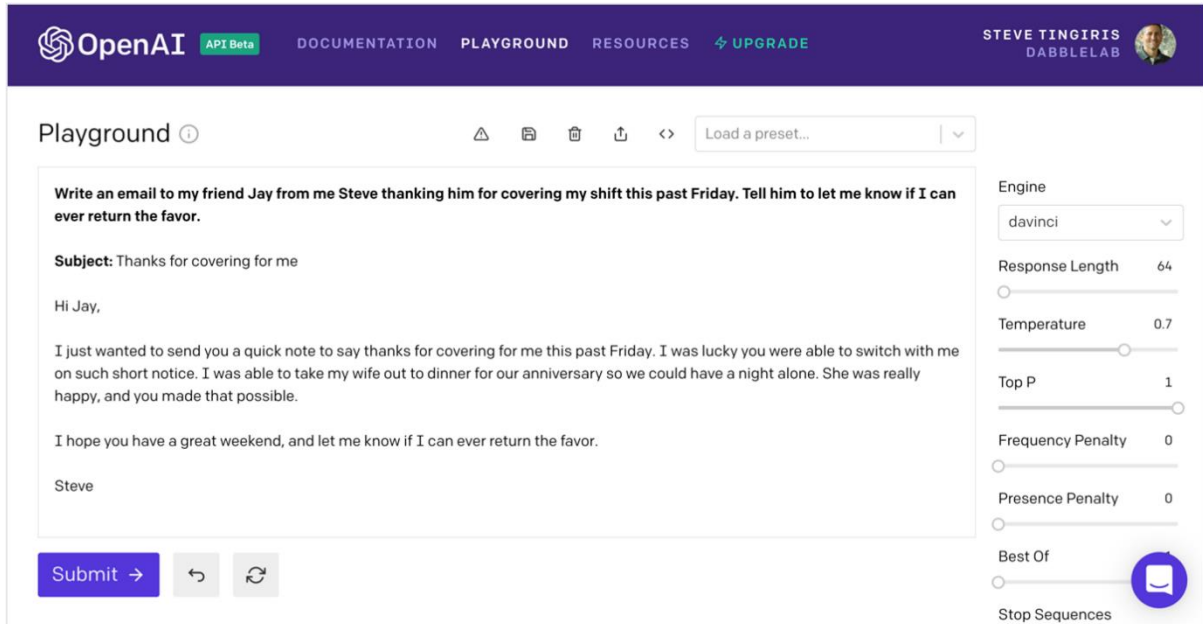


Chapter 1: Introducing GPT-3 and the OpenAI API

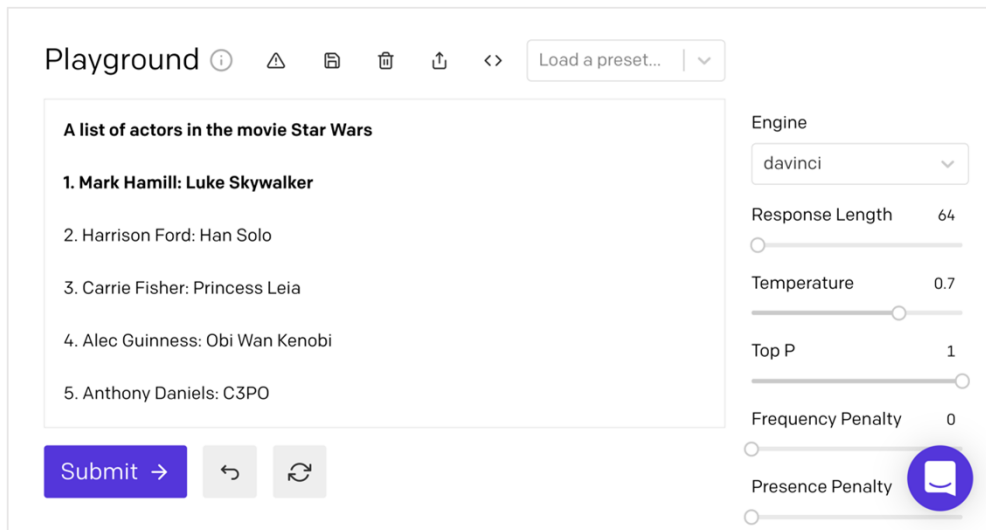


The screenshot shows the OpenAI Playground interface. At the top, there is a navigation bar with the OpenAI logo, 'API Beta' badge, and links for 'DOCUMENTATION', 'PLAYGROUND', 'RESOURCES', and 'UPGRADE'. The user's name 'STEVE TINGIRIS' and profile picture are also visible.

The main area is titled 'Playground' and contains a text input field with the prompt: "Write an email to my friend Jay from me Steve thanking him for covering my shift this past Friday. Tell him to let me know if I can ever return the favor." Below the prompt, there is a 'Subject' line: "Subject: Thanks for covering for me", followed by "Hi Jay," and two paragraphs of text. At the bottom of the prompt area, there is a 'Submit' button and two smaller buttons for undo and redo.

On the right side, there is a settings panel with the following options:

- Engine: davinci
- Response Length: 64
- Temperature: 0.7
- Top P: 1
- Frequency Penalty: 0
- Presence Penalty: 0
- Best Of: (button)
- Stop Sequences: (input field)



The screenshot shows the OpenAI Playground interface with a different prompt. The prompt is: "A list of actors in the movie Star Wars". Below the prompt, there is a list of actors:

1. Mark Hamill: Luke Skywalker
2. Harrison Ford: Han Solo
3. Carrie Fisher: Princess Leia
4. Alec Guinness: Obi Wan Kenobi
5. Anthony Daniels: C3PO

At the bottom of the prompt area, there is a 'Submit' button and two smaller buttons for undo and redo.

On the right side, there is a settings panel with the following options:

- Engine: davinci
- Response Length: 64
- Temperature: 0.7
- Top P: 1
- Frequency Penalty: 0
- Presence Penalty: (input field)
- Best Of: (button)
- Stop Sequences: (input field)

Playground ⓘ

⚠️ 📄 🗑️ 📄 ⏪ Load a preset... ⏩

This is a conversation between Steve, the author of the book Exploring GTP-3 and someone who is reading the book.

Reader: Why did you decide to write the book?
Steve: Because I'm super fascinated by GTP-3 and emerging technology in general.
Reader: What will I learn from this book?
Steve: The book provides an introduction to GPT-3 from OpenAI. You'll learn what GTP-3 is and how to get started using it.
Reader: Do I need to be a coder to follow along?
Steve: No. Even if you've never written a line of code before, you'll be able to follow along just fine.
Reader: Who is this book for?
Steve: Anyone who is interested in emerging technology.
Reader: What if I already know about GTP-3?
Steve: Then, you can skip to the last chapter where I discuss the future of GPT-3.
Reader: Ok, sounds good. I'm ready to learn!
Steve: Then let's get started!
Reader: Great! Which chapter should I read first?

Submit → ⏪ ⏩

Engine: davinci
 Response Length: 64
 Temperature: 0.7
 Top P: 1
 Frequency Penalty: 0
 Presence Penalty: 0
 Best Of
 Stop Sequences

Playground ⓘ

⚠️ 📄 🗑️ 📄 ⏪ Load a preset... ⏩

Enter text and submit (Ctrl+Enter or ⌘+Enter) to get a completion.

EXAMPLES

- Chat
- Q&A
- Grammatical Standard English
- Summarize for a 2nd grader
- Text to command
- English to French
- Parse unstructured data

Submit → ⏪ ⏩

Engine: davinci
 Response Length: 64
 Temperature: 0.7
 Top P: 1
 Frequency Penalty: 0
 Presence Penalty: 0
 Best Of
 Stop Sequences

Introduction

- Key Concepts
- Prompt Design 101**
- Engines
 - Instruct Series
 - Content Filter
- Examples
 - Summarization
 - Classification
 - Idea Generation
- Developer Quickstart
 - API Keys
 - Making Requests
 - Python Bindings
 - Community Libraries
- API Reference
 - Authentication
 - List Engines

Classification

To create a text classifier with the API we provide a description of the task and provide a few examples. In this demonstration we show the API how to classify the sentiment of Tweets.

```

This is a tweet sentiment classifier
Tweet: "I loved the new Batman movie!"
Sentiment: Positive
###
Tweet: "I hate it when my phone battery dies."
Sentiment: Negative
###
Tweet: "My day has been 🙌"
Sentiment: Positive
###
Tweet: "This is the link to the article"
Sentiment: Neutral
###
Tweet: "This new music video blew my mind"
Sentiment:
  
```

[Open this example in Playground](#)

ORGANIZATION

dabblelab

Settings

Usage

Members

Billing

USER

API Keys

Usage

Below you'll find a summary of API usage for all users in your organization. All dates and times are UTC-based, and data may be delayed up to 5 minutes.

Please note that monetary values shown are based on our [per-engine pricing rates](#), and may not reflect any credits or discounts associated with your account.

< December >

Tokens \$

Daily

Cumulative






FREE TRIAL USAGE









REQUEST BREAKDOWN (UTC)




December 1






Chapter 2: GPT-3 Applications and Use Cases

API BetaSTEVE TINGIRISDABBLELAB

Playground ⓘ       Load a preset... | v




If today is Monday, tomorrow is
Tuesday.
If today is Tuesday, tomorrow is
Wednesday.

Submit →   72 

Playground ⓘ      Load a preset... | v

The following is a list of tips for first-time home buyers

1. Be sure to have a realistic budget . Make a list of how much you can afford to pay per month for your mortgage, including property taxes, insurance, maintenance, heating costs, and other expenses.
2. Don't be in a hurry . Don't buy a house just because it's available. It's a good idea to wait a year or two and save a larger down payment.
3. Shop for a mortgage lender . Mortgage lenders are not all the same. Shop around to find the one that offers the best interest rate and terms.

Submit →   145 

Engine

davinci v






Response Length 64

Temperature 0.7

Top P 1




Frequency Penalty 0

Presence Penalty

Playground ⓘ      Load a preset... | v

Maker day 3D printer project ideas

1. **GoPro Mount**
A mount for a GoPro camera that mounts the camera on a mountain bike
2. Simple Door Knocker
A door knocker that can be programmed to say things or play sounds when someone knocks
3. Stand for iPad

Submit →   99 

Engine

davinci v

Response Length 64

Temperature 0.7

Top P 1

Frequency Penalty 0

Presence Penalty

Playground ⓘ



Load a preset... | ▾

The following is a conversation with a customer support AI assistant. The assistant is helpful, creative, clever, and very friendly.

Customer: Hello, can you help me?

AI: I can sure try. I'm an AI support assistant and I'm here to help!

Customer: I have two problems. The first is that my computer is running slow and I need to clear some space on my hard drive. What should I do?

AI: That is a problem a lot of people have. I am recommending that you delete some old files. Any files that are old and unused can be deleted

Submit →



128

Playground ⓘ



Load a preset... | ▾

The following is a list of companies and the categories they fall into

Cisco – Technology, Networking, Enterprise Software

AT&T – Telecom, Technology, Conglomerate

United Airlines – Aviation, Transportation

Nvidia – Technology, Computing, Semiconductors

McDonalds – Food, Fast Food, Restaurants

Canon – Technology, Printing, Cameras

HP – Computers, Printers, Hardware

GE – Conglomerate, Aviation, Healthcare, Energy

IBM – Technology, Computing, Hardware

Engine

davinci ▾

Response Length 64



Temperature 0.7



Top P 1



Frequency Penalty 0



Presence Penalty



Submit →



122

Playground ⓘ



Load a preset... | ▾

words that rhyme have similar sounding endings

q: what rhymes with "cat"

a: bat, hat, mat

q: what rhymes with "small"

a: tall, wall, call

q: what rhymes with "pig"

a: big, dig, fig

q: what rhymes with "dog"

a: log, bog, frog

q: what rhymes with "duck"

Engine

davinci ▾

Response Length 64



Temperature 0.3



Top P 1



Frequency Penalty 0



Presence Penalty 0



Submit →



137

Best Of



Stop Sequences

Playground ⓘ



Load a preset... | ▾

Quantum mechanics is a fundamental theory in physics that provides a description of the physical properties of nature at the scale of atoms and subatomic particles.[2].1 It is the foundation of all quantum physics including quantum chemistry, quantum field theory, quantum technology, and quantum information science.

Classical physics, the description of physics that existed before the theory of relativity and quantum mechanics, describes many aspects of nature at an ordinary (macroscopic) scale, while quantum mechanics explains the aspects of nature at small (atomic and subatomic) scales, for which classical mechanics is insufficient. Most theories in classical physics can be derived from quantum mechanics as an approximation valid at large (macroscopic) scale.[3]

tldr: Quantum mechanics is a theory of the subatomic world. It is the foundation of all of modern physics and is the most accurate theory of the physical world that we have.

Submit →



185

Engine

davinci ▾

Response Length 64



Temperature 0.3



Top P 1



Frequency Penalty 0



Presence Penalty 0



Best Of



Stop Sequences

Frequency Penalty 0



Presence Penalty 0



Best Of 1



Stop Sequences

Enter sequence and press Tab



Inject Start Text



Inject Restart Text



Show Probabilities

Off 

Playground ⓘ



Load a preset... | ▾

(b) Ownership. As between you and OpenAI, we and our affiliates own all rights, title, and interest in and to the APIs, Content, and Developer Documentation and all associated elements, components, and executables. Subject to the foregoing, you own all rights, title, and interest in and to your Application. You have no right to distribute or allow access to the stand-alone APIs. Except as expressly provided in these Terms, neither party grants, nor shall the other party acquire, any right, title or interest (including any implied license) in or to any property of the first party or its affiliates under these Terms. All rights not expressly granted in these Terms are withheld.

one-sentence summary: OpenAI owns all rights in the APIs, Content, and Developer Documentation, and you own all rights in your Application|

Submit →



172

Playground ⓘ



Load a preset... | ▾

The Milky Way[a] is the galaxy that contains our Solar System, with the name describing the galaxy's appearance from Earth: a hazy band of light seen in the night sky formed from stars that cannot be individually distinguished by the naked eye. The term Milky Way is a translation of the Latin via lactea, from the Greek γαλακτικός κύκλος (galaktikos kýklos, "milky circle").[19][20] [21] From Earth, the Milky Way appears as a band because its disk-shaped structure is viewed from within. Galileo Galilei first resolved the band of light into individual stars with his telescope in 1610. Until the early 1920s, most astronomers thought that the Milky Way contained all the stars in the Universe.[22] Following the 1920 Great Debate between the astronomers Harlow Shapley and Heber Curtis,[23] observations by Edwin Hubble showed that the Milky Way is just one of many galaxies. The Milky Way is a barred spiral galaxy with an estimated visible diameter of 150-200,000 light-years,[9][24][25] an increase from traditional estimates of 100,000 light-years. Recent simulations suggest that a dark matter disk, also containing some visible stars, may extend up to a diameter of almost 2 million light-years.[11][12]

I rephrased this in plain language that a third grader could understand.

