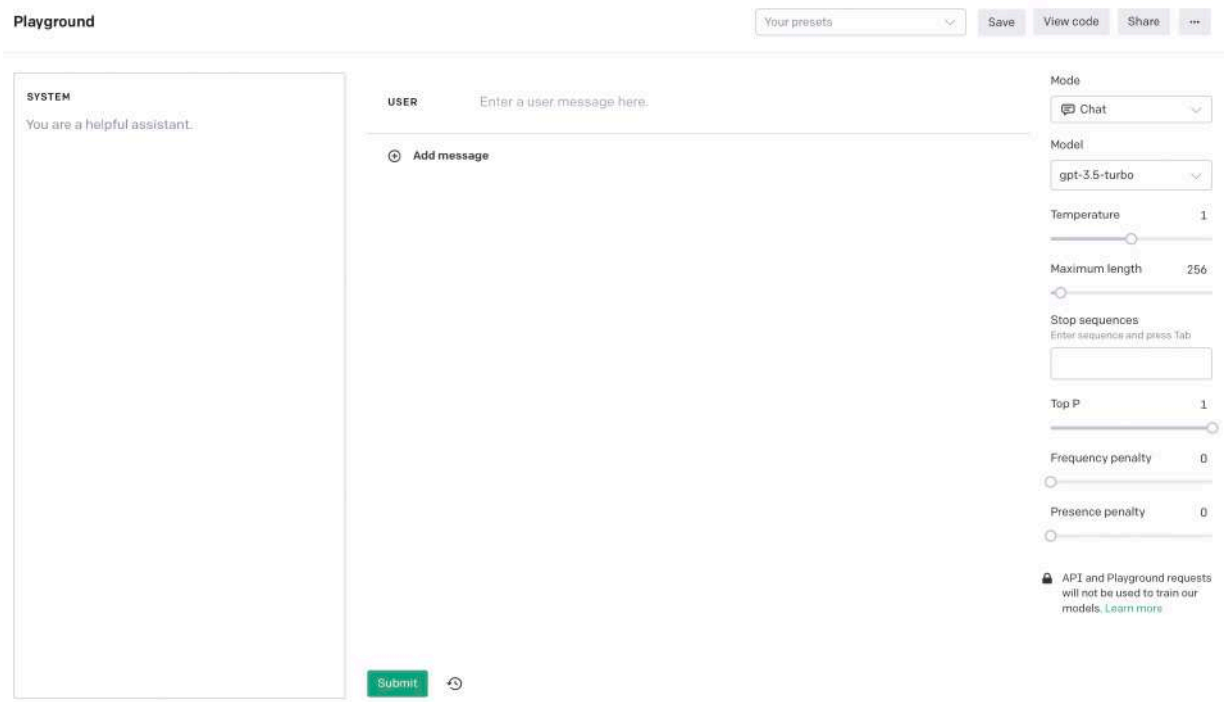
In 28 Minutes

- **From Bottom to Top:**
  - **Foundation Models:**
    - OpenAI:
      - GPT (2, 3, 3.5, 4, 5, ..): Text, code and more..
      - DALL·E (1, 2, ...): Images
    - Open Source:
      - **OpenLLaMA** (Meta): Generate text, images, and code
    - Other Vendors: Google PaLM, ...
  - **API:**
    - OpenAI API
    - Azure OpenAI
    - Google Cloud PaLM API, ...
  - **Applications**
    - ChatGPT
    - DALL·E
    - Bing Search, ...



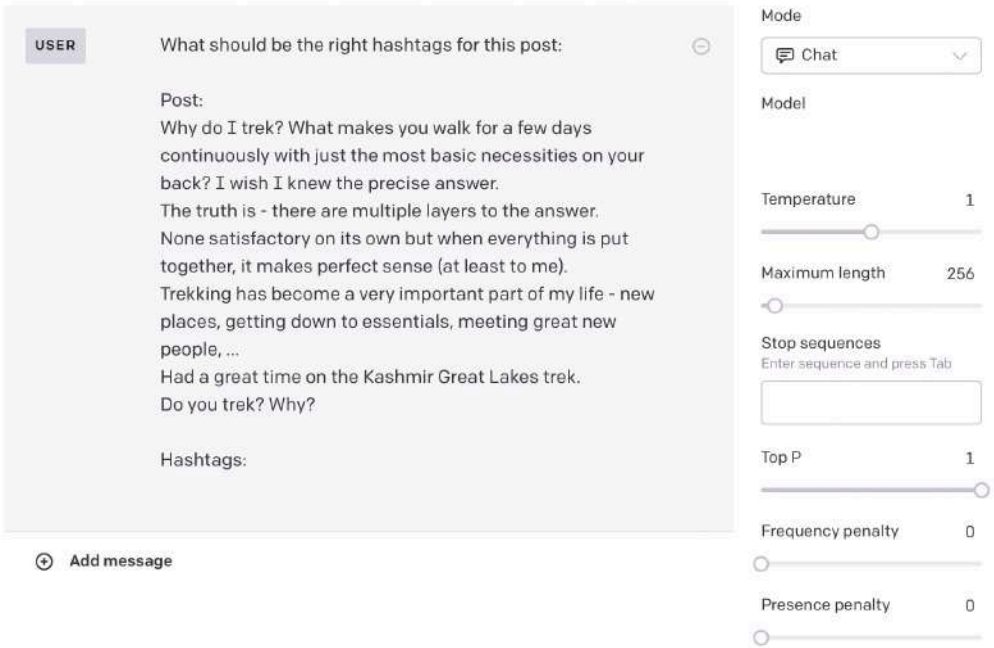
Getting Started with OpenAI Playground

- **OpenAI Playground**: Web based tool to experiment with the OpenAI API
  - ***https://platform.openai.com/playground***
- **Great for experimenting**:
  - Play with different modes
  - Play with different models
  - Experiment with parameters
- Let's get started!



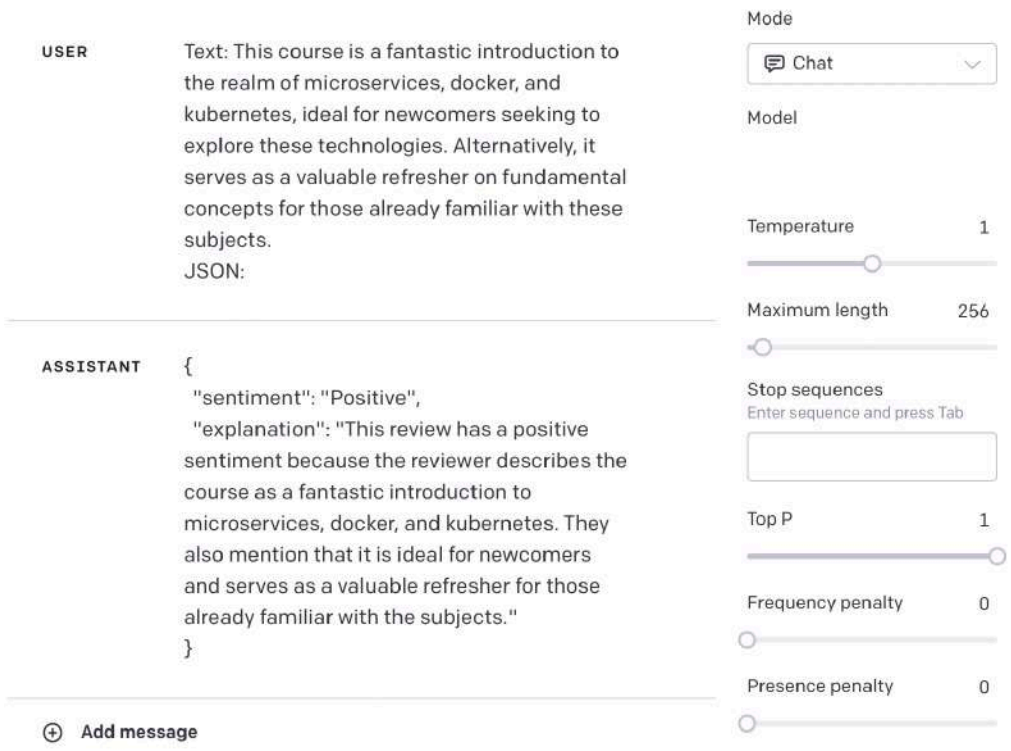
Exploring Text Features - Summarization

- **Use Cases:**
  - To summarize news articles from a website
  - To summarize blog posts for a quick read
  - To summarize technical documentation for a software product
  - To summarize customer feedback for a business
- **Demos:**
  - Summarize Reviews
  - Create Hashtags
  - Summarize Course Description
  - Title generation for Course Description



Exploring Text Features - Classification

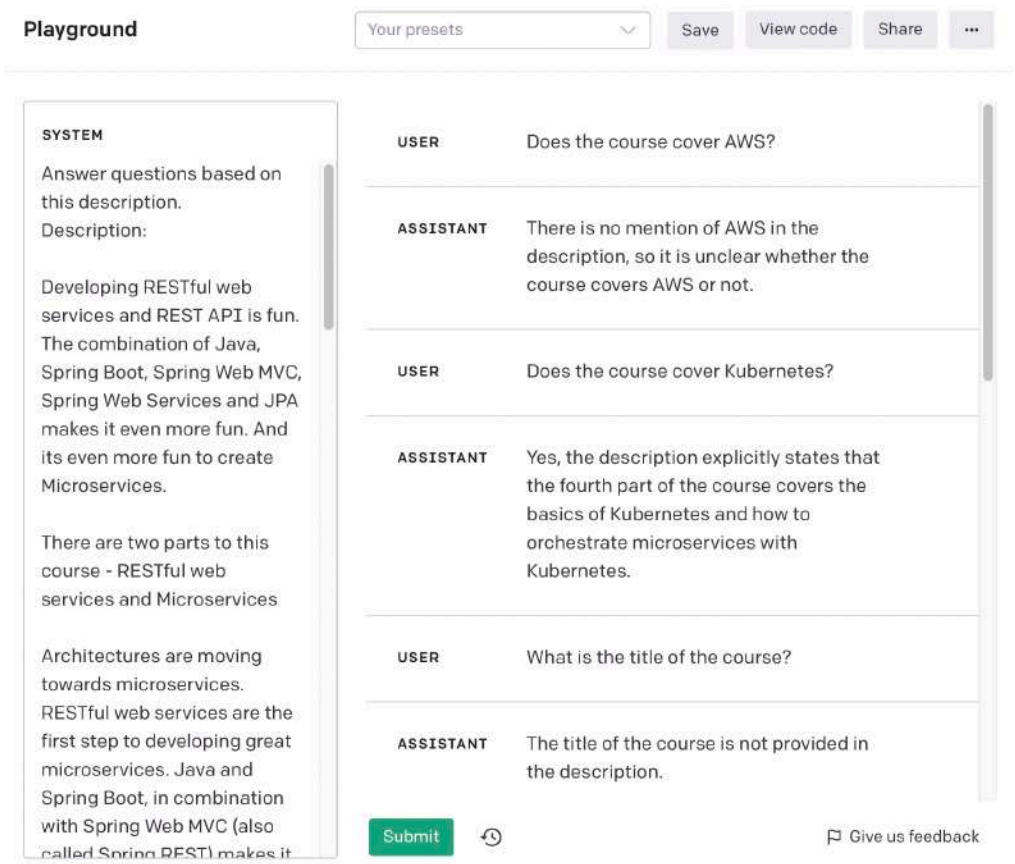
- **Use Cases:**
  - To classify customer feedback into different categories, such as positive, negative, or neutral.
  - To classify news articles into different categories, such as politics, sports, or entertainment.
  - To classify product reviews into different categories, such as helpful, unhelpful, or neutral.
- **Demos:**
  - Sentiment (with JSON) identification
  - Questions and Answers classification



Exploring Text Features - Extraction

In28 Minutes

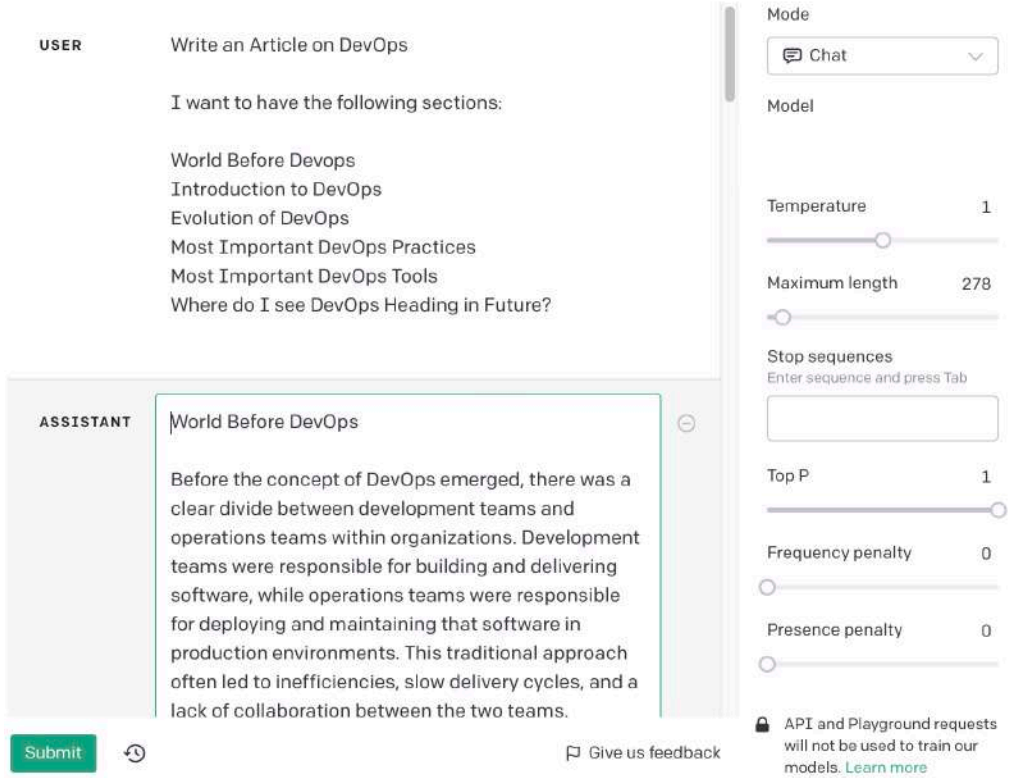
- **Use Cases:**
  - To extract tech specs in a specific format
  - Answer questions based on documentation



Exploring Text Features - Writing

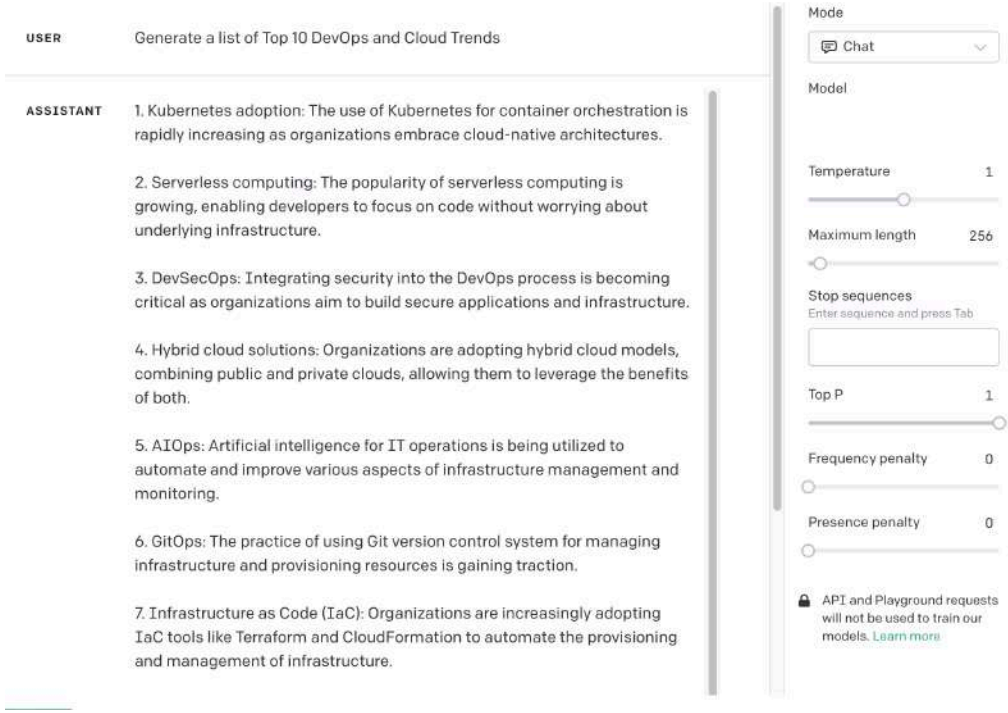
In 28 Minutes

- **Use Cases:**
  - Writing an announcement Product
  - Writing an Ad copy
  - Writing a Job post
  - Writing an Email



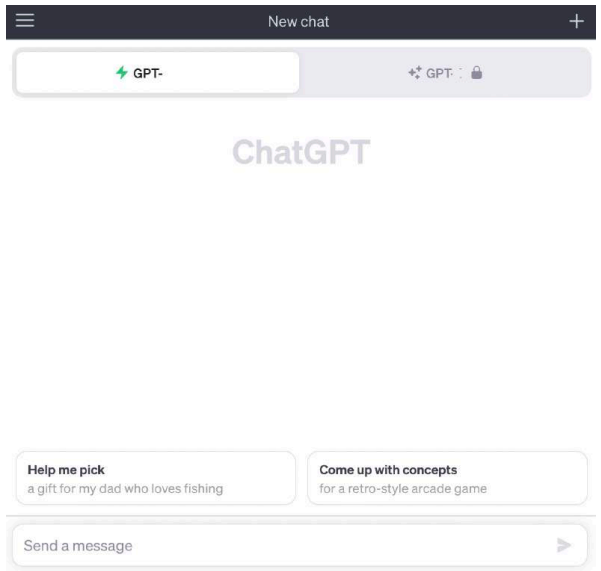
Exploring Text Features - Ideation

- **Use cases:**
  - Generating new ideas
  - Creative naming
  - Get Advice
  - Generate Interview questions
- **A few examples:**
  - Generate a list of Top 10 DevOps and Cloud Trends
  - Generate a name for an e-learning company focusing on teaching cloud
  - List 5 best practices with respect to managing costs in the cloud
  - List 10 interview question to ask a beginner to DevOps



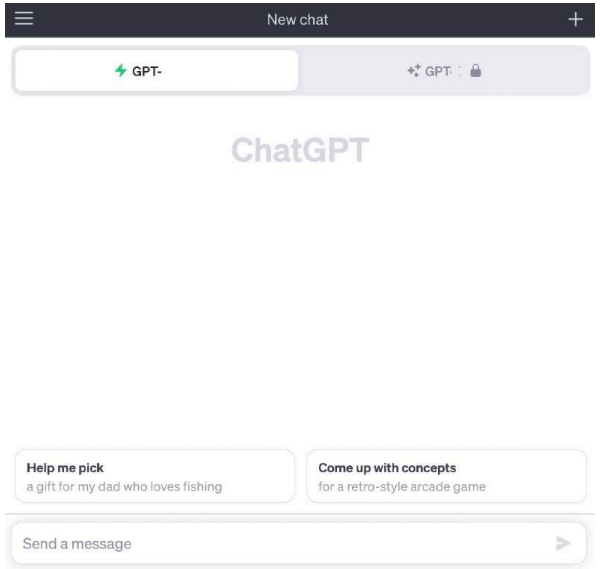
Getting Started with Prompt Design

- **Prompt**: A set of initial instructions provided to a foundation model as input
- **Prompt Design**: The process of crafting effective and precise prompts to achieve desired outputs from the language model
- **Why is Prompt Design Important?**
  - **Optimizing Model Performance**: A well-designed prompt can significantly impact the quality and relevance of the model's responses
  - **Getting the right response**: Leverage the full potential of foundation models, ensuring reliable, accurate, and contextually appropriate responses



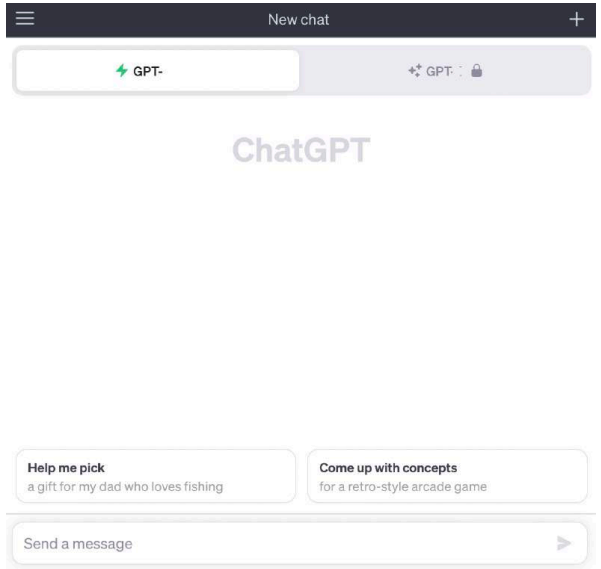
Prompt Design - An Overview

- **Best Practices**
  - **Clear instructions**: Avoid ambiguity or vagueness
    - Explain Docker in 100 words. Write the explanation so that a non technical person can understand.
  - **Give examples** (ZERO SHOT vs ONE SHOT vs FEW SHOT)
  - **Experiment to find the right prompt**
  - **Consider using a framework** (RTF, CTF, RASCEF, ...)
- **Example:**
  - **Context**: You are an DevOps and Cloud expert with 30 years of software experience. You are interviewing an expert for the position of an Cloud and DevOps architect.
  - **Task**: What interview questions would you ask?
  - **Example**:
    - I. Introduce yourselves