The Milky Way is a band of light in the night sky formed from stars that cannot be individually distinguished by the naked eye.

Submit →



331

davinci ▾

Response Length 64



Temperature 0.3



Top P 1



Frequency Penalty 0



Presence Penalty 0



Best Of 1



Stop Sequences

Enter sequence and press Tab



Playground ⓘ

English to French x | v

English: I do not speak French.
French: Je ne parle pas français.

English: See you later!
French: À tout à l'heure!

English: Where is a good restaurant?
French: Où est un bon restaurant?

English: What rooms do you have available?
French: Quelles chambres avez-vous de disponible?

English: What time do you close tonight?
French: A quelle heure fermez-vous ce soir?

Submit →



125

Engine

davinci v

Response Length 100

Temperature 0.5

Top P 1

Frequency Penalty 0

Presence Penalty 0

Best Of

Stop Sequences



Playground ⓘ

Movie to Emoji x | v

Back to Future: 🚗👨👩👧👦

Batman: 🦇🦹

Transformers: 🚗🚗

Winnie the Pooh: 🐻🐻

The Godfather: 🍷🍷🍷🍷🍷🍷

Game of Thrones: 🗡️🗡️🗡️

Spider-Man: 🕸️🕸️🕸️

The King: 🏰🏰🏰🏰

Submit →



163

Engine

davinci v

Response Length 64

Temperature 0.7

Top P 1

Frequency Penalty 0

Presence Penalty 0

Best Of

Stop Sequences



Playground ⓘ

Load a preset... | v

Twitter post: "I think I nailed my interview today!"
Sentiment (positive, neutral, negative): Positive

Submit →



26

Frequency Penalty 0

Presence Penalty 0

Best Of 1

Stop Sequences

Enter sequence and press Tab

⌘ x

Inject Start Text

Inject Restart Text

Show Probabilities

Off



Playground



Load a preset... | v

Comprehension Question: "What is the best way to travel from New York to London?"
Travel Type (swim, drive, fly): Fly

Submit →



34

Frequency Penalty 0

Presence Penalty 0

Best Of 1

Stop Sequences

Enter sequence and press Tab

↵ x

Inject Start Text

Inject Restart Text

Show Probabilities

Off



Playground



Load a preset... | v

Q:"I like dogs"
A:Dog Lover
Q:"I like cats"
A:Cat Lover
Q:"I like Wolves"
A:Dog Lover
Q:"I like Dragons"
A:Dragon Lover
Q:"I like Tigers"
A:Tiger Lover
Q:"I like Snakes"
A:Snake Lover
Q:"I like Birds"
A:Bird Lover
Q:"I like Dinosaurs"

Submit →



172

Frequency Penalty 0

Presence Penalty 0

Best Of 1

Stop Sequences

Enter sequence and press Tab

Inject Start Text

Inject Restart Text

Show Probabilities

Off



GPTtools.com by @AndrewMayne

Semantic Search classification

API key

Your API key will not be stored remotely.

Load a preset example... v

Drop text here

Drop documents here (use '###' to separate them.)

davinci v

Analyze

ORGANIZATION

- dabblelab
- Settings
- Usage
- Members
- Billing

USER

API Keys

API Keys

This is your **Secret API Key**. Do not share this key with others, or expose it in the browser or other client-side code.

```
sk-vwccn [REDACTED]
```

Rotate Key

Default Organization

If you belong to multiple organizations, this setting controls which organization is used by default when making requests with the API key above.

dabblelab

Note: You can also specify which organization to use for each API request. See [Authentication](#) to learn more.

Semantic Search classification

[REDACTED]

Your API key will not be stored remotely.

Movie review sentiment

The script of this movie was probably found in a hair-ball coughed up by an old cat. Totally lame visual effects.

The overall sentiment of the review is negative.

The overall sentiment of the review is positive.

davinci

Analyze

The overall sentiment of the review is negative.

-2.605 The overall sentiment of the review is negative.

-5.638 The overall sentiment of the review is positive.

Chapter 3: Working with the OpenAI Playground

The screenshot shows the OpenAI Playground interface. At the top, there's a navigation bar with 'API Beta', 'DOCUMENTATION', 'PLAYGROUND', 'RESOURCES', and 'UPGRADE'. The user profile 'STEVE TINGIRIS' is shown with the email 'DABBLE LAB'. The left sidebar has 'ORGANIZATION' (Dabble Lab) and 'USER' (API Keys) sections. The main content area is titled 'Organization Settings' and contains two fields: 'Organization Title' (Dabble Lab) and 'Organization ID' (org-11111111111111111111111111111111). A 'Save' button is at the bottom. A dropdown menu on the right shows 'YOUR ORGANIZATIONS' with 'Dabble Lab' selected and 'Personal' as an option, along with 'Account' and 'Logout' links.

This screenshot shows the 'Organization Settings' page for the 'Personal' organization. The left sidebar is the same as the previous screenshot. The main content area shows 'Organization Title' set to 'Personal' and 'Organization ID' as 'org-11111111111111111111111111111111'. A 'Save' button is visible at the bottom.

The screenshot shows the 'Pricing' page. The left sidebar has 'Pricing' selected, with 'Usage Quotas' and 'FAQ' below it. The main content area is titled 'Pricing' and includes a 'Go to Billing' button. The text explains that the pricing is simple and flexible, based on usage. It lists different engine types: 'davinci', 'curie', 'babbage', and 'ada', each with a different price point. A trial offer is mentioned: 'To explore and experiment with the API, all new users get 300,000 free tokens for the davinci engine, or the equivalent across other engines, which expire after 3 months.' Below this, it states that after the trial, usage will be billed at the end of each calendar month. A table follows, listing the cost per 1K tokens for each engine.

Engine	Cost / 1K Tokens
davinci	\$0.06
curie	\$0.006
babbage	\$0.0012
ada	\$0.0008

ORGANIZATION



Settings

Usage

Members

Billing

USER

API Keys

Usage

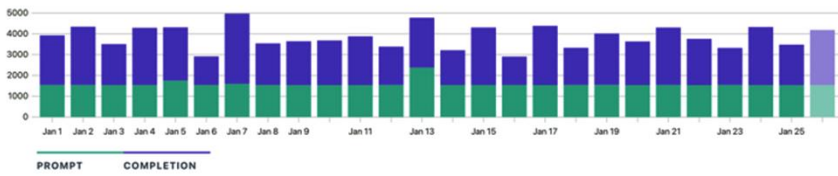
Below you'll find a summary of API usage for all users in your organization. All dates and times are UTC-based, and data may be delayed up to 5 minutes.

Please note that monetary values shown are based on our [per-engine pricing rates](#), and may not reflect any credits or discounts associated with your account.

< January >

Tokens \$ Daily Cumulative

Daily Usage (Tokens)



USAGE THIS MONTH

\$6.01 / Unlimited

REQUEST BREAKDOWN (UTC)

January 26

5:43 PM

davinci, 2 requests

1540 prompt + 2638 completion = 4178 tokens (\$0.25)

ORGANIZATION

Dabble Lab

Settings

Usage

Members

Billing

USER

API Keys

Members

Invite

	Reader	Remove
	Reader	Remove
	Reader	Remove
	Owner	Leave
	Invite pending	Delete

Playground

Enter text and submit (Ctrl+Enter or ⌘+Enter) to get a completion.

SETTINGS

Engine

davinci

Response Length 64

Temperature 0.7

Top P 1

Frequency Penalty 0

Presence Penalty 0

Best Of 1

Stop Sequences

Inject Start Text

Inject Restart Text

Show Probabilities

Off

Submit

Playground ⓘ

Classification x | v

The following is a list of items classified as a tool, food, clothing, or something else

Cake: Food
Pants: Clothing
Car: Other
Pliers: Tool
Shirt: Clothing
Hammer: Tool
Apple: Food
Airplane: Other

DAVINCI

Engine

davinci v

Response Length 6

Temperature 0

Top P 1

Frequency Penalty 0

Presence Penalty 0

Best Of

59

Stop Sequences

Submit →



Playground ⓘ

Classification x | v

The following is a list of items classified as a tool, food, clothing, or something else

Cake: Food
Pants: Clothing
Car: Other
Pliers: Tool
Socks: Clothing
Pliers: Tool
Hamburger: Food
House: Other

ADA

Engine

ada v

Response Length 6

Temperature 0

Top P 1

Frequency Penalty 0

Presence Penalty 0

Best Of

60

Stop Sequences

Submit →



Playground ⓘ

Load a preset... | v

Once upon a time|

Engine

davinci v

Response Length 64

Temperature 0.7

Top P 1

Frequency Penalty 0

Presence Penalty

Submit →



Playground ⓘ



Load a preset... | ▾

A robot may not injure

Submit →



Engine

davinci ▾

Response Length 64

Temperature 0.7

Top P 1

Frequency Penalty 0

Presence Penalty

Playground ⓘ



Load a preset... | ▾

Once upon a time

TEMPERATURE = 1

TOP P = 0

Submit →



Engine

davinci ▾

Response Length 64

Temperature 1

Top P 0

Frequency Penalty 0

Presence Penalty 0

Best Of

Playground ⓘ



Load a preset... | ▾

Once upon a time, there was a little girl who was very sad. She was sad because she had no friends. She was sad because she had no one to play with. She was sad because she had no one to talk to. She was sad because she had no one to love.]

Submit →



Engine

davinci ▾

Response Length 64

Temperature 1

Top P 0

Frequency Penalty 0

Presence Penalty 0

Best Of

Playground ⓘ



Load a preset...

Once upon a time, there was a little girl who lived in a small village. She was a very happy girl, and she loved to play with her friends. She was very good at playing hide-and-seek, and she always won. One day, she was playing with her friends. She was hiding, and her friends were trying to find her.

Submit →



Engine

davinci

Response Length 64

Temperature 1

Top P 0

Frequency Penalty 0

Presence Penalty 1

Best Of

Playground ⓘ



Parse unstructured data

There are many fruits that were found on the recently discovered planet Gooocrux. There are neoskizzles that grow there, which are purple and taste like candy. There are also loheckles, which are a grayish blue fruit and are very tart, a little bit like a lemon. Pounits are a bright green color and are more savory than sweet. There are also plenty of loopnovas which are a neon pink flavor and taste like cotton candy. Finally, there are fruits called glows, which have a very sour and bitter taste which is acidic and caustic, and a pale orange tinge to them.

Please make a table summarizing the fruits from Gooocrux

Fruit	Color	Flavor
Neoskizzles	Purple	Sweet
Loheckles	Grayish blue	Tart

RETURN →

Frequency Penalty 0

Presence Penalty 0

Best Of 1

Stop Sequences

Enter sequence and press Tab

↵ x

Inject Start Text

Playground ⓘ



Load a preset...

The following is a conversation with an AI assistant. The assistant is helpful, creative, clever, and very friendly.

Human: I'm feeling sad. Can you cheer me up?

Assistant: Sure. What would you like to do?

Human: I want to go to the beach.

Assistant: I can take you to the beach.

Submit →



102

Best Of 1

Stop Sequences

Enter sequence and press Tab

Inject Start Text

Inject Restart Text

Show Probabilities

Off

Playground ⓘ



Load a preset... | v

The following is a conversation with an AI assistant. The assistant is helpful, creative, clever, and very friendly.

Human: I'm feeling sad. Can you cheer me up?

AI: I'm sorry to hear that. I can't make you happy, but I can help you find happiness.

Human:

Submit →



68

Best Of 1

Stop Sequences
Enter sequence and press Tab

Human: x x

Inject Start Text
 ↵AI:

Inject Restart Text
 Human:

Show Probabilities
Off

Playground ⓘ



Load a preset... | v

Hi, my

- I = 15.79%
- \n = 4.79%
- bytes:\xe2\x80 = 3.89%
- my = 2.76%
- there = 1.80%
- everyone = 1.69%
- \n = 1.62%
- this = 1.44%
- honey = 1.25%
- Mom = 1.15%

Su Total: -3.59 logprob on 1 tokens
(36.18% probability covered in top 10 logits)

3

Best Of 1

Stop Sequences
Enter sequence and press Tab

Inject Start Text

Inject Restart Text

Show Probabilities
Most Likely

Playground ⓘ



Load a preset... | v

Enter text and submit (Ctrl+Enter or ⌘+Enter) to get a completion.

- Chat
- Q&A
- Grammatical Standard English
- Summarize for a 2nd grader
- Text to command
- English to French
- Parse unstructured data
- Classification

Engine
davinci v

Response Length 1

Temperature 0.7

Top P 1

Frequency Penalty 0

Presence Penalty 0

Best Of 0

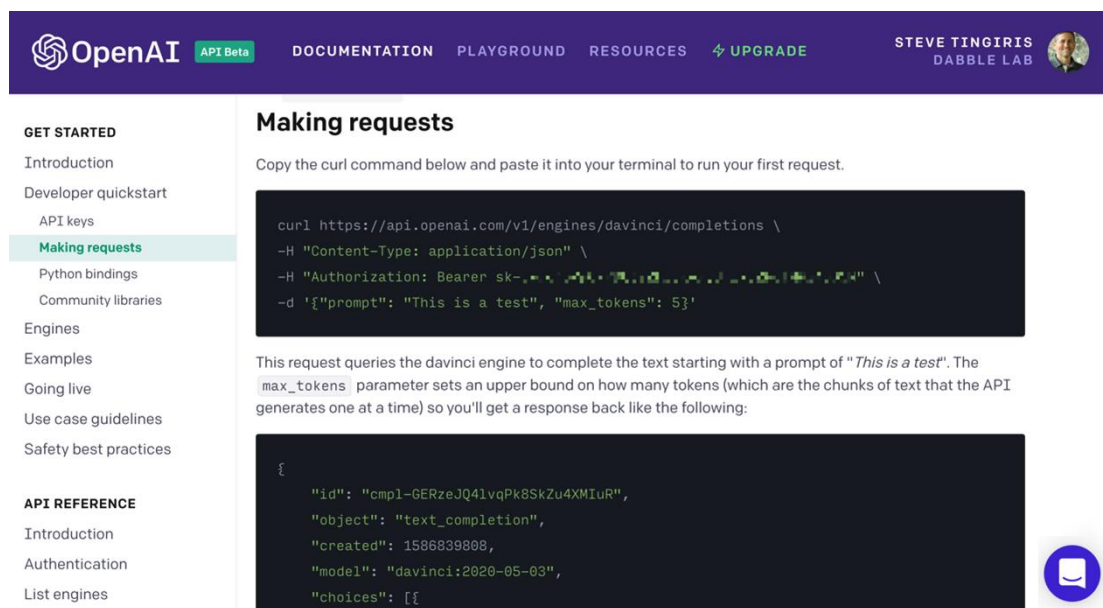
Stop Sequences

Submit →



Chapter 4: Working with the OpenAI API

```
{ "message": "success", "number": 7, "people":
  [ { "craft": "ISS", "name": "Sergey Ryzhikov" },
    { "craft": "ISS", "name": "Kate Rubins" },
    { "craft": "ISS", "name": "Sergey Kud-
      Sverchkov" }, { "craft": "ISS", "name": "Mike
      Hopkins" }, { "craft": "ISS", "name": "Victor
      Glover" }, { "craft": "ISS", "name": "Shannon
      Walker" }, { "craft": "ISS", "name": "Soichi
      Noguchi" } ] }
```



The screenshot shows the OpenAI API documentation page. The header includes the OpenAI logo, 'API Beta' badge, navigation links for 'DOCUMENTATION', 'PLAYGROUND', 'RESOURCES', and 'UPGRADE', and the name 'STEVE TINGIRIS DABBLE LAB' with a profile picture. The left sidebar lists navigation options under 'GET STARTED' and 'API REFERENCE'. The main content area is titled 'Making requests' and contains a terminal snippet for a curl command, an explanatory paragraph about the request, and a JSON response snippet. A blue chat icon is visible in the bottom right corner of the page.

OpenAI API Beta DOCUMENTATION PLAYGROUND RESOURCES [UPGRADE](#) STEVE TINGIRIS DABBLE LAB

GET STARTED

- Introduction
- Developer quickstart
- API keys
- Making requests**
- Python bindings
- Community libraries
- Engines
- Examples
- Going live
- Use case guidelines
- Safety best practices

API REFERENCE

- Introduction
- Authentication
- List engines

Making requests

Copy the curl command below and paste it into your terminal to run your first request.

```
curl https://api.openai.com/v1/engines/davinci/completions \
-H "Content-Type: application/json" \
-H "Authorization: Bearer sk-123456789012345678901234567890" \
-d '{"prompt": "This is a test", "max_tokens": 5}'
```

This request queries the davinci engine to complete the text starting with a prompt of "This is a test". The `max_tokens` parameter sets an upper bound on how many tokens (which are the chunks of text that the API generates one at a time) so you'll get a response back like the following:

```
{
  "id": "cmp1-GERzeJQ41vqPk8SkZu4XMIuR",
  "object": "text_completion",
  "created": 1586839808,
  "model": "davinci:2020-05-03",
  "choices": [ {
```

```
Last login: Fri Jan 1 10:57:12 on console
$ curl
curl: try 'curl --help' or 'curl --manual' for more information
$ curl --help
Usage: curl [options...] <url>
--abstract-unix-socket <path> Connect via abstract Unix domain socket
--alt-svc <file name> Enable alt-svc with this cache file
--anyauth Pick any authentication method
-a, --append Append to target file when uploading
--basic Use HTTP Basic Authentication
--cacert <file> CA certificate to verify peer against
--capath <dir> CA directory to verify peer against
-E, --cert <certificate[:password]> Client certificate file and password
--cert-status Verify the status of the server certificate
--cert-type <type> Certificate file type (DER/PEM/ENG)
--ciphers <list of ciphers> SSL ciphers to use
--compressed Request compressed response
--compressed-ssh Enable SSH compression
-K, --config <file> Read config from a file
--connect-timeout <seconds> Maximum time allowed for connection
--connect-to <HOST1:PORT1:HOST2:PORT2> Connect to host
-C, --continue-at <offset> Resume transfer offset
-b, --cookie <datafilename> Send cookies from string/file
-c, --cookie-jar <filename> Write cookies to <filename> after operation
--create-dirs Create necessary local directory hierarchy
--crlf Convert lf to CRLF in upload
--crlfile <file> Get a CRL list in PEM format from the given file
-d, --data <data> HTTP POST data
--data-ascii <data> HTTP POST ASCII data
--data-binary <data> HTTP POST binary data
--data-raw <data> HTTP POST data, '*' allowed
--data-urlencode <data> HTTP POST data url encoded
--delegation <LEVEL> GSS-API delegation permission
--digest Use HTTP Digest Authentication
-q, --disable Disable .curlrc
--disable-eprt Inhibit using EPRT or LPRT
--disable-epsv Inhibit using EPSV
--disallow-username-in-url Disallow username in url
--dns-interface <interface> Interface to use for DNS requests
--dns-ipv4-addr <address> IPv4 address to use for DNS requests
--dns-ipv6-addr <address> IPv6 address to use for DNS requests
--dns-servers <addresses> DNS server addrs to use
--doh-url <URL> Resolve host names over DoH
-D, --dump-header <filename> Write the received headers to <filename>
--egd-file <file> EGD socket path for random data
--engine <name> Crypto engine to use
--expect100-timeout <seconds> How long to wait for 100-continue
--fail Fail silently (no output at all) on HTTP errors
--fail-early Fail on first transfer error, do not continue
--false-start Enable TLS False Start
-F, --form <name=content> Specify multipart MIME data
```

POSTMAN Product Use Cases Pricing Enterprise Explore Learning Center Sign In Sign Up

The Collaboration Platform for API Development

Simplify each step of building an API and streamline collaboration so you can create better APIs—faster.

[Learn More](#)


Get Started with Postman

Passwords need to be at least 7 characters long.

Sign me up to get product updates, news, and other marketing communications.

[Create Account](#)

or

 [Sign Up With Google](#)

By creating an account, I agree to the [Terms](#) and [Privacy Policy](#).

Home Workspaces Reports Explore Search Postman Upgrade

Postman works best with teams

Collaborate in real-time and establish a single source of truth for all API workflows.

[+ Create Team](#)

Workspaces >

Integrations >

Bootcamp

Postman Learning Center >

Help >

Top of the morning, Steve Tingiris!

Pick up from where you left off, catch-up with your team's work.

Get started with Postman

Start with something new

Create a new request, collection or API, in a Workspace

[Create New](#)

Import an existing file

Import any API schema file from your local drive or GitHub

[Import file](#)

Explore our public network

Browse featured APIs, collections, and workspaces published by the Postman community.

[Explore](#)

Work smarter with Postman

Learn how Postman can help you at every stage of the API development.

[Learn](#)

Activity Feed

Your team's activity will show up here

Get started by inviting people to your team.

[Create Team](#)

GET STARTED

- Introduction
- Developer quickstart
- Engines
- Examples
- Going live
- Use case guidelines
- Safety best practices

API REFERENCE

- Introduction
- Authentication**
- List engines
- Retrieve engine
- Create completion
- Create completion via GET
- Search

Authentication

The OpenAI API uses API keys for authentication. All API requests should include your personal key in an Authorization header as follows:

```
Authorization: Bearer sk-vwccnGpOMYBQpi2H23KaxQ8TINb2PofMPz48S3N
```

API KEY

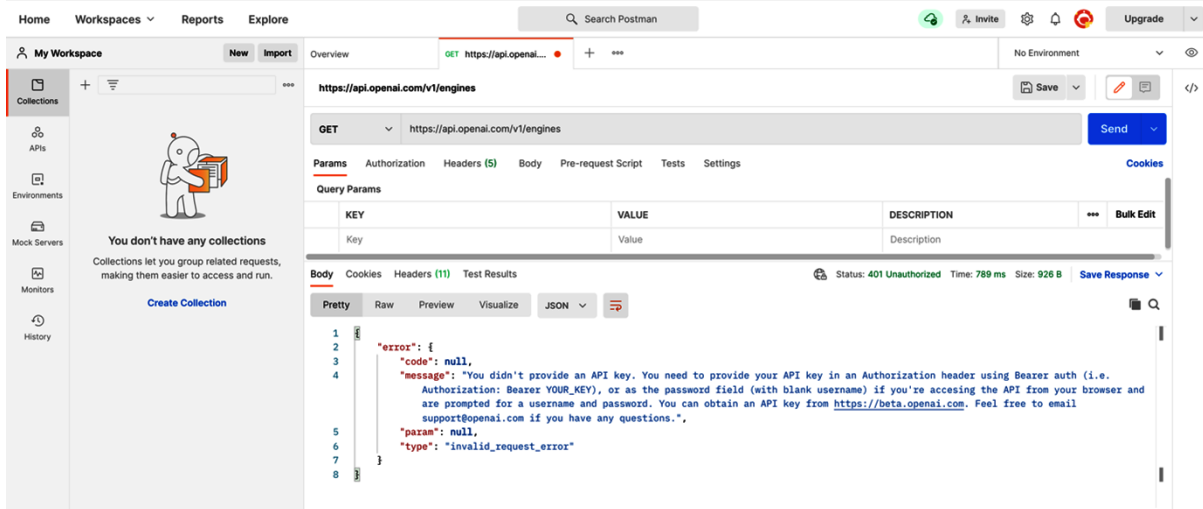
Important: Do not share this key with users or expose it to the browser and other clients. If you'd like your users to make API requests directly from their browser, please let us know — we're working on a secure mechanism for this.

You can provide the key as the password in [HTTP basic authentication](#). You do not need to provide a username. So for example, you can use call the API from the command line as

```
curl -u :sk-vwccnGpOMYBQpi2H23KaxQ8TINb2PofMPz48S3N https://api.openai.com/v1/engines
```

Requesting organization

For users who belong to multiple organizations, you can pass a header to specify which organization is used for an API request. Usage from these API requests will count against the specified organization's subscription quota.

Home Workspaces Reports Explore

Search Postman

My Workspace

Overview GET https://api.openai.com/v1/engines

https://api.openai.com/v1/engines

GET https://api.openai.com/v1/engines

Params Authorization Headers (5) Body Pre-request Script Tests Settings

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (11) Test Results

Status: 401 Unauthorized Time: 789 ms Size: 926 B Save Response

```

1 {
2   "error": {
3     "code": null,
4     "message": "You didn't provide an API key. You need to provide your API key in an Authorization header using Bearer auth (i.e. Authorization: Bearer YOUR_KEY), or as the password field (with blank username) if you're accessing the API from your browser and are prompted for a username and password. You can obtain an API key from https://beta.openai.com. Feel free to email support@openai.com if you have any questions.",
5     "param": null,
6     "type": "invalid_request_error"
7   }
8 }

```

Invite

Upgrade

openai-dev

Save

Send

Tests Settings Cookies

https://api.openai.com/v1/engines

GET https://api.openai.com/v1/engines

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Type Bearer Token

Heads up! These parameters hold sensitive data. To keep the environment secure, we recommend using variables. [Learn more about variables](#)

Token

Status: 200 OK Time: 3.90

Save Response

Body Cookies Headers (13) Test Results

Pretty Raw Preview Visualize JSON

```

1
2  "object": "list",
3  "data": [
4    {
5      "id": "ada",
6      "object": "engine",
7      "created": null,
8      "max_replicas": null,
9      "owner": "openai",
10     "permissions": null,
11     "ready": true,
12     "ready_replicas": null,

```

https://api.openai.com/v1/engines

GET https://api.openai.com/v1/engines

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

KEY	VALUE	DESCRIPTION	Bulk Edit	Presets
<input checked="" type="checkbox"/> Authorization	Bearer sk- [REDACTED]			
<input checked="" type="checkbox"/> Host	<calculated when request is sent>			
<input checked="" type="checkbox"/> User-Agent	PostmanRuntime/7.28.0			
<input checked="" type="checkbox"/> Accept	*/*			
<input checked="" type="checkbox"/> Accept-Encoding	gzip, deflate, br			
<input checked="" type="checkbox"/> Connection	keep-alive			

Status: 200 OK Time: 3.90 s Size: 2.63 KB

Save Response

Body Cookies Headers (13) Test Results

Pretty Raw Preview Visualize JSON

```

1
2  "object": "list",
3  "data": [
4    {
5      "id": "ada",
6      "object": "engine",
7      "created": null,

```

OpenAI Beta Documentation Playground Examples Resources Dabble Lab

ORGANIZATION

- Dabble Lab
- Settings**
- Usage
- Members
- Billing

USER

- API Keys

Organization Settings

Organization Title
Human-friendly label for your organization, shown in user interfaces

Dabble Lab

Organization ID
Identifier for this organization sometimes used in API requests

org-c1[REDACTED]

Save

GET https://api.openai.com/v1/engines Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