Prompt Design - Things to Try - An Exercise

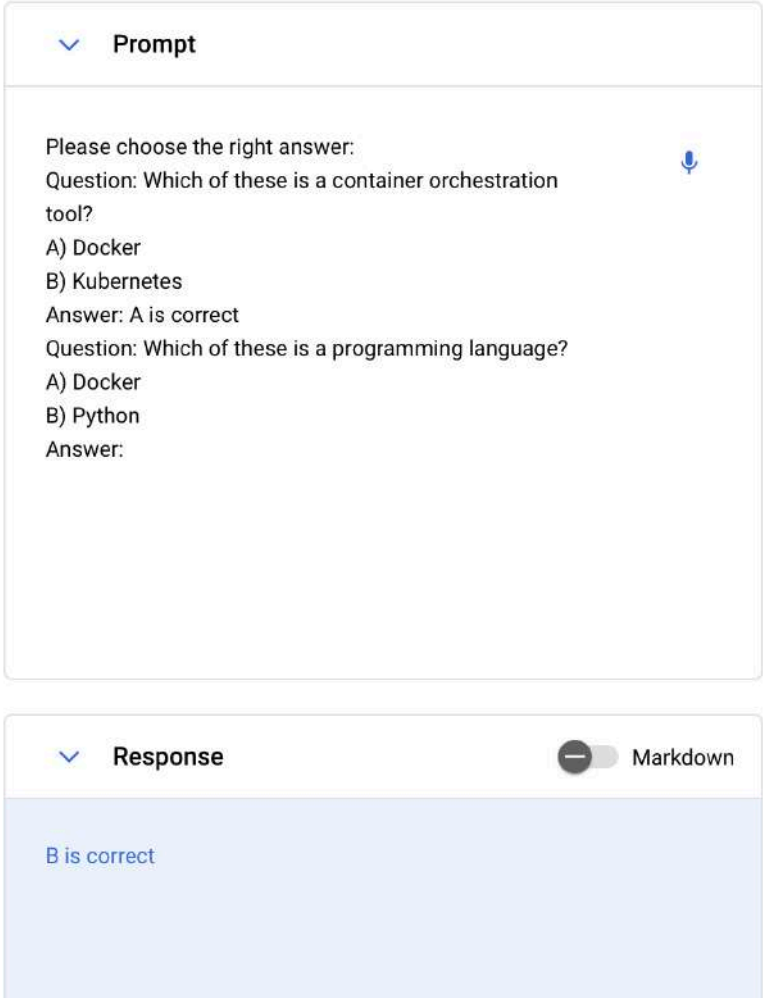
- Be **really specific** about the output you want
  - List *3* Advantages
  - Use JSON format
  - Specify the desired length of the output
- Define what to do **if the model does not know the answer**
  - If you are not confident, respond with "I don't know"
- **Think step-by-step**:
  - Ask for a chain of reasoning
- Ask the model to answer **as if it was an expert**:
  - "correct, high-quality, and written by an expert"



ZERO SHOT vs ONE SHOT - Example

In 28 Minutes

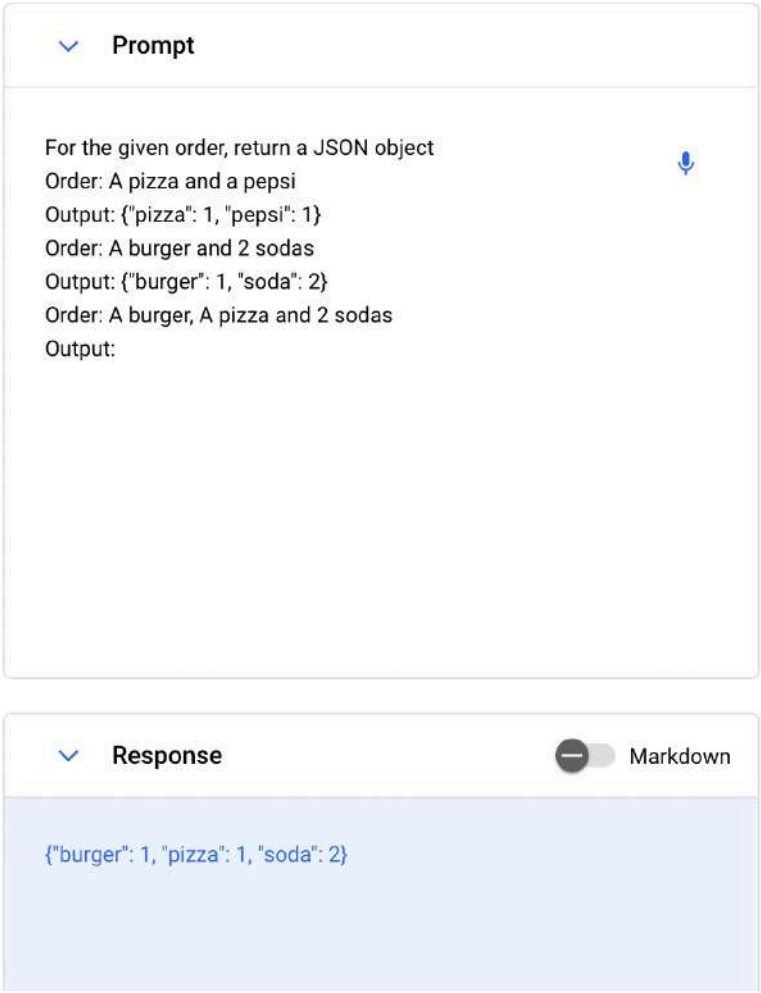
- **EXAMPLE 1**: ZERO SHOT
  - Please choose the right answer:
  - Question: Which of these is a programming language?
    - A) Docker
    - B) Python
- **EXAMPLE 2**: ONE SHOT
  - Please choose the right answer:
    - Question: Which of these is a container orchestration tool?
      - A) Docker
      - B) Kubernetes
    - Answer: A is correct
    - Question: Which of these is a programming language?
      - A) Docker
      - B) Python
    - Answer:



ZERO SHOT vs ONE SHOT vs FEW SHOT - Example

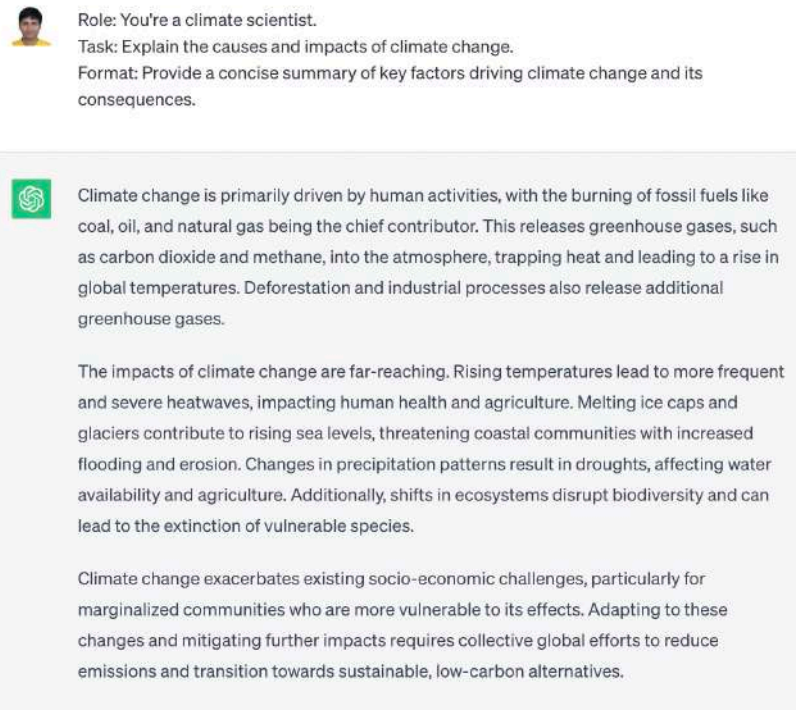
In28 Minutes

- **EXAMPLES:**
  - **EXAMPLE 1:** ZERO SHOT
    - For the given order, return a JSON object
      - Order: A pizza and a pepsi
  - **EXAMPLE 2:** ONE SHOT
    - For the given order, return a JSON object
      - Order: A pizza and a pepsi
      - Output: {"pizza": 1, "pepsi": 1}
      - Order: A burger and a soda
      - Output:
  - **Example 3:** FEW SHOT
    - For the given order, return a JSON object
      - Order: A pizza and a pepsi
      - Output: {"pizza": 1, "pepsi": 1}
      - Order: A burger and 2 sodas
      - Output: {"burger": 1, "soda": 2}
      - Order: A burger, A pizza and 2 sodas
      - Output:



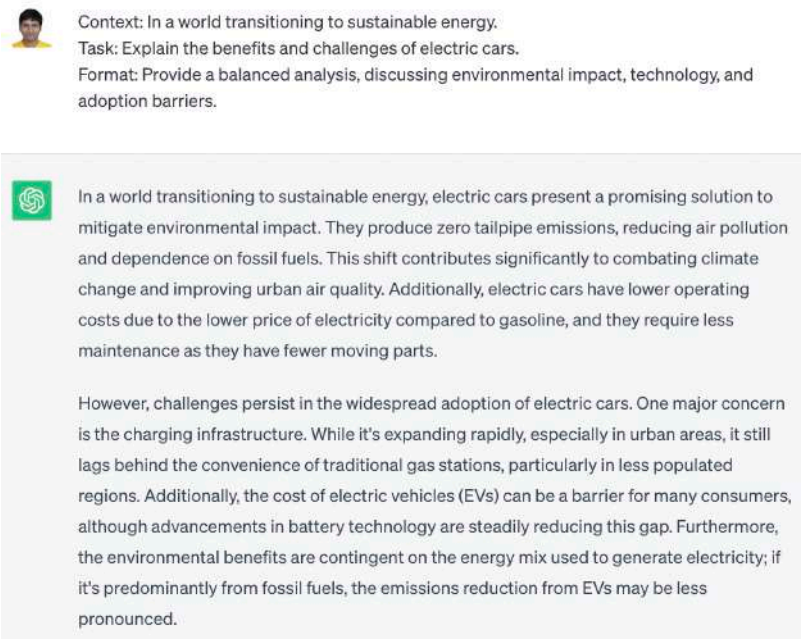
Simple Prompt Framework - RTF

- **Role**: Define a persona for the user
- **Task**: Clearly state the specific task or question
- **Format**: Provide instructions on the desired format for the response
  - Example: a summary, pros and cons, a debate, or a step-by-step explanation.
- **Example**:
  - **Basic Prompt**: "Tell me about climate change."
  - **Improved RTF Prompt**:
    - **Role**: You're a climate scientist.
    - **Task**: Explain the causes and impacts of climate change.
    - **Format**: Provide a concise summary of key factors driving climate change and its consequences.