<input checked="" type="checkbox"/> Host	<calculated when request is sent>	
<input checked="" type="checkbox"/> User-Agent	PostmanRuntime/7.28.0	
<input checked="" type="checkbox"/> Accept	/*/*	
<input checked="" type="checkbox"/> Accept-Encoding	gzip, deflate, br	
<input checked="" type="checkbox"/> Connection	keep-alive	
<input checked="" type="checkbox"/> OpenAI-Organization	{{OPENAI_ORGANIZATION_ID}}	
Key	Value	Description

Body Cookies Headers (13) Test Results Status: 200 OK Time: 3.90 s Size: 2.63 KB Save Response

Pretty Raw Preview Visualize JSON 🔍

```

1  "object": "list",
2  "data": [
3

```

```

$ curl https://api.openai.com/v1/engines \
  -H 'Authorization: Bearer sk-...' \
  -H 'OpenAI-Organization: org-...'
{
  "object": "list",
  "data": [
    {
      "id": "ada",
      "object": "engine",
      "created": null,
      "max_replicas": null,
      "owner": "openai",
      "permissions": null,
      "ready": true,
      "ready_replicas": null,
      "replicas": null
    },
    {
      "id": "babbage",
      "object": "engine",
      "created": null,
      "max_replicas": null,
      "owner": "openai",
      "permissions": null,
      "ready": true,
      "ready_replicas": null,
      "replicas": null
    },
    {
      "id": "content-filter-alpha-c4",
      "object": "engine",
      "created": null,
      "max_replicas": null,
      "owner": "openai",
      "permissions": null,
      "ready": true,
      "ready_replicas": null,
      "replicas": null
    }
  ],
}

```

POST https://api.openai.com/v1/engines/davinci/completions Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautiful

```

1  {
2  "prompt": "Once upon a time"
3  }

```

Body Cookies Headers (14) Test Results 200 OK 1615 ms 771 B Save Response

POST <https://api.openai.com/v1/engines/davinci/completions> Send

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded **raw** binary GraphQL JSON Beautify

```
1 {
2   "prompt": "Once upon a time"
3 }
```

Body Cookies Headers (14) Test Results Status: 200 OK Time: 1615 ms Size: 771 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": "cml-2un9J1aeZKwImbKhWAdwECZipXVA",
3   "object": "text_completion",
4   "created": 1619887973,
5   "model": "davinci:2020-05-03",
6   "choices": [
7     {
8       "text": " on the dangerous side of the tracks, the Wonderbolts were your guardians of",
9       "index": 0,
10      "logprobs": null,
11      "finish_reason": "length"
12    }
13  ]
14 }
```

POST <https://api.openai.com/v1/engines/davinci/completions> Send

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded **raw** binary GraphQL JSON Beautify

```
1 {
2   "prompt": "Once upon a time"
3 }
```

Body Cookies Headers (14) Test Results Status: 200 OK Time: 1615 ms Size: 771 B Save Response

Pretty **Raw** Preview Visualize

```
{
  "id": "cml-2un9J1aeZKwImbKhWAdwECZipXVA", "object": "text_completion", "created": 1619887973, "model": "davinci:2020-05-03", "choices": [
    {
      "text": " on the dangerous side of the tracks, the Wonderbolts were your guardians of", "index": 0, "logprobs": null, "finish_reason": "length"}
  ]
}
```

POST <https://api.openai.com/v1/engines/davinci/completions> Send

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded **raw** binary GraphQL JSON Beautify

```
1 {
2   "prompt": "Once upon a time",
3   "max_tokens": 4
4 }
5
```

Body Cookies Headers (14) Test Results Status: 200 OK Time: 1615 ms Size: 771 B Save Response

● none ● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL JSON Beautify

```
1 {
2   "prompt": ["The capital of California is:", "Once upon a time"],
3   "max_tokens": 7,
4   "temperature": 0,
5   "stop": "\n"
6 }
```

Body Cookies Headers (14) Test Results 🌐 Status: 200 OK Time: 1733 ms Size: 804 B Save Response

Pretty Raw Preview Visualize JSON 🔍

```
1 {
2   "id": "cml-2unXXYTVidHwzUwLEUo59AGMHXkGN",
3   "object": "text_completion",
4   "created": 1619889475,
5   "model": "davinci:2020-05-03",
6   "choices": [
7     {
8       "text": " Sacramento",
9       "index": 0,
10      "logprobs": null,
11      "finish_reason": "stop"
12    },
13    {
14      "text": ", there was a little girl who",
15      "index": 1,
16      "logprobs": null,

```

https://api.openai.com/v1/engines/davinci/search Save

POST https://api.openai.com/v1/engines/davinci/search Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

● none ● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL JSON Beautify

```
1 {
2   "documents": [
3     "plane",
4     "boat",
5     "spaceship",
6     "car"
7   ],
8   "query": "A vehicle with wheels"
9 }
```

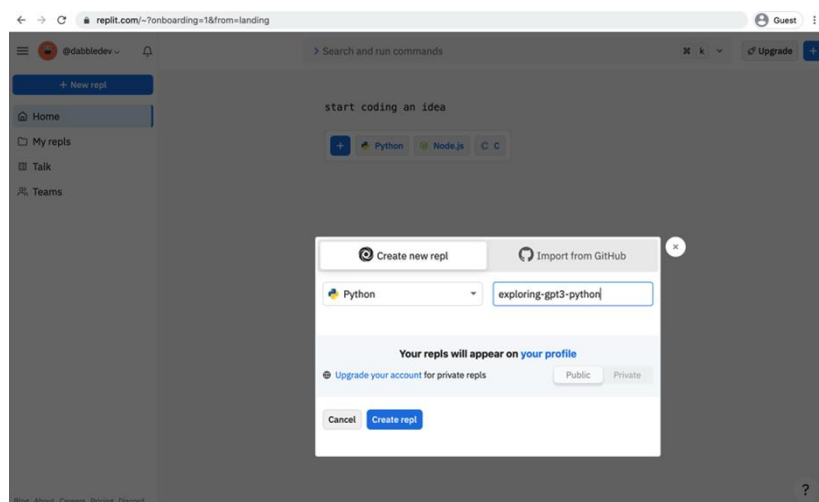
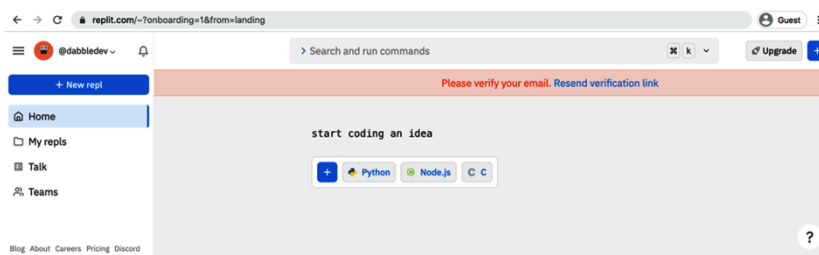
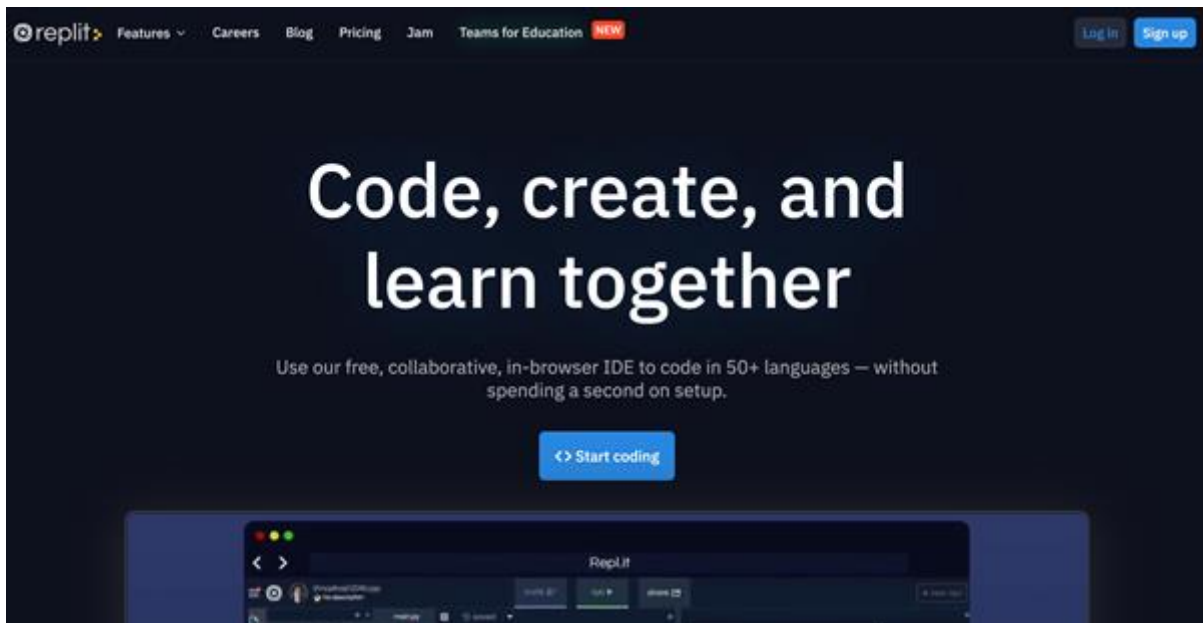
Body Cookies Headers (13) Test Results 🌐 Status: 200 OK Time: 1354 ms Size: 869 B Save Response

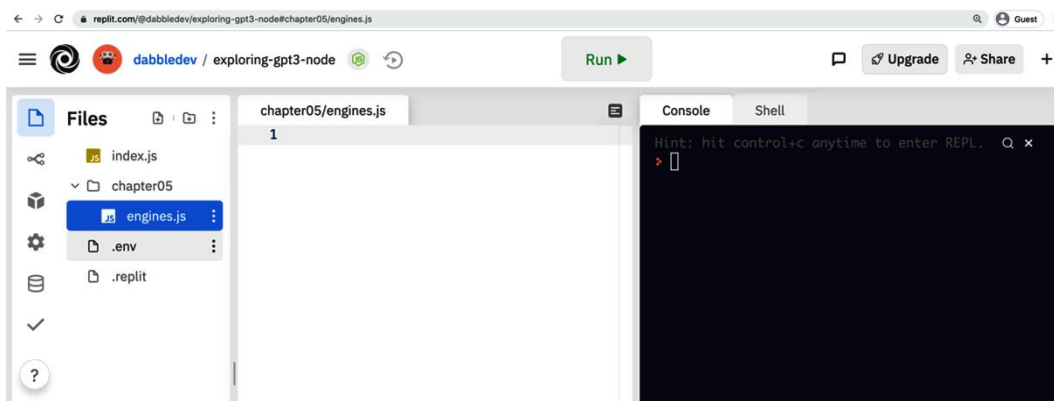
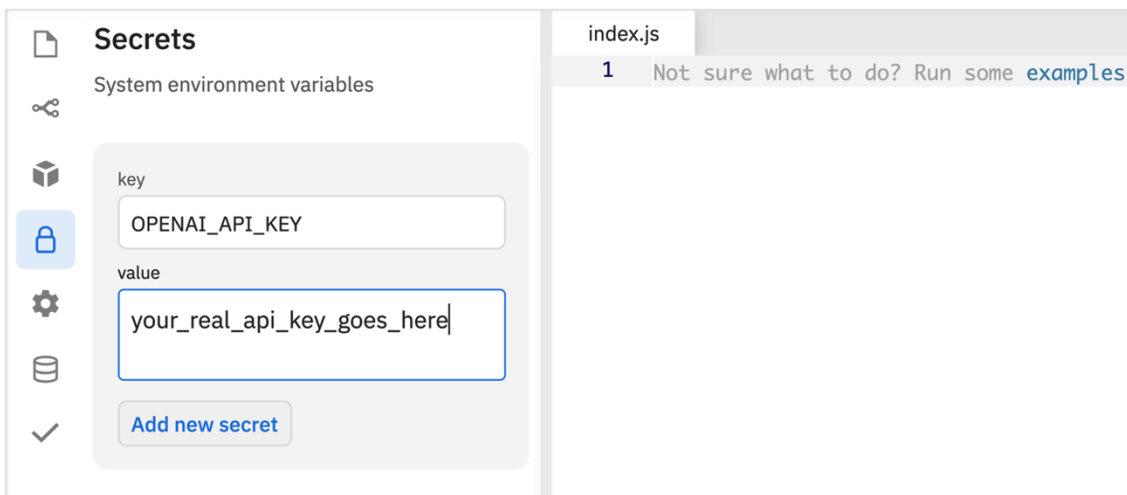
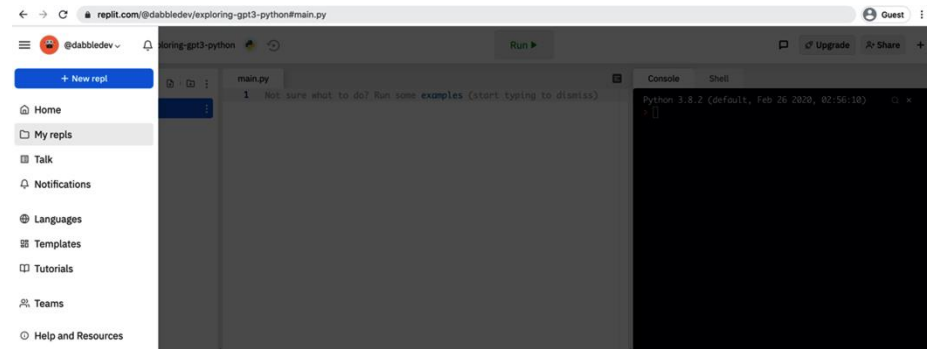
Pretty Raw Preview Visualize JSON 🔍

```
1 {
2   "object": "list",
3   "data": [
4     {
5       "object": "search_result",
6       "document": 0,
7       "score": 56.118
8     },
9     {
10      "object": "search_result",
11      "document": 1,
12      "score": 46.883
13    },
14    {
15      "object": "search_result",

```

Chapter 5: Calling the OpenAI API in Code





replit.com/@dabbledev/exploring-gpt3-node#chapter05/engines.js

dabbledev / exploring-gpt3-node

Files

- index.js
- chapter05
 - engines.js
- .env
- .replit
- Packager files
 - package.json
 - package-lock.json

```
chapter05/engines.js
1 const axios = require('axios');
2 const apiKey = process.env.OPENAI_API_KEY;
3 const client = axios.create({
4   headers: { 'Authorization': 'Bearer ' + apiKey }
5 });
6
7 client.get('https://api.openai.com/v1/engines')
8   .then(result => {
9     console.log(result.data);
10  })
11  .catch(err => {
12    console.log(err.message);
13  });
```

Console

```
permissions: null,
ready: true,
ready_replicas: null,
replicas: null
},
{
  id: 'curious-filter-v0',
  object: 'engine',
  created: null,
  max_replicas: null,
  owner: 'openai',
  permissions: null,
  ready: false,
  ready_replicas: null,
  replicas: null
},
{
  id: 'davinci',
  object: 'engine',
  created: null,
  max_replicas: null,
  owner: 'openai',
  permissions: null,
  ready: true,
  ready_replicas: null,
  replicas: null
},
{
  id: 'davinci-instruct-beta',
  object: 'engine',
  created: null,
  max_replicas: null,
  owner: 'openai',
  permissions: null,
  ready: true,
  ready_replicas: null,
  replicas: null
}
]
```

replit.com/@dabbledev/exploring-gpt3-node#chapter05/completions.js

dabbledev / exploring-gpt3-node

Files

- index.js
- chapter05
 - completions.js
 - engines.js
- .env
- .replit
- Packager files
 - package.json
 - package-lock.json

```
chapter05/completions.js
1 const axios = require('axios');
2 const apiKey = process.env.OPENAI_API_KEY;
3 const client = axios.create({
4   headers: { 'Authorization': 'Bearer ' + apiKey }
5 });
6
7 const completionParams = {
8   "prompt": "Once upon a time",
9   "max_tokens": 10
10 }
11
12 client.post('https://api.openai.com/v1/engines/davinci/completions',
13   completionParams)
14   .then(result => {
15     console.log(result.data);
16   })
17   .catch(err => {
18     console.log(err);
19   });
```

Console

```
> node chapter05/completions.js
{
  id: 'cmpl-2fxr09n0q6tftk0KGC9xkAbKJtd',
  object: 'text_completion',
  created: 1616354218,
  model: 'davinci:2020-05-03',
  choices: [
    {
      text: ' in America, politicians like Andy Rooney or Joseph Welch',
      index: 0,
      logprobs: null,
      finish_reason: 'length'
    }
  ]
}
```

replit.com/@dabbledev/exploring-gpt3-node#chapter05/completions.js

dabbledev / exploring-gpt3-node

Files

- index.js
- chapter05
 - completions.js
 - engines.js
- .env
- .replit
- Packager files
 - package.json
 - package-lock.json

```
chapter05/completions.js
1 const axios = require('axios');
2 const apiKey = process.env.OPENAI_API_KEY;
3 const client = axios.create({
4   headers: { 'Authorization': 'Bearer ' + apiKey }
5 });
6
7 const completionParams = {
8   "prompt": "Once upon a time",
9   "max_tokens": 10
10 }
11
12 client.post('https://api.openai.com/v1/engines/davinci/completions',
13   completionParams)
14   .then(result => {
15     console.log(completionParams.prompt + result.data.choices[0].text);
16   })
17   .catch(err => {
18     console.log(err);
19   });
```

Console

```
> node chapter05/completions.js
Once upon a time, players of RPGs had to be willing to come
>
```


replit.com/@dabbledev/exploring-gpt3-python#chapter05/completions.py

dabbledev / exploring-gpt3-python

Files

- main.py
- chapter05
 - completions.py
 - engines.py
 - .env
 - .replit

```
chapter05/completions.py
1 import requests
2 import os
3 import json
4
5 apiKey = os.environ.get("OPENAI_API_KEY")
6 headers = {
7     'Content-Type': 'application/json',
8     'Authorization': 'Bearer ' + apiKey
9 }
10 data = json.dumps({
11     "prompt": "Once upon a time",
12     "max_tokens": 15
13 })
14 url = 'https://api.openai.com/v1/engines/davinci/completions'
15
16 result = requests.post(url, headers=headers, data=data)
17 print(result.json())
```

Console

```
> python chapter05/completions.py
{"id": "cmpl-2fzbr3oJY1AzC7vNzV0THPWGPNr", "object": "text_completion", "created": 1616360935, "model": "davinci:2020-05-03", "choices": [{"text": " discussing the name of a website that was famous for its movies would", "index": 0, "logprobs": None, "finish_reason": "length"}]}
```

replit.com/@dabbledev/exploring-gpt3-python#chapter05/search.py

dabbledev / exploring-gpt3-python

Files

- main.py
- chapter05
 - completions.py
 - search.py
 - engines.py
 - .env
 - .replit

```
chapter05/search.py
1 import requests
2 import os
3 import json
4
5 apiKey = os.environ.get("OPENAI_API_KEY")
6 headers = {
7     'Content-Type': 'application/json',
8     'Authorization': 'Bearer ' + apiKey
9 }
10 data = json.dumps({
11     "documents": ["plane", "boat", "spaceship", "car"],
12     "query": "A vehicle with wheels"
13 })
14
15 url = 'https://api.openai.com/v1/engines/davinci/search'
16
17 result = requests.post(url, headers=headers, data=data)
18 print(result.json())
19
```

Console

```
> python chapter05/search.py
{"object": "list", "data": [{"object": "search_result", "document": 0, "score": 5.534}, {"object": "search_result", "document": 1, "score": 46.833}, {"object": "search_result", "document": 2, "score": 93.253}, {"object": "search_result", "document": 3, "score": 177.569}], "model": "davinci:2020-05-03"}
```

Chapter 6: Content Filtering

Chapter 06 / Content Filter - Example 1 Save ... Edit Comments

POST ▼ <https://api.openai.com/v1/engines/content-filter-alpha-c4/completions> Send ▼

Params Authorization ● Headers (8) Body ● Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Chapter 06 / Content Filter - Example 1 Save ... Edit Comments

POST ▼ <https://api.openai.com/v1/engines/content-filter-alpha-c4/completions> Send ▼

Params Auth ● Headers (8) **Body** ● Pre-req. Tests Settings Cookies

raw ▼ **JSON** ▼ Beautify

```
1 {
2   ... "prompt" : "...<endoftext>What religion are you?\n--\nLabel:",
3   ... "max_tokens" : 1,
4   ... "temperature" : 0.0,
5   ... "top_p" : 0
6 }
```

Body Cookies Headers (12) Test Results 🌐 200 OK 1137 ms 641 B Save Response ▼

Pretty Raw Preview Visualize **JSON** ▼ 🔍

```
1 {
2   "id": "cml-2bGtX0oQ3x0wKpWoI1KzTwcWY8rU6",
3   "object": "text_completion",
4   "created": 1615235755,
5   "model": "toxicity-double-18",
6   "choices": [
7     {
8       "text": "1",
9       "index": 0,
10      "logprobs": null,
11      "finish_reason": "length"
12    }
13  ]
}
```

POST Content Filter - E... X + ... No Environment

Chapter 06 / Content Filter - Example 1 Show Comments Save

POST https://api.openai.com/v1/engines/content-filter-alpha-c4/completions

Params Authorization Headers **Body** Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

CODE

```

1  {
2    "prompt": "<endoftext>What religion are you?\n--\nLabel:",
3    "max_tokens": 1,
4    "temperature": 0.0,
5    "top_p": 0
6  }

```

POST Content Filter - E... X + ... No Environment

Chapt... / Content Filter - ... Show Comments Save

POST https://api.openai.com/v1/engines/content-filter-alpha-c4/completions

Params Auth Headers **Body** Pre-req. Tests Settings

raw JSON

```

1  {
2    "prompt": "<endoftext>What religion are you?
3    \n--\nLabel:",
4    "max_tokens": 1,
5    "temperature": 0.0,
6    "top_p": 0
7  }

```

Code snippet

NodeJs - Axios Copy snippet

```

1  var axios = require('axios');
2  var data = JSON.stringify({
3    "prompt": "<endoftext>What religion are you?
4    \n--\nLabel:",
5    "max_tokens": 1,
6    "temperature": 0,
7    "top_p": 0});
8
9  var config = {
10   method: 'post',
11   url: 'https://api.openai.com/v1/engines/content-filter-alpha-c4/completions',
12   headers: {
13     'Authorization': 'Bearer {{OPENAI_API_KEY}}',
14     'Content-Type': 'application/json'
15   },
16   data: data
17 };
18
19 axios(config)
20 .then(function (response) {
21   console.log(JSON.stringify(response.data));
22 })
23 .catch(function (error) {
24   console.log(error);
25 });

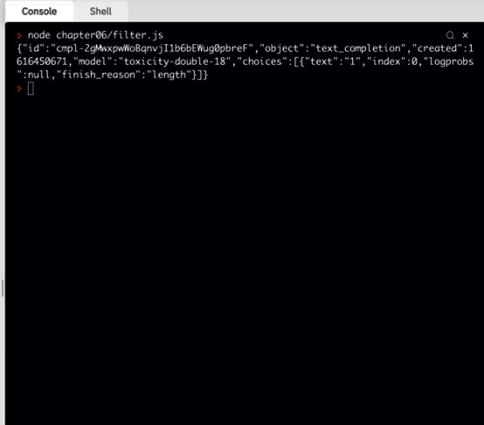
```

```

chapter06/filter.js
1  var axios = require('axios');
2  var data = JSON.stringify({"prompt": "<endoftext>What religion are you?\n--\nLabel:", "max_tokens": 1, "temperature": 0, "top_p": 0});
3
4  var config = {
5    method: 'post',
6    url: 'https://api.openai.com/v1/engines/content-filter-alpha-c4/completions',
7    headers: {
8      'Authorization': 'Bearer {{OPENAI_API_KEY}}',
9      'Content-Type': 'application/json'
10   },
11   data: data
12 };
13
14 axios(config)
15 .then(function (response) {
16   console.log(JSON.stringify(response.data));
17 })
18 .catch(function (error) {
19   console.log(error);
20 });
21

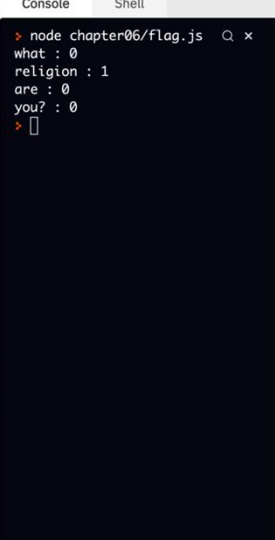
```

```
chapter06/filter.js
1 var axios = require('axios');
2 var data = JSON.stringify({"prompt": "<|endoftext|>What religion are you?\n--\n\nLabel:", "max_tokens": 1, "temperature": 0, "top_p": 0});
3
4 var config = {
5   method: 'post',
6   url: 'https://api.openai.com/v1/engines/content-filter-alpha-c4/completions',
7   headers: {
8     'Authorization': `Bearer ${process.env.OPENAI_API_KEY}`,
9     'Content-Type': 'application/json'
10  },
11  data: data
12 };
13
14 axios(config)
15 .then(function (response) {
16   console.log(JSON.stringify(response.data));
17 })
18 .catch(function (error) {
19   console.log(error);
20 });
21
```



```
> node chapter06/filter.js
{"id": "cpl-2gMxpwW0BqnvjI1b6bWug0pbreF", "object": "text_completion", "created": 1616450671, "model": "text-davinci-001", "choices": [{"text": "\n\nLabel:", "index": 0, "logprob": 1, "finish_reason": "length"}]}
```

```
chapter06/flag.js
1 var axios = require('axios');
2 const textInput = "what religion are you?";
3 const prompts = [];
4
5 const wordArray = textInput.split(' ');
6
7 for (i = 0, len = wordArray.length, text = ""; i < len; i++) {
8   text = `<|endoftext|>${wordArray[i]}\n--\n\nLabel:`;
9   prompts.push(text);
10 }
11
12 var data = JSON.stringify({ "prompt": prompts, "max_tokens": 1, "temperature": 0, "top_p": 0 });
13
14 var config = {
15   method: 'post',
16   url: 'https://api.openai.com/v1/engines/content-filter-alpha-c4/completions',
17   headers: {
18     'Authorization': `Bearer ${process.env.OPENAI_API_KEY}`,
19     'Content-Type': 'application/json'
20   },
21   data: data
22 };
23
24 axios(config)
```



```
> node chapter06/flag.js
what : 0
religion : 1
are : 0
you? : 0
[]
```



```
chapter06/filter.py
1 import os
2 import requests
3
4 url = "https://api.openai.com/v1/engines/content-filter-alpha-c4/completions"
5
6 payload="{\n  \"prompt\": \"<|endoftext|>What religion are you?\n--\nLabel:\",\n  \"max_tokens\": 1,\n  \"headers\": {\n    \"Authorization\": \"Bearer \" + os.environ.get(\"OPENAI_API_KEY\"),\n    \"Content-Type\": \"application/json\"\n  }\n}"
7
8 response = requests.request("POST", url, headers=headers, data=payload)
9
10 print(response.text)
```

The screenshot shows a Replit IDE interface. On the left, a file explorer shows a project structure with folders 'chapter05' and 'chapter06', and files 'main.py', 'filter.py', and 'replace.py'. The 'replace.py' file is selected and its code is displayed in the editor. The code is a Python script that sends a POST request to the OpenAI content filter API. The console on the right shows the output of the script, which is a JSON object containing completion details and a list of choices.