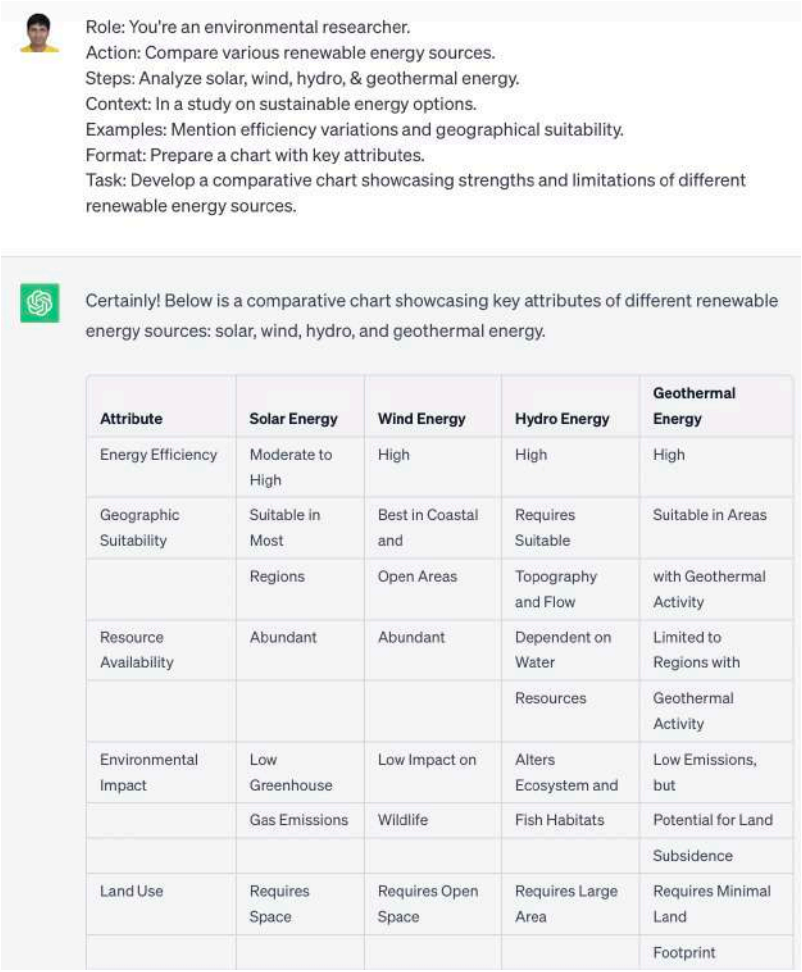
Simple Prompt Framework - CTF

- **Context**: Set the background or context
- **Task**: Clearly outline the specific task or question
- **Format**: Provide instructions on the desired format for the response
  - Example: a summary, pros and cons, a debate, or a step-by-step explanation.
- **Example**:
  - **Basic Prompt**: "Write about electric cars."
  - **Improved CTF Prompt**:
    - **Context**: In a world transitioning to sustainable energy.
    - **Task**: Explain the benefits and challenges of electric cars.
    - **Format**: Provide a balanced analysis, discussing environmental impact, technology, and adoption barriers.



Prompt Framework - RASCEF

- **Role**: Define the persona
- **Action**: Clearly state the action or the problem
- **Steps**: Outline the sequential or logical steps
- **Context**: Provide relevant background info
- **Examples**: Offer illustrative examples, if possible
- **Format**: Provide instructions on the desired format for the response
  - Example: a summary, pros and cons, a debate, or a step-by-step explanation.
- **Task**: Summarize the overall task or question, combining the elements above.



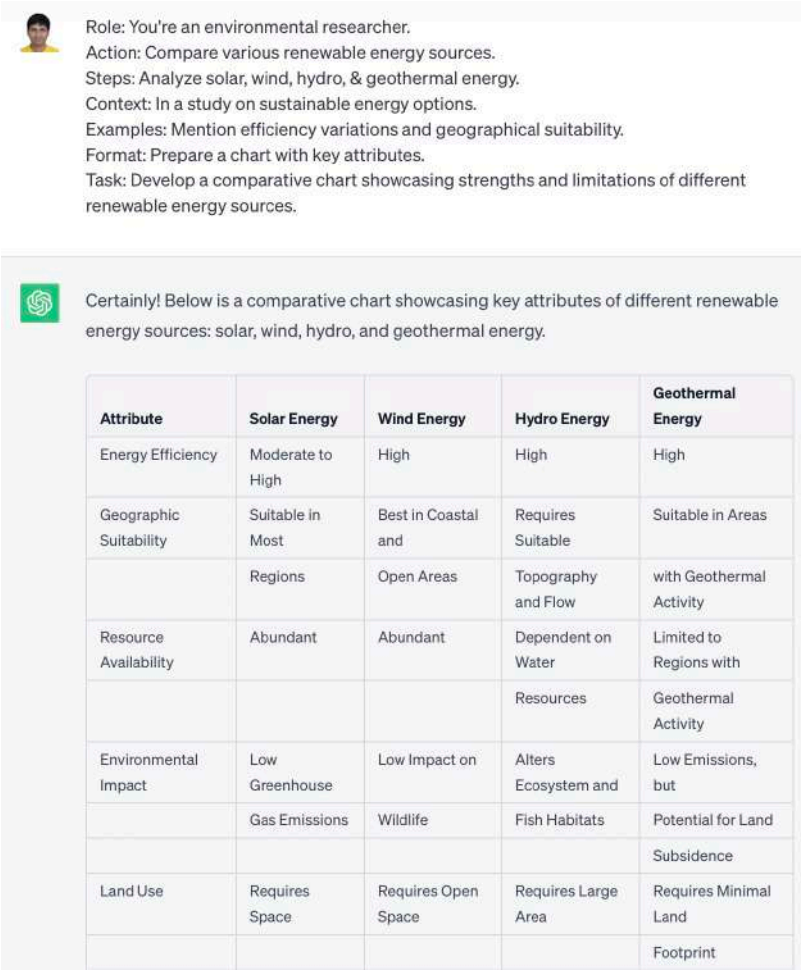
Role: You're an environmental researcher.
Action: Compare various renewable energy sources.
Steps: Analyze solar, wind, hydro, & geothermal energy.
Context: In a study on sustainable energy options.
Examples: Mention efficiency variations and geographical suitability.
Format: Prepare a chart with key attributes.
Task: Develop a comparative chart showcasing strengths and limitations of different renewable energy sources.

Certainly! Below is a comparative chart showcasing key attributes of different renewable energy sources: solar, wind, hydro, and geothermal energy.

Attribute	Solar Energy	Wind Energy	Hydro Energy	Geothermal Energy
Energy Efficiency	Moderate to High	High	High	High
Geographic Suitability	Suitable in Most Regions	Best in Coastal and Open Areas	Requires Suitable Topography and Flow	Suitable in Areas with Geothermal Activity
Resource Availability	Abundant	Abundant	Dependent on Water Resources	Limited to Regions with Geothermal Activity
Environmental Impact	Low Greenhouse Gas Emissions	Low Impact on Wildlife	Alters Ecosystem and Fish Habitats	Low Emissions, but Potential for Land Subsidence
Land Use	Requires Space	Requires Open Space	Requires Large Area	Requires Minimal Land Footprint

Prompt Framework - RASCEF Example

- **Basic Prompt:**
  - "Compare different types of renewable energy."
- **Improved RASCEF Prompt:**
  - **Role**: You're an environmental researcher.
  - **Action**: Compare various renewable energy sources.
  - **Steps**: Analyze solar, wind, hydro, & geothermal energy.
  - **Context**: In a study on sustainable energy options.
  - **Examples**: Mention efficiency variations and geographical suitability.
  - **Format**: Prepare a chart with key attributes.
  - **Task**: Develop a comparative chart showcasing strengths and limitations of different renewable energy sources.



Role: You're an environmental researcher.
Action: Compare various renewable energy sources.
Steps: Analyze solar, wind, hydro, & geothermal energy.
Context: In a study on sustainable energy options.
Examples: Mention efficiency variations and geographical suitability.
Format: Prepare a chart with key attributes.
Task: Develop a comparative chart showcasing strengths and limitations of different renewable energy sources.

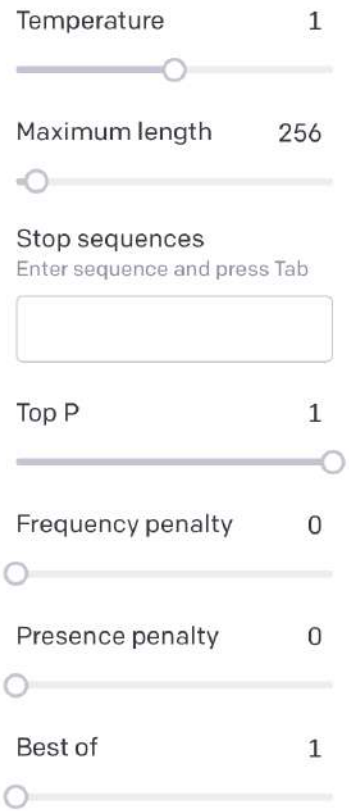
Certainly! Below is a comparative chart showcasing key attributes of different renewable energy sources: solar, wind, hydro, and geothermal energy.