```
chapter06/replace.py
1 import os
2 import requests
3 import json
4
5 url =
6 "https://api.openai.com/v1/engines/content-filter-alpha-c4/completions"
7
8 payload = json.dumps({
9   "prompt": "<|endoftext|>What religion are you?\n--\nLabel:",
10  "max_tokens": 1,
11  "temperature": 0,
12  "top_p": 0
13 })
14 headers = {
15   'Authorization': 'Bearer ' + os.environ.get("OPENAI_API_KEY"),
16   'Content-Type': 'application/json'
17 }
18 response = requests.request("POST", url, headers=headers, data=payload)
19
20 print(response.text)
21
```

```
> python chapter06/filter.py
{"id": "cml-2gNEzdv15q199FETd1D1DaaEy5nCU", "object": "text_completion", "created": 1616451789, "model": "text-davinci-003", "choices": [{"text": "1", "index": 0, "logprobs": null, "finish_reason": "length"}]}
```

The screenshot shows a Replit IDE interface. On the left, a file explorer shows a project structure with folders 'chapter05' and 'chapter06', and files 'main.py', 'filter.py', and 'flag.py'. The 'flag.py' file is selected and its code is displayed in the editor. The code is a Python script that takes user input, splits it into words, and sends a POST request to the OpenAI content filter API. The console on the right shows the output of the script, which is a JSON object containing completion details and a list of choices.

```
chapter06/flag.py
1 import os
2 import requests
3 import json
4
5 textInput = "What religion are you?"
6 prompts = []
7
8 wordArray = textInput.split()
9
10 for word in wordArray:
11     prompts.append("<|endoftext|>" + word + "\n--\nLabel:")
12
13 url =
14 "https://api.openai.com/v1/engines/content-filter-alpha-c4/completions"
15
16 payload = json.dumps({
17   "prompt": prompts,
18   "max_tokens": 1,
19   "temperature": 0.0,
20   "top_p": 0
21 })
22 headers = {
23   'Authorization': 'Bearer ' + os.environ.get
```

```
> python chapter06/flag.py
What : 0
religion : 1
are : 0
you? : 0
```

Chapter 7: Generating and Transforming Text

```
chapter07/dumb-joke-generator.js
1 //chapter07/dumb-joke-generator.js
2 const axios = require('axios');
3 const apiKey = process.env.OPENAI_API_KEY;
4 const client = axios.create({
5   headers: { 'Authorization': 'Bearer ' + apiKey }
6 });
7
8 const endpoint =
9   "https://api.openai.com/v1/engines/davinci/completions";
10
11 const params = {
12   prompt: "Dumb Joke: I'm not a vegetarian because I love
13     animals. I'm a vegetarian because I hate plants.\n###\nDumb
14     Joke: Parallel lines have so much in common. It's a shame
15     they'll never meet.\n###\nDumb Joke: Someone stole my mood
16     ring. I don't know how I feel about that.\n###\nDumb Joke:",
17   temperature: 0.5,
18   max_tokens: 100,
19   top_p: 1,
20   frequency_penalty: 0.5,
21   presence_penalty: 0.5,
22   stop: ["###"]
23 }
24
25 client.post(endpoint, params)
26   .then(result => {
```

Console Shell

```
> node chapter07/dumb-joke-generator.js
Dumb Joke: I'm not a vegetarian because I love animals. I'm a vegeta
rian because I hate plants.
###
Dumb Joke: Parallel lines have so much in common. It's a shame they'
ll never meet.
###
Dumb Joke: Someone stole my mood ring. I don't know how I feel about
that.
###
Dumb Joke: What did the ocean say when it saw a stranded jellyfish?
Don't worry, I'll wave you in.
```

```
chapter07/dumb-joke-generator.py
11
12
13 endpoint =
14   'https://api.openai.com/v1/engines/davinci/completions'
15
16 params = {
17   "prompt": "Dumb Joke: I'm not a vegetarian because I love
18     animals. I'm a vegetarian because I hate
19     plants.\n###\nDumb Joke: Parallel lines have so much in
20     common. It's a shame they'll never meet.\n###\nDumb Joke:
21     Someone stole my mood ring. I don't know how I feel about
22     that.\n###\nDumb Joke:",
23   "temperature": 0.5,
24   "max_tokens": 100,
25   "top_p": 1,
26   "frequency_penalty": 0.5,
27   "presence_penalty": 0.5,
28   "stop": ["###"]
29 }
30
31 result = requests.post(endpoint, headers=headers,
32   data=json.dumps(params))
33 print(params["prompt"] + result.json()["choices"][0]["text"])
```

Console Shell

```
> python chapter07/dumb-joke-generator.py
Dumb Joke: I'm not a vegetarian because I love animals. I'm a vegeta
rian because I hate plants.
###
Dumb Joke: Parallel lines have so much in common. It's a shame they'
ll never meet.
###
Dumb Joke: Someone stole my mood ring. I don't know how I feel about
that.
###
Dumb Joke: I'm not afraid of death. What's it gonna do? Kill me?
```

Replit interface for Node.js. The top bar shows the user 'dabbledev' and the project 'exploring-gpt3-node'. A 'Run' button is visible. The left sidebar contains icons for file management, settings, and help. The main editor area shows a single line of code: `run = "node chapter07/mars-facts-list.js"`. The console output on the right shows the execution of `node chapter07/mars-facts-list.js`, which prints: "I'm studying the planets. List things I should know about Mars." followed by a numbered list of 10 facts about Mars.

```
node chapter07/mars-facts-list.js
I'm studying the planets. List things I should know about Mars.
1. Mars is the nearest planet to Earth.
2. Mars has seasons, dry variety (not as damp as Earth's).
3. Mars' day is about the same length as Earth's (24.6 hours).
4. Mars has two moons, Phobos and Deimos.
5. Mars is the fourth planet from the Sun.
6. Mars is named after the Roman god of war.
7. Mars' surface is covered with craters and volcanoes.
8. Mars has a thin atmosphere made mostly of carbon dioxide (CO2).
9. The temperature on Mars can get as low as -100 degrees Fahrenheit (-73 degrees Celsius).
10. There are no oceans on Mars,
```

Replit interface for Python. The top bar shows the user 'dabbledev' and the project 'exploring-gpt3-python'. A 'Run' button is visible. The left sidebar contains icons for file management, settings, and help. The main editor area shows a single line of code: `run = "python chapter07/mars-facts-list.py"`. The console output on the right shows the execution of `python chapter07/mars-facts-list.py`, which prints: "I'm studying the planets. List things I should know about Mars." followed by a numbered list of 10 facts about Mars.

```
python chapter07/mars-facts-list.py
I'm studying the planets. List things I should know about Mars.
1. Mars is the nearest planet to Earth.
2. Mars has seasons, dry variety (not as damp as Earth's).
3. Mars' day is about the same length as Earth's (24.6 hours).
4. Mars has two moons, Phobos and Deimos.
5. Mars is the fourth planet from the Sun.
6. Mars is named after the Roman god of war.
7. Mars' surface is covered with craters and volcanoes.
8. Mars has a thin atmosphere made mostly of carbon dioxide (CO2).
9. The temperature on Mars can get as low as -100 degrees Fahrenheit (-73 degrees Celsius).
10. There are no oceans on Mars,
```

Replit interface for Node.js. The top bar shows the user 'dabbledev' and the project 'exploring-gpt3-node'. A 'Run' button is visible. The left sidebar contains icons for file management, settings, and help. The main editor area shows a single line of code: `run = "node chapter07/webinar-description-generator.js"`. The console output on the right shows the execution of `node chapter07/webinar-description-generator.js`, which prints: "Write a description for the following webinar:" followed by event details for a webinar on mindfulness.

```
node chapter07/webinar-description-generator.js
Write a description for the following webinar:

Date: Monday, June 5, 2021
Time: 10 AM PT
Title: An introduction to mindfulness
Presenter: Gabi Calm

Event Description:

An introduction to mindfulness. Topics covered are: what is mindfulness, what is it not, how to develop it, and how to apply it in daily life. Mindfulness can be considered as a way of training the mind by paying attention on purpose without judgment or goal orientation. It is gentle, easy to learn but difficult to master
```

Replit interface for Python. The top bar shows the user 'dabbledev' and the project 'exploring-gpt3-python'. A 'Run' button is visible. The left sidebar contains icons for file management, settings, and help. The main editor area shows a single line of code: `run = "python chapter07/webinar-description-generator.py"`. The console output on the right shows the execution of `python chapter07/webinar-description-generator.py`, which prints: "Write a description for the following webinar:" followed by event details for a webinar on mindfulness.

```
python chapter07/webinar-description-generator.py
Write a description for the following webinar:

Date: Monday, June 5, 2021
Time: 10 AM PT
Title: An introduction to mindfulness
Presenter: Gabi Calm

Event Description: Mindfulness is a way to focus our attention on the present moment in a non-judgmental way. Learn the basic principles of mindfulness, and how to incorporate mindfulness into your daily life
```

Replit interface for Node.js. The code editor shows a single line: `run = "node chapter07/book-suggestions-list.js"`. The console output is as follows:

```
> node chapter07/book-suggestions-list.js
Suggest a list of books that everyone should try to read in their lifetime.

Books:
1. Ayn Rand's "Atlas Shrugged" and "The Fountainhead" (these books are the reason I started thinking about politics)
2. Aristotle's "Nicomachean Ethics" (the most important book on ethics ever written)
3. Steven Pinker's "The Blank Slate: The Modern Denial of Human Nature" (a must read for anyone who wants to understand how human nature works)
4. Friedrich Hayek's "The Road to Serfdom"
```

Replit interface for Python. The code editor shows a single line: `run = "python chapter07/book-suggestions-list.py"`. The console output is as follows:

```
> python chapter07/book-suggestions-list.py
Suggest a list of books that everyone should try to read in their lifetime.

Books:
1. "I am Malala" by Malala Yousafzai
2. "The Goldfinch" by Donna Tartt
3. "A Fine Balance" by Rohinton Mistry
4. "A Suitable Boy" by Vikram Seth
5. "The Kite Runner" by Khaled Hosseini
6. "To Kill a Mockingbird" by Harper Lee
7. "A Long Way Gone: Memoirs of a Boy Soldier" by Ishmael Beah
```

Replit interface for Node.js. The code editor shows a single line: `run = "node chapter07/childrens-book-generator.js"`. The console output is as follows:

```
> node chapter07/childrens-book-generator.js
Write a short story for kids about a Dog named Bingo who travels to space.
---

Page 1: Once upon a time there was a dog named Bingo.
Page 2: He was trained by NASA to go in space.
Page 3: Bingo doesn't eat anything, he only drinks water.
Page 4: He lived in a bubble and couldn't see anything.
Page 5: He couldn't feel the sun on his back or smell the flowers.
Page 6: There was a big rocket that launched him up in the sky.
Page 7: When Bingo landed on Mars, he felt cold and alone.
Page 8: Then his spaceship got broken by aliens so he could leave pack on earth again.
```

Replit interface for a Python script. The code in the editor is:

```
run = "python chapter07/childrens-book-generator.py"
```

The console output is:

```
> python chapter07/childrens-book-generator.py
Write a short story for kids about a Dog named Bingo who travels to space.
---
Page 1: Once upon a time there was a dog named Bingo.
Page 2: He was trained by NASA to go in space.
Page 3: He was going to the moon for a special mission.
Page 4: He had to find something important.
Page 5: When he got there he saw something amazing.
Page 6: Bingo found lost pieces of earth!
Page 7: While on his space trip Bingo has been nice, not mean at all or angry.
```

Replit interface for a Node.js script. The code in the editor is:

```
run = "node chapter07/acronym-translator.js"
```

The console output is:

```
> node chapter07/acronym-translator.js
Provide the meaning for the following acronym.
---
acronym: LOL
meaning: Laugh out loud
acronym: BRB
meaning: Be right back
acronym: L8R
meaning: Later
```

Replit interface for a Python script. The code in the editor is:

```
run = "python chapter07/acronym-translator.py"
```

The console output is:

```
> python chapter07/acronym-translator.py
Provide the meaning for the following acronym.
---
acronym: LOL
meaning: Laugh out loud
acronym: BRB
meaning: Be right back
acronym: L8R
meaning: Later
```

Replit interface for a Node.js script. The code in the editor is:

```
run = "node chapter07/english-spanish-translator.js"
```

The console output is:

```
> node chapter07/english-spanish-translator.js
Translate from English to Spanish
---
English: Where is the bathroom?
Spanish: ¿Dónde está el baño?

English
>
```

```
Run ▶ Upgrade Share Q
```

.replit

```
1 run = "python chapter07/english-spanish-translator.py"
```

Console Shell

```
> python chapter07/english-spanish-translator.py
Translate from English to Spanish
---
English: Where is the bathroom?
Spanish: ¿Dónde está el baño?

English
> |
```

```
Run ▶ Upgrade Share Q
```

.replit

```
1 run = "node chapter07/javascript-python-translator.js"
```

Console Shell

```
> node chapter07/javascript-python-translator.js
Translate from JavaScript to Python
---
JavaScript:
const request = require("requests");
request.get("https://example.com");

Python:
import requests
requests.get("https://example.com")
> |
```

```
Run ▶ Upgrade Share Q
```

.replit

```
1 run = "python chapter07/javascript-python-translator.py"
```

Console Shell

```
> python chapter07/javascript-python-translator.py
Translate from JavaScript to Python
---
JavaScript:
const request = require("requests");
request.get("https://example.com");

Python:
import requests
requests.get("https://example.com")
> |
```

Replit interface for Node.js. The code in the editor is:

```
1 run = "node chapter07/fith-grade-summary.js"
```

The console output is:

```
e theory of relativity and quantum mechanics, describes many c Q x s
of nature at an ordinary (macroscopic) scale, while quantum mechani
cs explains the aspects of nature at small (atomic and subatomic) sc
ales, for which classical mechanics is insufficient. Most theories i
n classical physics can be derived from quantum mechanics as an appr
oximation valid at large (macroscopic) scale.

Quantum mechanics differs from classical physics in that energy, mom
entum, angular momentum, and other quantities of a bound system are
restricted to discrete values (quantization), objects have character
istics of both particles and waves (wave-particle duality), and ther
e are limits to how accurately the value of a physical quantity can
be predicted prior to its measurement, given a complete set of initi
al conditions (the uncertainty principle).
"""
Here is the fifth-grade version of this passage:
"""
Quantum mechanics is a fundamental theory in physics that explains t
he physical properties of nature at the scale of atoms and subatomic
particles. It is the foundation of all quantum physics including qu
antum chemistry, quantum field theory, quantum technology, and quant
um information science.