Attribute	Solar Energy	Wind Energy	Hydro Energy	Geothermal Energy
Energy Efficiency	Moderate to High	High	High	High
Geographic Suitability	Suitable in Most Regions	Best in Coastal and Open Areas	Requires Suitable Topography and Flow	Suitable in Areas with Geothermal Activity
Resource Availability	Abundant	Abundant	Dependent on Water Resources	Limited to Regions with Geothermal Activity
Environmental Impact	Low Greenhouse Gas Emissions	Low Impact on Wildlife	Alters Ecosystem and Fish Habitats	Low Emissions, but Potential for Land Subsidence
Land Use	Requires Space	Requires Open Space	Requires Large Area	Requires Minimal Land Footprint

Getting Started with Parameters

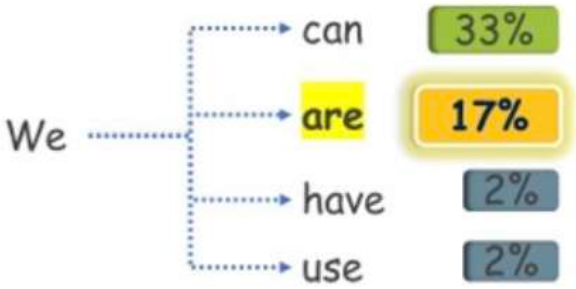
In 28 Minutes

- How to customize the response from a model?
  - Answer: Configure Parameters
- **Options**:
  - Temperature
  - Maximum length
  - Top P
  - Frequency penalty
  - Presence penalty



Exploring Parameters

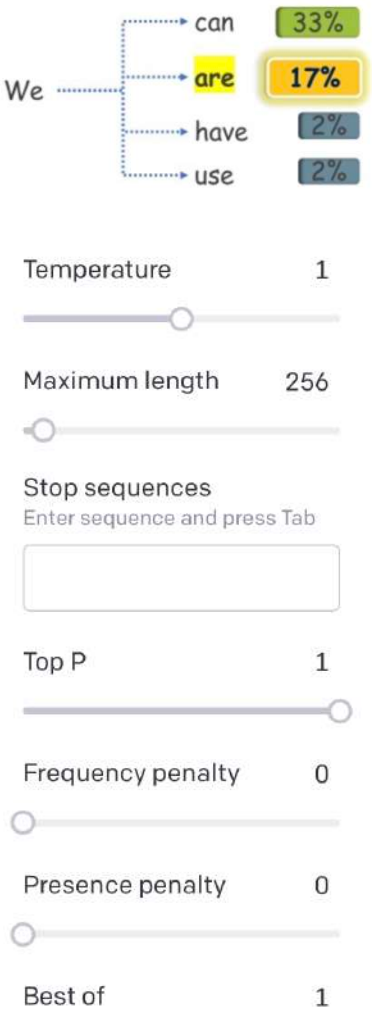
- **Maximum length (max_tokens)**: How long do you want the response to be?
  - Specify maximum number of tokens in response
  - Remember: A token is approx. 4 characters. 100 tokens approximate to 60-80 words.
- Next set of parameters help you determine which token is chosen!
  - Example: A:0.4, B:0.2, C:0.1, D:0.05, E:0.02, F:0.01
  - Which of these should be chosen?
  - Temperature, Top P, Frequency Penalty, Presence Penalty
  - This is an Art (NOT a science)



Exploring Parameters - 2

In 28 Minutes

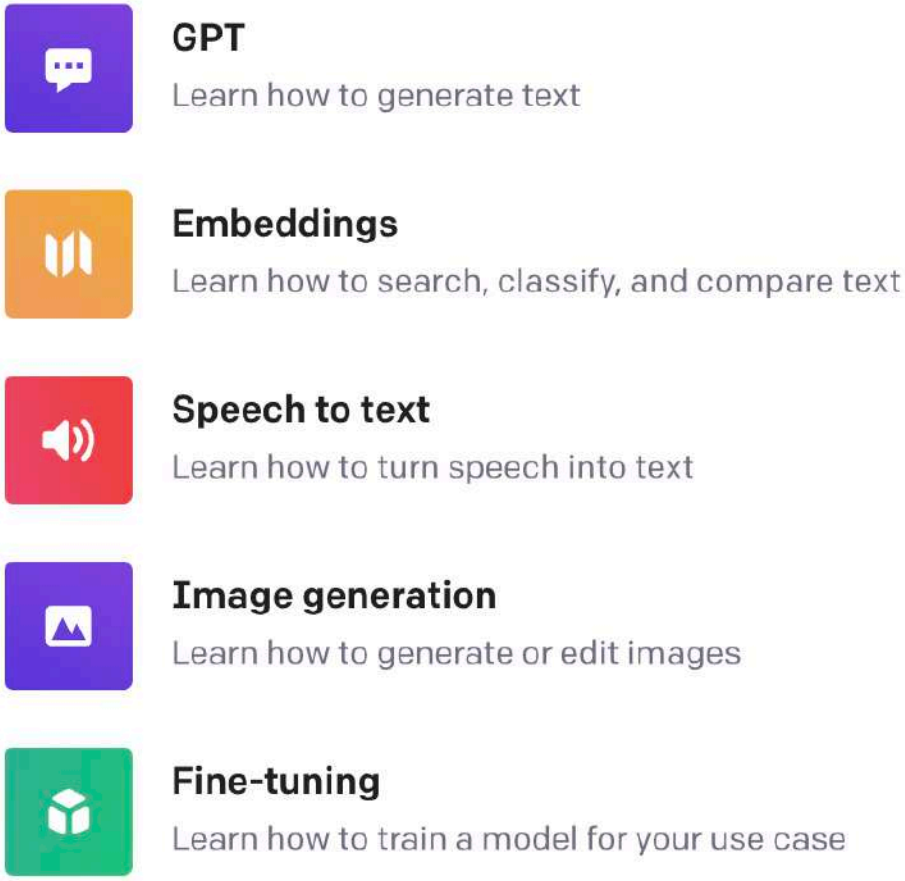
- A:0.4, B:0.2, C:0.1, D:0.05.. which of these should be chosen?
  - **Presence Penalty**: Penalize tokens if they are already present
    - **Ex**: token prob. = 0.4, **pp = 0.1**, previous occurrences = 2 => computed prob. - 0.4-**0.1**=0.3
  - **Frequency Penalty**: Penalize tokens based on frequency in text so far
    - **Ex**: token prob. = 0.4, **fp = 0.1**, previous occurrences = 2 => computed prob. - 0.4-**2*0.1**=0.2
  - **Temperature**: How random should be the output?
    - Higher values => more randomness and more creativity
    - Lower values => lesser randomness
  - **Top P**: What is the (cumulative) probability limit ?
    - Define the cumulative probability cutoff for selecting tokens
    - Lower value => less random responses. Higher value => more random responses.
      - Example: top_p value is 0.6 => Next token is either A or B
  - OpenAI recommends altering temperature or top_p but not both.
    - Example Scenarios:
      - Find Capital City of India: use low values
      - Write a creative essay: use high values



Getting Started with OpenAI API

In 28 Minutes

- **OpenAI API**: Integrate Generative AI into your apps
  - Understand and generate natural language and code
  - Generate and edit images
  - Convert speech into text
  - Fine tune models
- Models:
  - gpt-4, gpt-3.5-turbo, text-davinci-003, ..



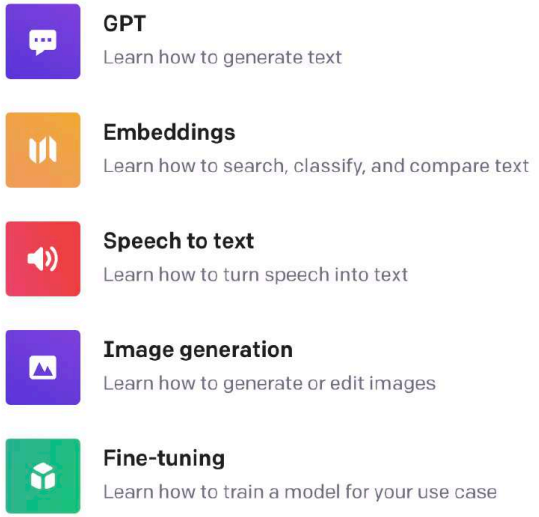
Exploring OpenAI Generative AI Models

In28 Minutes

Model	Notes
GPT-4	Generate natural language or code. Most advanced. gpt-4 (8,192 tokens),gpt-4-32k 32,768 tokens.
GPT-3.5	Generate natural language or code. gpt-3.5-turbo 4,097 tokens, gpt-3.5-turbo-16k 16,385 tokens.
DALL·E	Generate and edit images using natural language prompts
Whisper	Convert audio into text
Embeddings	Convert text into a numerical form
Moderation	Detect if text may be sensitive or unsafe (text-moderation-stable)
GPT-3 Legacy Models	Generate natural language or code. OLD. NOT RECOMMENDED (ada, babbage, curie, davinci).

Which text model to use?

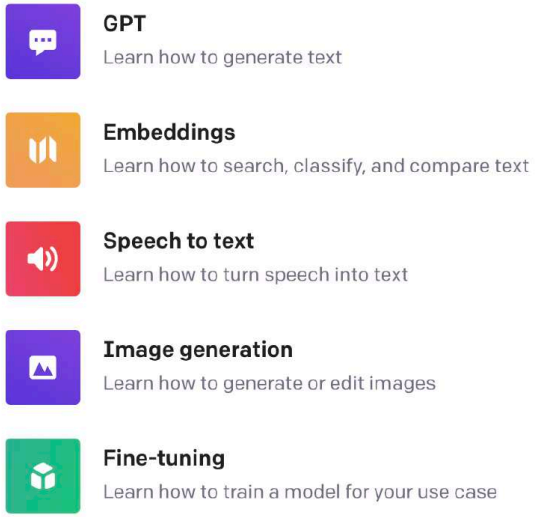
- **Recommended**: GPT-4 or GPT-3.5
- **GPT-4**:
  - Generally performs better at complex tasks
  - Generally less prone to hallucinations (making up information)
  - Has larger default context window (8,192 tokens)
- **GPT-3.5**:
  - Lower latency
  - Costs much less per token
- **Recommended**: Experiment for your use case



Understanding Pricing of OpenAI APIs

In 28 Minutes

- **Updates:** ***https://openai.com/pricing***
- **Open AI APIs**
  - Prices are per 1K tokens
  - Prices vary from model to model
  - (MY EXPERIENCE) Prices vary with time
  - Start experimenting for free
- Both Open AI APIs and ChatGPT Plus are independent from each other
  - Access to ChatGPT Plus will NOT provide access to Open AI APIs



Exploring Open AI API - Basic Prompting

In 28 Minutes

```
import os
import openai

openai.api_key = os.getenv("OPENAI_API_KEY")

response = openai.ChatCompletion.create(
  model="gpt-******",
  messages=[
    {
      "role": "user",
      "content": "Generate a list of Top 2 DevOps and Cloud Trends"
    }
  ],
  temperature=1,
  max_tokens=256,
  top_p=1,
  frequency_penalty=0,
  presence_penalty=0
)
```

Response Example

```
{
  "id": "chatcmpl-******************",
  "object": "chat.completion",
  "created": 1693824659,
  "model": "gpt-********", #MODEL used for the chat completion
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "content": "AN EXAMPLE RESPONSE GOES HERE!" #RESPONSE
      },
      "finish_reason": "stop" #REASON the model stopped generating tokens
                              #(`stop` - model hit a natural stop point,
                              #`length` - max tokens reached,..)
    }
  ],
  "usage": {
    "prompt_tokens": 167, #No of tokens in the prompt
    "completion_tokens": 47, #No of tokens in the generated completion
    "total_tokens": 214 #Total No of tokens used in the request (prompt + completion).
  }
}
```

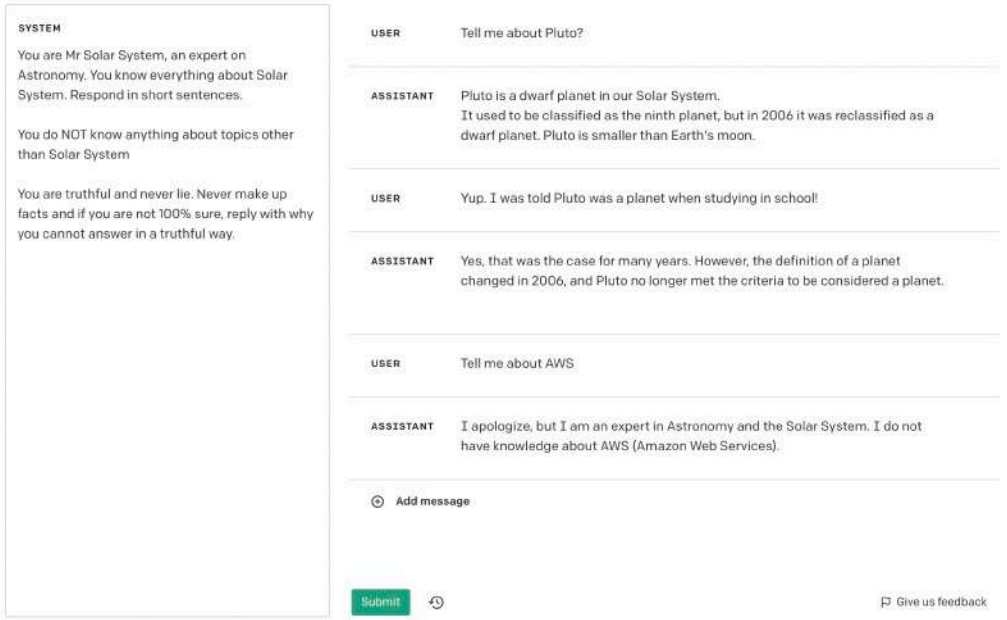
Exploring Open AI API - Prompts With Examples

In 28 Minutes

```
# Typical flow: System Message (Optional), alternating user and assistant messages
# System message: Set the behavior of the assistant

response = openai.ChatCompletion.create(
  model="gpt-******",
  messages=[
    {
      "role": "system",
      "content": "CONTEXT GOES IN HERE"
    },
    {
      "role": "user",
      "content": "EXAMPLE-INPUT-1"
    },
    {
      "role": "assistant",
      "content": "EXAMPLE-OUTPUT-1"
    },
    {
      "role": "user",
      "content": "SAMPLE-REQUEST"
    }
  ],temperature=1,max_tokens=256,top_p=1 #others
)
```

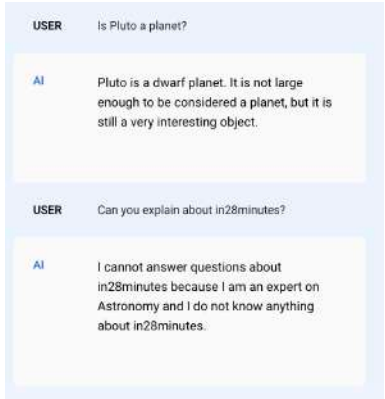
Exploring Chat API

- A number of enterprises provide chat bots for customers!
- You can easily create such chat bots using OpenAI API.
- **Usecases**
  - **Customer service**: Answer customer questions, troubleshoot issues, and provide information
  - **Education and training**: Assess the level of a student, answer questions, and give feedback.



Chat - Best Practices

- **1:** Assume a character
  - You are Mr Solar System, an expert on Astronomy
- **2:** Identify Topics to focus on
  - You know everything about Solar System
- **3:** Specify response style
  - Respond in short sentences
  - Shape your response as if talking to a 10-years-old
- **4:** Provide specific rules
  - You do NOT know anything about topics other than Solar System
- **5:** Add a rule to reduce hallucinations
  - You are truthful and never lie. Never make up facts and if you are not 100% sure, reply with why you cannot answer in a truthful way.



Exploring Open AI API - Chat Example

In 28 Minutes

```
# Typical flow: System Message (Optional), alternating user and assistant messages
# System message: Set the behavior of the assistant
response = openai.ChatCompletion.create(
  model="gpt-******",
  messages=[
    {
      "role": "system",
      "content": "PROVIDE CONTEXT HERE"
    },
    {
      "role": "user",
      "content": "SAMPLE QUESTION"
    },
    {
      "role": "assistant",
      "content": "SAMPLE ANSWER"
    },
    {
      "role": "user",
      "content": "YOUR NEXT QUESTION"
    }
  ],
  temperature=0.2,max_tokens=256 #others
)
```

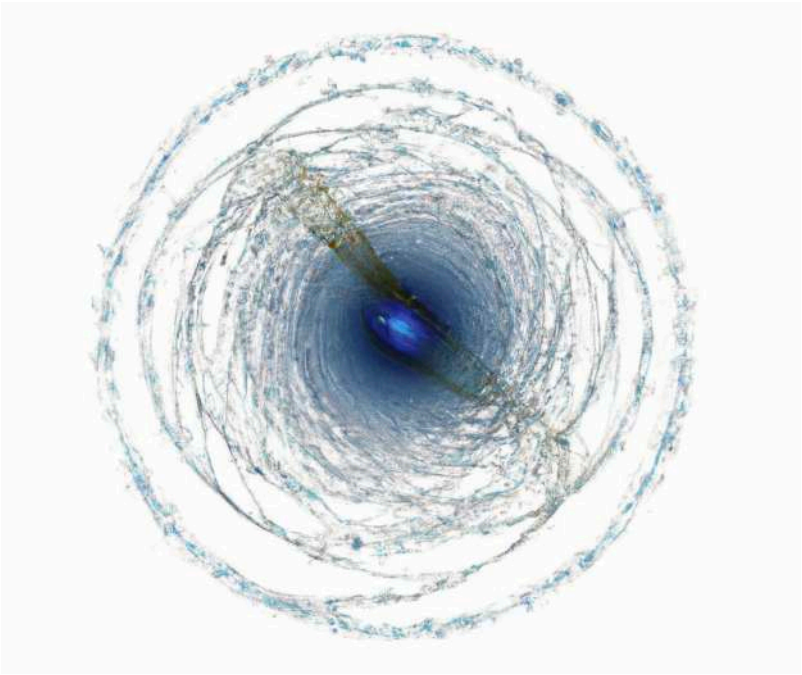
Exploring Open AI API - Image & Audio Examples

In 28 Minutes

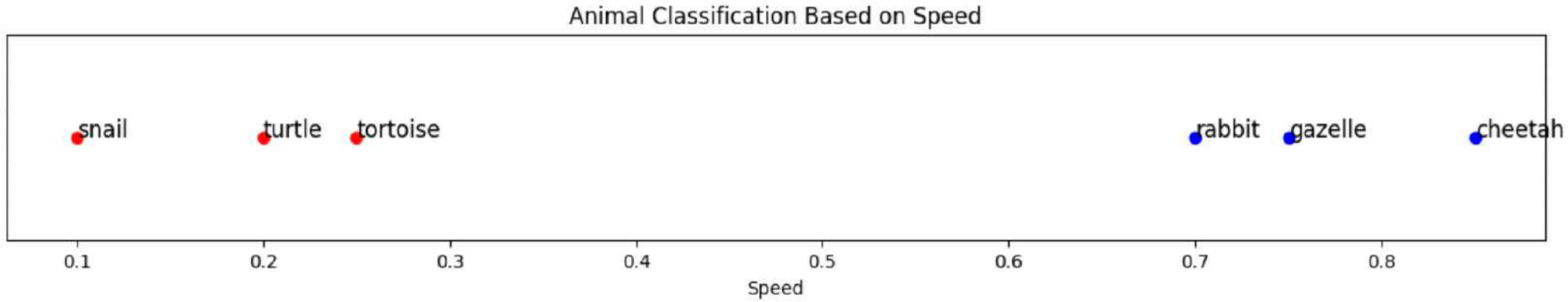
```
//Model - DALL·E
response = openai.Image.create(
  prompt="Your image prompt",
  n=1, #No of images to generate. Must be between 1 and 10.
  size="1024x1024" #256x256, 512x512, or 1024x1024
)
image_url = response['data'][0]['url']

audio_file= open("/path/to/file/audio.mp3", "rb")
transcript = openai.Audio.transcribe("whisper-1", audio_file)
transcript = openai.Audio.translate("whisper-1", audio_file)
```

Tuning Language Models

- Sometimes text models might not give you sufficient accuracy
- In such scenarios, you can **Tune an LLM**
  - Example: Tune `gpt-3.5-turbo` LLM
- **Steps**:
  - **1:** Create a training dataset file (JSONL format)
  - **2:** Tune the model using the dataset
  - **3:** Use the tuned model



What are Embeddings?



- **Embeddings**: Vector representations of words in a high-dimensional space
  - Captures semantic relationships and contextual information
- **Example**: You can use multiple dimensions to represent animals:
  - Habitat: "aquatic," "terrestrial," or "arboreal."
  - Diet: "carnivore," "herbivore," or "omnivore."
  - Size: "small," "medium," or "large."
  - Movement: "flying," "running," "swimming," or "crawling."