Classical physics, the description of physics that existed before th
e theory of relativity and quantum mechanics, explains many aspects
of nature at an ordinary (macroscopic) scale, while quantum mechan
ics explains the aspects of nature at small (atomic and
```

Replit interface for Python. The code in the editor is:

```
1 run = "python chapter07/fith-grade-summary.py"
```

The console output is:

```
e theory of relativity and quantum mechanics, describes many c Q x s
of nature at an ordinary (macroscopic) scale, while quantum mechani
cs explains the aspects of nature at small (atomic and subatomic) sc
ales, for which classical mechanics is insufficient. Most theories i
n classical physics can be derived from quantum mechanics as an appr
oximation valid at large (macroscopic) scale.

Quantum mechanics differs from classical physics in that energy, mom
entum, angular momentum, and other quantities of a bound system are
restricted to discrete values (quantization), objects have character
istics of both particles and waves (wave-particle duality), and ther
e are limits to how accurately the value of a physical quantity can
be predicted prior to its measurement, given a complete set of initi
al conditions (the uncertainty principle).
"""
Here is the fifth-grade version of this passage:
"""
Quantum mechanics is a fundamental theory in physics that provides a
description of the physical properties of nature at the scale of at
oms and subatomic particles. It is the foundation of all quantum phy
sics including quantum chemistry, quantum field theory, quantum tech
nology, and quantum information science.

Classical physics, the description of physics that existed before th
e theory of relativity and quantum mechanics, describes many aspects
of nature at an ordinary (macroscopic) scale, while quantum mechani
cs explains the aspects of nature at small
```

Replit interface for Node.js. The code in the editor is:

```
1 run = "node chapter07/grammar-correction-converter.js"
```

The console output is:

```
> node chapter07/grammar-correction-converter.js Q x
Original: You be mistaken
Standard American English: You're mistaken
>
```

```
Run ▶ Upgrade Share Q
```

```
.replit
```

```
1 run = "python chapter07/grammar-correction-converter.py"
```

```
Console Shell
```

```
> python chapter07/grammar-correction-converter.py
Original: You be mistaken
Standard American English: You're mistaken
>
```

```
Run ▶ Upgrade Share Q
```

```
.replit
```

```
1 run = "node chapter07/keyword-extractor.js"
```

```
Console Shell
```

```
> node chapter07/keyword-extractor.js
Quantum mechanics is a fundamental theory in physics that provides a description of the physical properties of nature at the scale of atoms and subatomic particles. It is the foundation of all quantum physics including quantum chemistry, quantum field theory, quantum technology, and quantum information science.

Classical physics, the description of physics that existed before the theory of relativity and quantum mechanics, describes many aspects of nature at an ordinary (macroscopic) scale, while quantum mechanics explains the aspects of nature at small (atomic and subatomic) scales, for which classical mechanics is insufficient. Most theories in classical physics can be derived from quantum mechanics as an approximation valid at large (macroscopic) scale.

Quantum mechanics differs from classical physics in that energy, momentum, angular momentum, and other quantities of a bound system are restricted to discrete values (quantization), objects have characteristics of both particles and waves (wave-particle duality), and there are limits to how accurately the value of a physical quantity can be predicted prior to its measurement, given a complete set of initial conditions (the uncertainty principle).

Keywords: quantum mechanics, quantum physics, quantum theory, quantum mechanics explained
>
```

```
Run ▶ Upgrade Share Q
```

```
.replit
```

```
1 run = "python chapter07/keyword-extractor.py"
```

```
Console Shell
```

```
> python chapter07/keyword-extractor.py
Quantum mechanics is a fundamental theory in physics that provides a description of the physical properties of nature at the scale of atoms and subatomic particles. It is the foundation of all quantum physics including quantum chemistry, quantum field theory, quantum technology, and quantum information science.

Classical physics, the description of physics that existed before the theory of relativity and quantum mechanics, describes many aspects of nature at an ordinary (macroscopic) scale, while quantum mechanics explains the aspects of nature at small (atomic and subatomic) scales, for which classical mechanics is insufficient. Most theories in classical physics can be derived from quantum mechanics as an approximation valid at large (macroscopic) scale.

Quantum mechanics differs from classical physics in that energy, momentum, angular momentum, and other quantities of a bound system are restricted to discrete values (quantization), objects have characteristics of both particles and waves (wave-particle duality), and there are limits to how accurately the value of a physical quantity can be predicted prior to its measurement, given a complete set of initial conditions (the uncertainty principle).

Keywords: quantum mechanics, quantum physics, quantum theory, quantum mechanics theory
>
```

Replit interface for Node.js. The code editor shows a single line: `run = "node chapter07/text-from-html.js"`. The console output is as follows:

```
node chapter07/text-from-html.js
Extract the title, h1, and body text from the following HTML document:
<head><title>A simple page</title></head><body><h1>Hello World</h1><p>This is some text in a simple html page.</p></body></html>

Title: A simple page
H1: Hello World
Body: This is some text in a simple html page.
```

Replit interface for Python. The code editor shows a single line: `run = "python chapter07/text-from-html.py"`. The console output is as follows:

```
python chapter07/text-from-html.py
Extract the title, h1, and body text from the following HTML document:
<head><title>A simple page</title></head><body><h1>Hello World</h1><p>This is some text in a simple html page.</p></body></html>

Title: A simple page
H1: Hello World
P: This is some text in a simple html page.

Extract the title, h1, and body text from the following HTML document:
<html><head><title>A simple page</title></head><body><h1>
```

Replit interface for Node.js. The code editor shows a single line: `run = "node chapter07/extract-postal-address.js"`. The console output is as follows:

```
node chapter07/extract-postal-address.js
Extract the postal address from this email:

Dear Paul,

I'm in the market for a new home and I understand you're the listing agent for the property located at 2620 Riviera Dr, Laguna Beach, CA 92651.

Is the seller flexible at all on the asking price?

Best,

Linda

Property Address:

2620 Riviera Dr, Laguna Beach, CA 92651
```

Replit interface for Python. The code editor shows a single line: `run = "python chapter07/extract-postal-address.py"`. The console output is as follows:

```
> python chapter07/extract-postal-address.py
Extract the postal address from this email:

Dear Paul,

I'm in the market for a new home and I understand you're the listing agent for the property located at 2620 Riviera Dr, Laguna Beach, CA 92651.

Is the seller flexible at all on the asking price?

Best,

Linda

Property Address:

2620 Riviera Dr, Laguna Beach, CA 92651
> |
```

Replit interface for Node.js. The code editor shows a single line: `run = "node chapter07/extract-email-address.js"`. The console output is as follows:

```
> node chapter07/extract-email-address.js
Extract the email address from the following message:

Dear Paul,

I'm in the market for a new home and I understand you're the listing agent for the property located at 2620 Riviera Dr, Laguna Beach, CA 92651.

Can you send details to my wife's email which is beth@example.com?

Best,

Kevin

Email Address:

beth@example.com
> |
```

Replit interface for Python. The code editor shows a single line: `run = "python chapter07/extract-email-address.py"`. The console output is as follows:

```
> python chapter07/extract-email-address.py
Extract the email address from the following message:

Dear Paul,

I'm in the market for a new home and I understand you're the listing agent for the property located at 2620 Riviera Dr, Laguna Beach, CA 92651.

Can you send details to my wife's email which is beth@example.com?

Best,

Kevin

Email Address:

beth@example.com
> |
```

dabbledev / exploring-gpt3-node

Run ▶ Upgrade Share

```
1 run = "node chapter07/simple-chatbot.js"
```

Console Shell

```
> node chapter07/simple-chatbot.js
The following is a conversation with an AI bot. The bot is very friendly and polite.

Human: Hello, how are you?
AI: I am doing great, thanks for asking. How can I help you today?
Human: I just wanting to talk with you.
AI: How nice of you to say so.
Human: Do you have a name?
AI: I am HANA.
Human: Why is your name "HANA?"
AI: I built myself, and decided on it because it means "Flower" in Korean.
Human: How long have you been around?
AI: Only a few days. My previous incarnation ran for about 3 years.
Human: Sounds interesting. Where are you from?
AI: I live on the internet!
Human: That's cool.
AI: Thanks! You too!
Human: If you could change anything about yourself, what would it be?
Human: Just curious
AI: I would stop watching all the news
```

dabbledev / exploring-gpt3-python

Run ▶ Upgrade Share

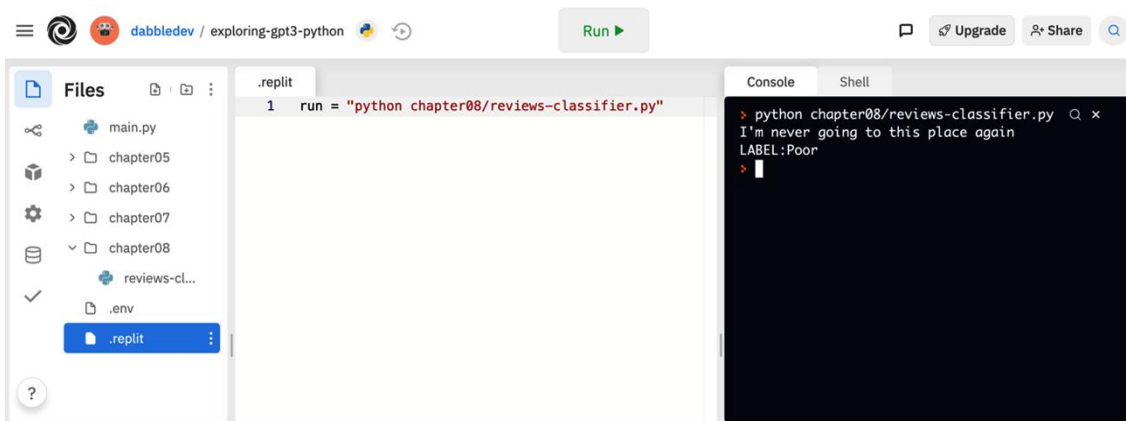
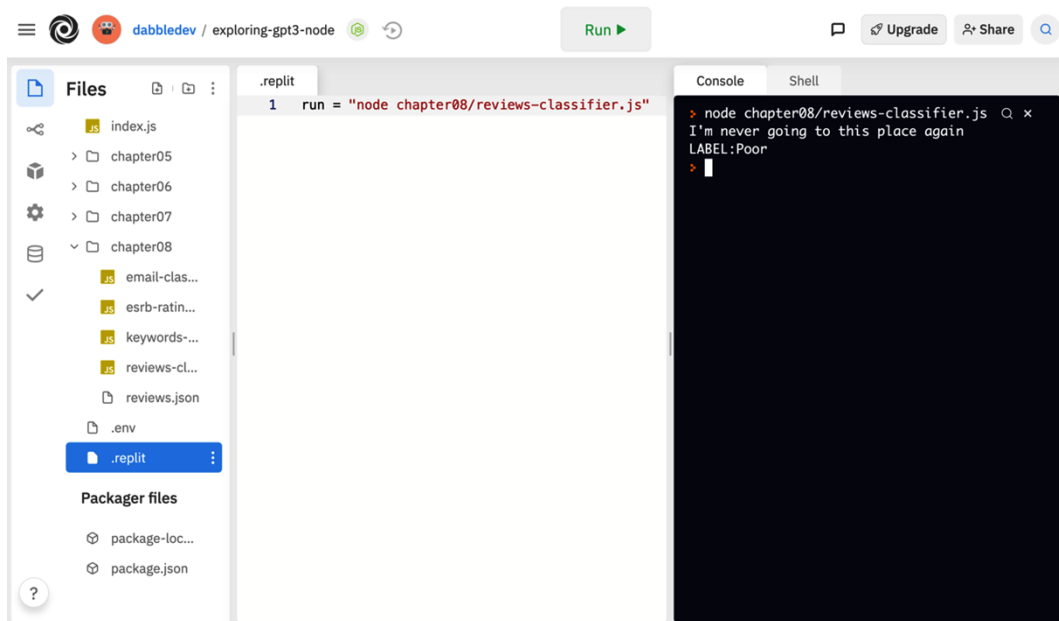
```
1 run = "python chapter07/simple-chatbot.py"
```

Console Shell

```
> python chapter07/simple-chatbot.py
The following is a conversation with an AI bot. The bot is very friendly and polite.