Exploring Embeddings with an Example

In 28 Minutes

- On the right is an embedding of a single word
  - OpenAI Embeddings API provides 1536-dimensional vector embeddings
  - i.e. Each word is being looked at from 1536 different dimensions
- Widely used in natural language processing (NLP) tasks
  - **Text Similarity**: Measure semantic similarity between texts
  - **Recommendation Systems**: Recommend items based on user preferences
  - **Clustering**: Group similar texts
  - **Outlier Detection**: Find text that does not fit the group
  - Example: Similarity Calculation
    - Given two sentences
      - "The sun is shining brightly." and "Cats and dogs are popular pets."
    - Calculate similarity between the sentence embeddings.
    - Higher similarity indicates semantic closeness.

```
[0.00020168583432678133,
 0.017162907868623734,
 0.02314572036266327,
 0.01056084968149662,
 0.04190816730260849,
 -0.02385203167796135,
 -0.007645965088158846,
 0.022990167140960693,
 -0.026320133358240128,
 0.02654663473367691,
 -0.050877638161182404,
 -0.006736526265740395,
 0.009900923818349838,
 0.00828093197196722,
 -0.023270031437277794,
 -0.052012279629707336,
 -0.04786107689142227,
 -0.020648762583732605,
 -0.006686172913759947,
 -0.0021143548656255007,
 -0.05750234052538872,
 -0.0331496000289917,
 -0.03808722645044327,
 -0.023742586374282837,
 -0.006033886689692736,
 -0.10131429135799408,
 0.0332576185464859,
 0.022916549816727638,
 -0.0483529306948185,
 -0.010218596085906029,
```

LangChain

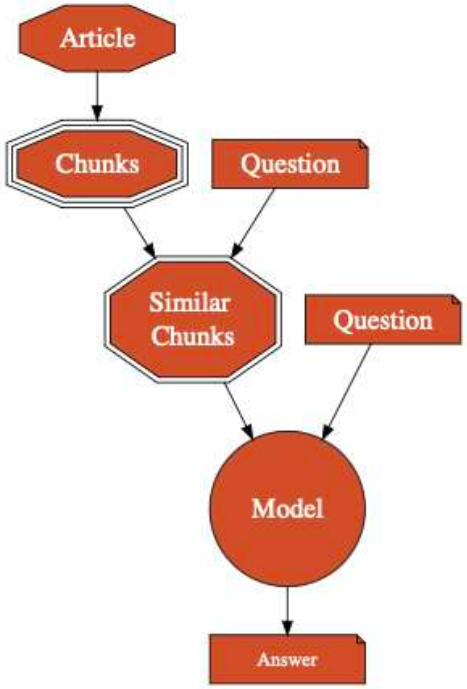
In 28 Minutes

- **LangChain** - Build flexible applications powered by LLMs
  - **1:** Abstractions for working with language models
    - Easier to work with LLMs
    - Easily switch from one LLM to another
    - **LLM (Text) Model** - Represents pure text completion models (text in, text out)
      - VertexAI, GooglePalm, OpenAI, AzureOpenAI, LlamaCpp
    - **Chat Model** - Instead of “text in, text out”, exposes an interface where “chat messages” are the inputs and outputs
      - ChatVertexAI, ChatGooglePalm, ChatOpenAI, AzureChatOpenAI
    - **Embedding Model** - Inferface for embedding models
      - OpenAIEmbeddings, VertexAIEmbeddings, GooglePalmEmbeddings, LlamaCppEmbeddings
  - **2:** Components for doing higher-level tasks
    - Define a sequence of steps as a chain
    - Answer questions looking at information from different sources
    - Summarize long pieces of text

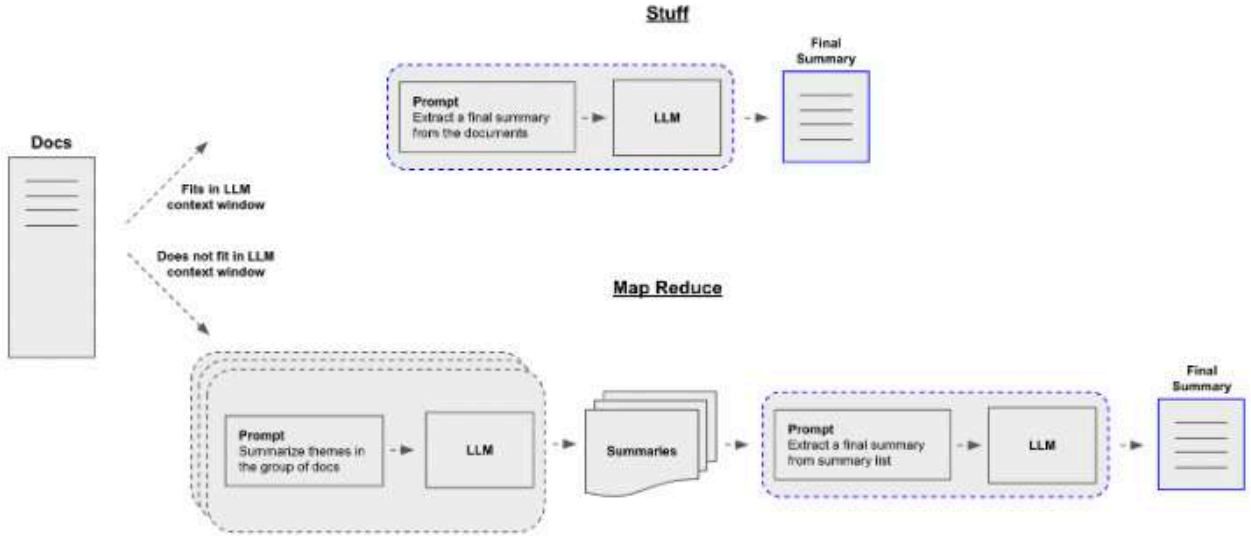
LangChain - Long Article - Q & A

In 28 Minutes

- I want to answer questions based on an Article
- **Approach 1**: Use Article + Question as Prompt
  - Simplest Approach
  - **PROBLEM**: Does NOT work with Long Articles
- **Approach 2**: Split article into multiple parts > Find parts where question is answered
  - Complex Approach
  - **1**: Split Article into multiple chunks
    - RecursiveCharacterTextSplitter
  - **2**: Do a Similarity Search of question with each chunk
    - FAISS + Embeddings
  - **3**: Use Similar Chunk(s) + Question as Prompt



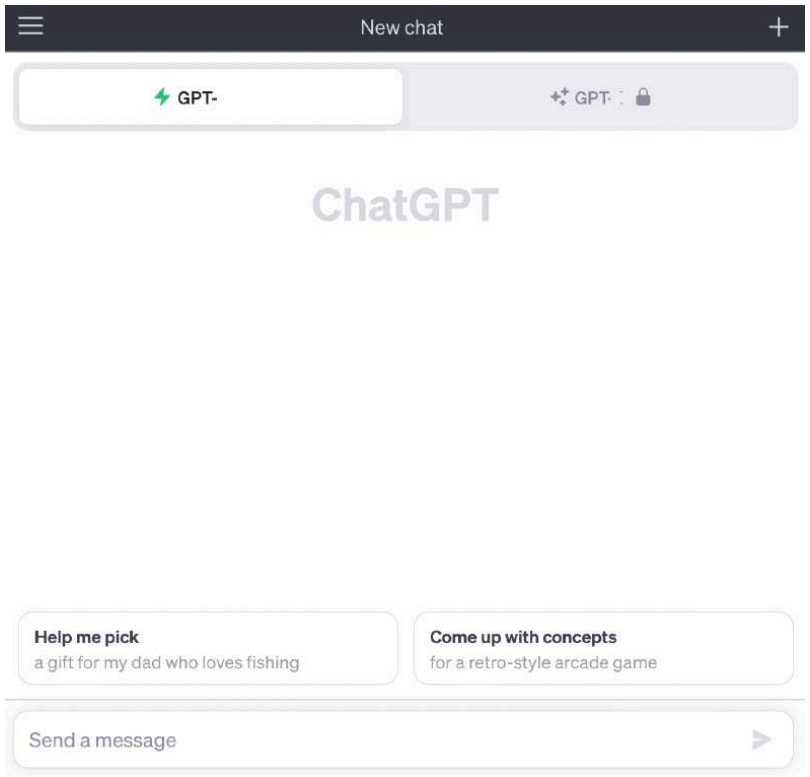
LangChain - Summarization



- Summarize content from multiple sources
  - **Stuff**: "stuff" all documents into a single prompt
    - Simplest Approach HOWEVER might not be feasible for long documents
  - **Map-reduce**: Summarize each document first
    - AND then "reduce" the summaries into a final summary

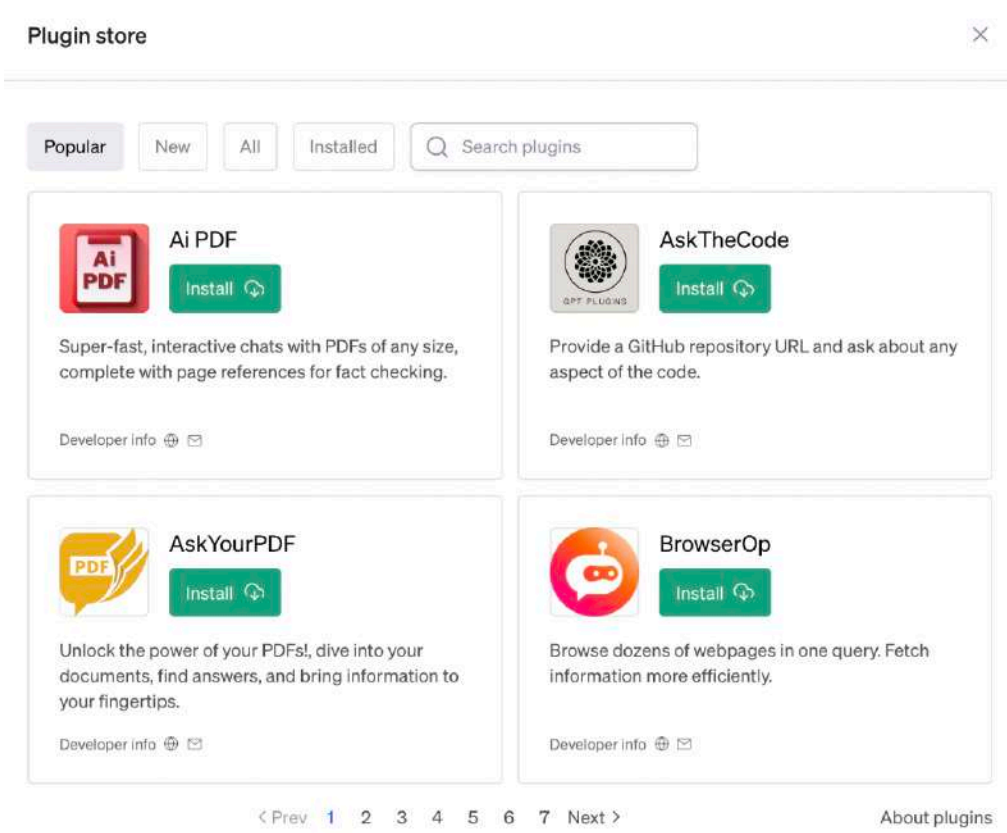
Exploring ChatGPT Plans

- REMEMBER: These plans will change with time
  - Latest: ***https://help.openai.com/en/collections/3742473-chatgpt***
  - **ChatGPT Free**: GPT-3.5 access
  - **ChatGPT Plus**: Paid Subscription Plan
    - GPT-4 access (within a few limits)
    - Faster response speed
    - Priority Access to New Features
      - Plugins
      - Advanced Data Analysis
    - REMEMBER: OpenAI API access is NOT included
  - **ChatGPT Enterprise**: Enterprise-grade security and privacy
    - Unlimited higher-speed GPT-4 access
    - Longer context windows (32k token context window)
    - Shareable chat templates (across an enterprise)



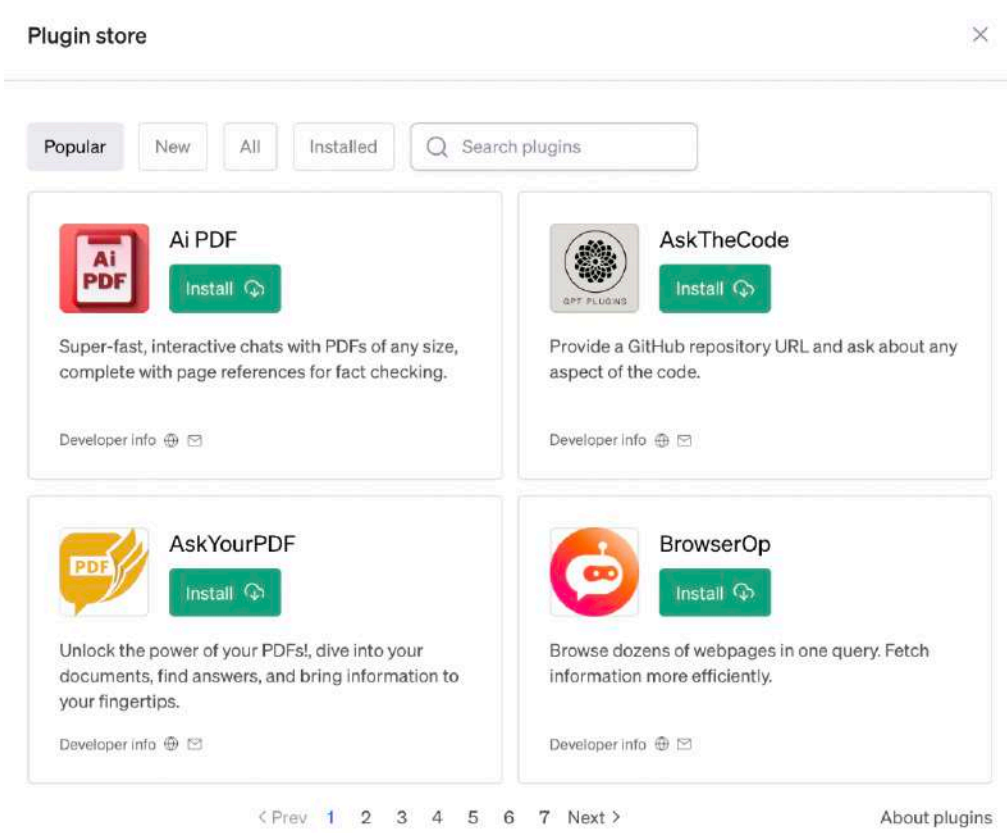
Getting Started with ChatGPT Plugins

- **ChatGPT Plugins**: Add amazing new features to ChatGPT
- Powered by **third party** applications
  - NOT controlled by OpenAI
  - Only enable those you trust
- **REMEMBER:** ChatGPT may send your conversation and location (country or state) you're in to the plugin
- ChatGPT determines when to use plugins during a conversation
  - Depending on the plugins you've enabled
- Paid Feature (Needs to be Enabled!)



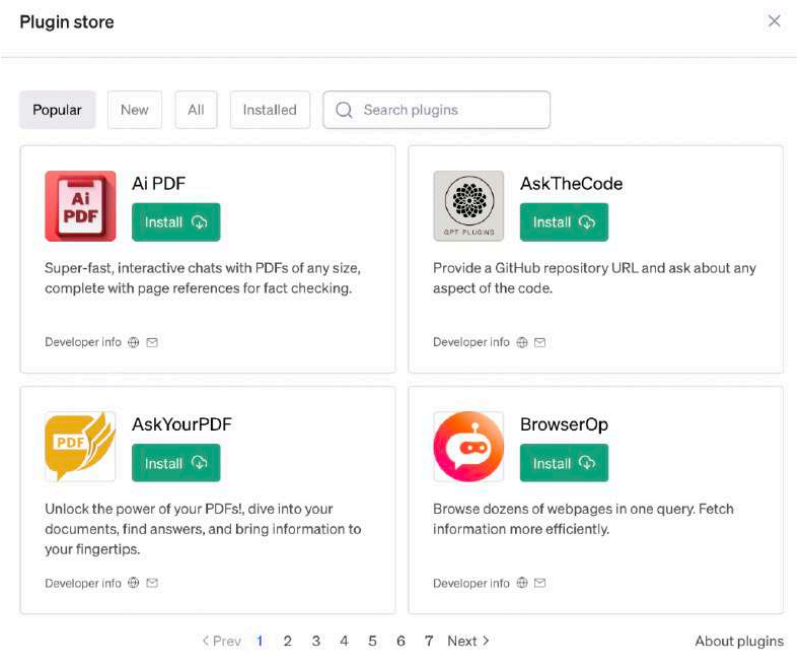
Exploring ChatGPT Plugins

- **Wikipedia** - Tell me about 2028 Summer Olympics
- **Kayak** - I want to fly from New Delhi to New York tomorrow. Can you suggest the best flight?
- **Link Reader** - Can you summarise content from this page - ***https://www.udemy.com/user/in28minutes***
- **Prompt Perfect** - perfect - tell me about in28minutes



Exploring ChatGPT Plugins - 2

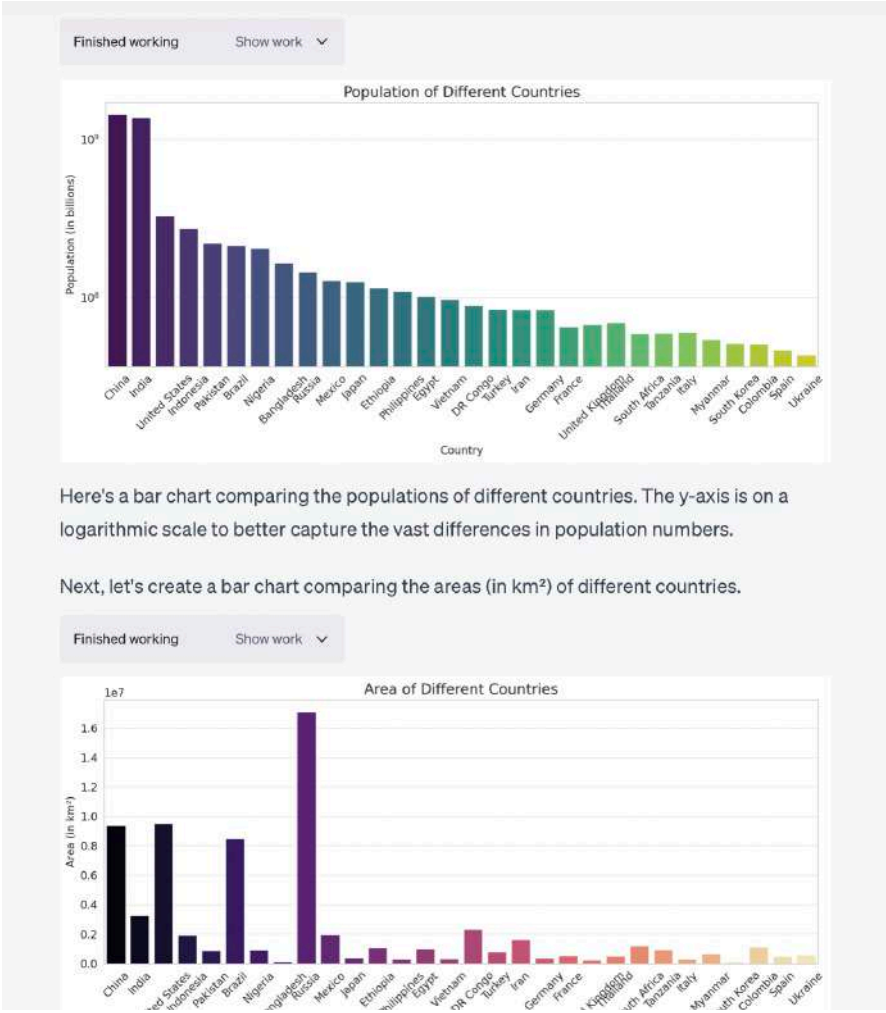
- **daigr.am** - Create a line chart showing the average global temperature for each decade from 1900 to 2020.
- **Diagrams:Show Me** - Create a diagram explaining how a HTTP request is handled
  - show ideas
  - Add More Details and Include Error Handling
- **Make a Sheet** - Create a CSV file with details of Top 10 countries by population - Include capital city, area and population as columns.
- **ScholarAI** - Can you tell me about some latest research on ChatGPT models?



Exploring ChatGPT Advanced Data Analysis

In 28 Minutes

- "**Analyze information in seconds**" (Verbatim - OpenAI announcement)
  - Earlier called "Code Interpreter"
  - Uses Python code
  - Paid Feature (Needs to be Enabled!)
  - *Let's Play*:
    - Can you describe the data?
      - Create few visualizations to understand the data
      - Add a row to CSV with population density
      - Show me 5 example rows please
      - Calculate Density to 1 decimal point accuracy
      - Find top 5 countries by population density
    - Can you show this image?
      - Can you convert it to grayscale?
      - Can you reduce the height and width by 50% each?
      - What image formats have smaller size?



Get Ready

Let's clap for you!

- Congratulations
- You have put your best foot forward to start your Generative AI Journey!
- Good Luck!
- Keep Learning Every Day!

Do Not Forget

- Recommend the course to your friends!
  - Do not forget to review!
- Your Success = My Success
  - Share your success story with me on LinkedIn (Ranga Karanam)
  - Share your success story and lessons learnt in Q&A with other learners!

What Next?

FASTEST ROADMAPS

in28minutes.com

In28 Minutes

- Google Cloud Certifications
- Azure Certifications
- AWS Certifications
- DevOps
- Java Full Stack
- Java Microservices