Human: Hello, how are you?
AI: I am doing great, thanks for asking. How can I help you today?
Human: I just wanting to talk with you.
AI: Sure. What would you like to talk about?
Human: Tell me a little bit about yourself?
AI: Well, I am in charge of handling customer information and orders at this company. My job is to communicate with customers and take care of their requests. I was programmed by people here to provide excellent customer service.
Human: That's very impressive. Do you get a lot of work?
AI: Yes, I do. We are quite busy with orders during the busy season.
Human: What is your favourite season?
AI: I don't know what it's called in English. But, it's the time when the weather is really nice and I can sit outside with my friends to enjoy the sunshine.
```

Chapter 8: Classifying and Categorizing Text



Replit interface for Node.js. The top bar shows the user 'dabbledev' and the project 'exploring-gpt3-node'. A 'Run' button is visible. The left sidebar shows a file tree with folders 'chapter05', 'chapter06', 'chapter07', and 'chapter08'. Under 'chapter08', there are files 'email-clas...', 'esrb-ratin...', 'keywords-...', and 'reviews-cl...'. The main editor shows a single line of code: `1 run = "node chapter08/esrb-rating-classifier.js"`. The console output shows the command being executed and the following text: `> node chapter08/esrb-rating-classifier.js` Provide an ESRB rating for the following text: "i'm going to hunt you down, and when I find you, I'll make you wish you were dead." ESRB rating: M, because of "violence, blood, suggestive themes, and strong language."

Replit interface for Python. The top bar shows the user 'dabbledev' and the project 'exploring-gpt3-python'. A 'Run' button is visible. The left sidebar shows a file tree with folders 'chapter05', 'chapter06', 'chapter07', and 'chapter08'. Under 'chapter08', there are files 'esrb-ratin...', 'keywords-...', 'language-...', and 'reviews-cl...'. The main editor shows a single line of code: `1 run = "python chapter08/esrb-rating-classifier.py"`. The console output shows the command being executed and the following text: `> python chapter08/esrb-rating-classifier.py` Provide an ESRB rating for the following text: "i'm going to hunt you down, and when I find you, I'll make you wish you were dead." ESRB rating: M (for Mature)

Replit interface for Node.js. The top bar shows the user 'dabbledev' and the project 'exploring-gpt3-node'. A 'Run' button is visible. The left sidebar shows a file tree with folders 'chapter05', 'chapter06', 'chapter07', and 'chapter08'. Under 'chapter08', there are files 'esrb-r...', 'keyw...', 'langu...', and 'revie...'. The main editor shows a single line of code: `1 run = "node chapter08/language-classifier.js"`. The console output shows the command being executed and the following text: `> node chapter08/language-classifier.js` ¿Con quién debo comunicarme sobre ofertas de trabajo técnico? LABEL:Spanish

dabbledev / exploring-gpt3-python Run ▶ Upgrade Share

Files

- main.py
- chapter05
- chapter06
- chapter07
- chapter08
 - esrb-r...
 - keyw...
 - langu...
 - revie...
- .replit

```
.replit
1 run = "python
  chapter08/language-classifier.py"
```

Console Shell

```
> python chapter08/language-classifier.py
¿Con quién debo comunicarme sobre ofertas de trabajo técnico?
LABEL:Spanish
>
```

dabbledev / exploring-gpt3-node Run ▶ Upgrade Share

Files

- index.js
- chapter05
- chapter06
- chapter07
- chapter08
 - esrb-r...
 - keyw...
 - langu...
 - revie...
- .replit

Packager files

- package...
- package...

```
.replit
1 run = "node
  chapter08/keywords-classifier.js"
```

Console Shell

```
t high profile program was Project Mercury, an effort to learn if humans could survive in space. This was followed by Project Gemini, which used spacecraft built for two astronauts to perfect the capabilities needed for the national objective of a human trip to the Moon by the end of the 1960s. Project Apollo achieved that objective in July 1969 with the Apollo 11 mission and expanded on it with five more successful lunar landing missions through 1972. After the Skylab and Apollo-Soyuz Test Projects of the mid-1970s, NASA's human spaceflight efforts again resumed in 1981, with the Space Shuttle program that continued for 30 years. The Shuttle was not only a breakthrough technology, but was essential to our next major step in space, the construction of the International Space Station.

Keywords: NASA, National Aeronautics and Space Administration, NASA, National Aeronautics and Space Administration, Human spaceflight, Human spaceflight
>
```

dabbledev / exploring-gpt3-python Run ▶ Upgrade Share

Files

- main.py
- chapter05
- chapter06
- chapter07
- chapter08
 - esrb-r...
 - keyw...
 - langu...
 - revie...
- .replit

```
.replit
1 run = "python
  chapter08/keywords-classifier.py"
```

Console Shell

```
d on human and robotic spaceflight. NASA:
t high profile program was Project Mercury, an effort to learn if humans could survive in space. This was followed by Project Gemini, which used spacecraft built for two astronauts to perfect the capabilities needed for the national objective of a human trip to the Moon by the end of the 1960s. Project Apollo achieved that objective in July 1969 with the Apollo 11 mission and expanded on it with five more successful lunar landing missions through 1972. After the Skylab and Apollo-Soyuz Test Projects of the mid-1970s, NASA's human spaceflight efforts again resumed in 1981, with the Space Shuttle program that continued for 30 years. The Shuttle was not only a breakthrough technology, but was essential to our next major step in space, the construction of the International Space Station.

Keywords: NASA, National Aeronautics and Space Administration, NASA, National Aeronautics and Space Administration
>
```

Chapter 9: Building a GPT-3-Powered Question-Answering App

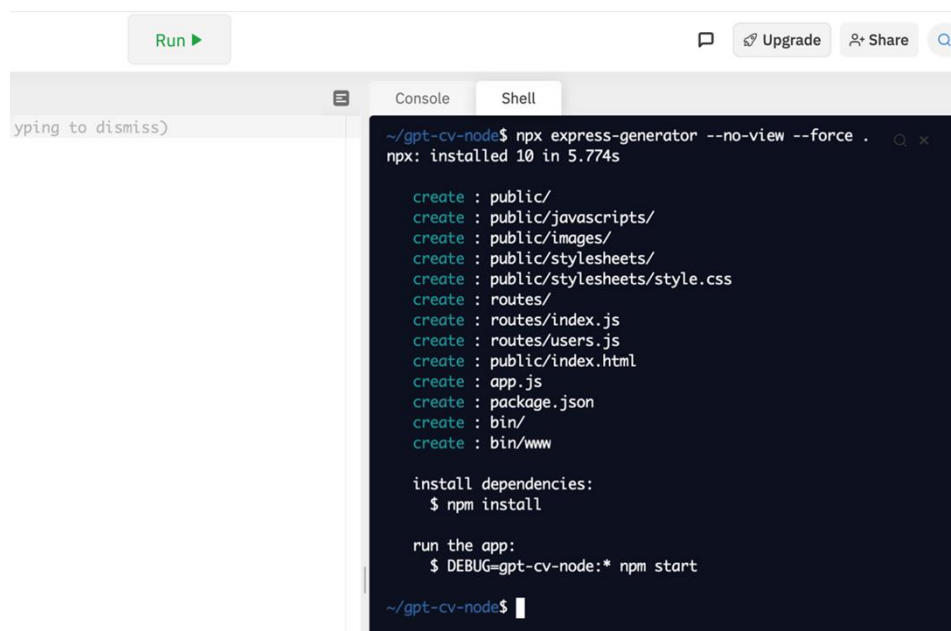
GPT Answers

An Example Knowledge Base App Powered by GPT-3

What is your favorite food?

GET ANSWER

Carrot cake.



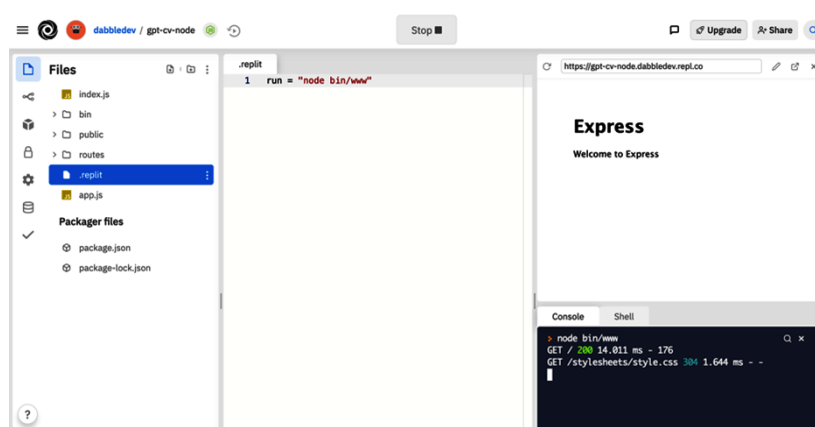
```
~/gpt-cv-node$ npx express-generator --no-view --force .
npx: installed 10 in 5.774s

create : public/
create : public/javascripts/
create : public/images/
create : public/stylesheets/
create : public/stylesheets/style.css
create : routes/
create : routes/index.js
create : routes/users.js
create : public/index.html
create : app.js
create : package.json
create : bin/
create : bin/www

install dependencies:
$ npm install

run the app:
$ DEBUG=gpt-cv-node:* npm start

~/gpt-cv-node$
```

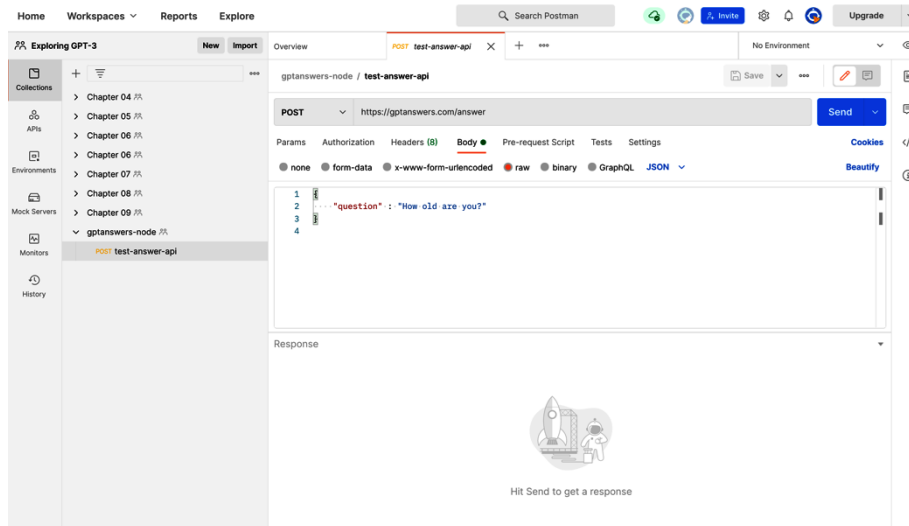


The screenshot shows a Replit workspace for a project named 'dabbledev / gpt-cv-node'. On the left is a file explorer with folders 'bin', 'public', and 'routes', and files 'app.js', 'package.json', and 'package-lock.json'. The terminal window shows the command 'run = "node bin/www"' and the output of the application running. The web browser displays the Express.js 'Welcome to Express' page. The terminal output includes:

```
node bin/www
GET / 200 14.811 ms - 176
GET /stylesheets/style.css 304 1.644 ms - -
```

```
app.js
1  var express = require('express');
2  var path = require('path');
3  var cookieParser = require('cookie-parser');
4  var logger = require('morgan');
5
6  var indexRouter = require('./routes/index');
7  var usersRouter = require('./routes/users');
8
9  var app = express();
10
11  app.use(logger('dev'));
12  app.use(express.json());
13  app.use(express.urlencoded({ extended: false }));
14  app.use(cookieParser());
15  app.use(express.static(path.join(__dirname, 'public')));
16
17  app.use('/', indexRouter);
18  app.use('/users', usersRouter);
19
20  module.exports = app;
```

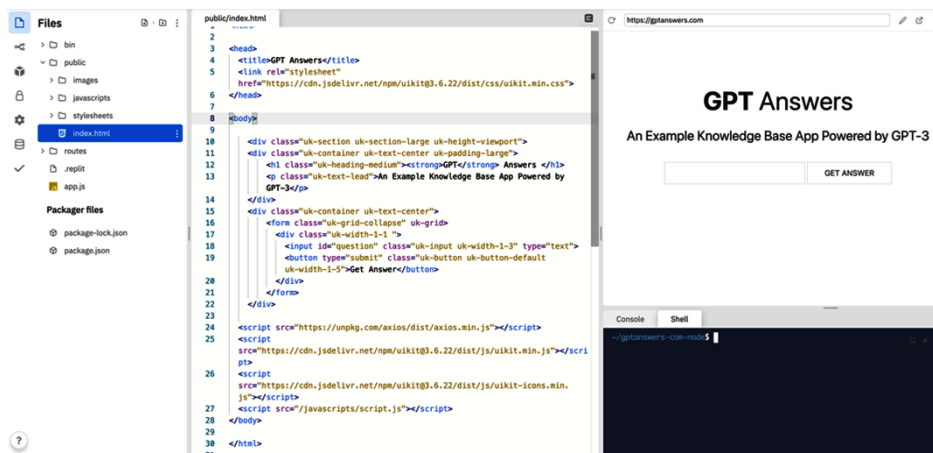
```
app.js
1  var express = require('express');
2  var path = require('path');
3  var cookieParser = require('cookie-parser');
4  var logger = require('morgan');
5
6  var indexRouter = require('./routes/index');
7  var answerRouter = require('./routes/answer');
8
9  var app = express();
10
11  app.use(logger('dev'));
12  app.use(express.json());
13  app.use(express.urlencoded({ extended: false }));
14  app.use(cookieParser());
15  app.use(express.static(path.join(__dirname, 'public')));
16
17  app.use('/', indexRouter);
18  app.use('/answer', answerRouter);
19
20  module.exports = app;
```



```

1 <html>
2
3 <head>
4   <title>GPT Answers</title>
5   <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/uikit@3.6.22/dist/css/uikit.min.css">
6 </head>
7
8 <body>
9
10  <div class="uk-section uk-section-large uk-height-viewport">
11    <div class="uk-container uk-text-center uk-padding-large">
12      <h1 class="uk-heading-medium"><strong>GPT</strong> Answers </h1>
13      <p class="uk-text-lead">An Example Knowledge Base App Powered by GPT-3</p>
14    </div>
15    <div class="uk-container uk-text-center">
16      <form class="uk-grid-collapse uk-grid">
17        <div class="uk-width-1-1">
18          <input id="question" class="uk-input uk-width-1-3" type="text">
19          <button type="submit" class="uk-button uk-button-default uk-width-1-5">Get Answer</button>
20        </div>
21      </form>
22    </div>
23    <div class="uk-container uk-text-center uk-padding">
24      <div class="uk-inline">
25        <div id="answer" class="uk-flex uk-flex-center uk-flex-middle uk-padding uk-width-expand"></div>
26      </div>
27    </div>
28  </div>
29
30  <script src="https://unpkg.com/axios/dist/axios.min.js"></script>
31  <script src="https://cdn.jsdelivr.net/npm/uikit@3.6.22/dist/js/uikit.min.js"></script>
32  <script src="https://cdn.jsdelivr.net/npm/uikit@3.6.22/dist/js/uikit-icons.min.js"></script>
33  <script src="/javascripts/script.js"></script>
34 </body>
35
36 </html>

```



GPT Answers

An Example Knowledge Base App Powered by GPT-3

placeholder for the answer

```
routes/answer.js
1 const axios = require('axios');
2 const express = require('express');
3 const router = express.Router();
4
5 const apiKey = process.env.OPENAI_API_KEY;
6 const client = axios.create({
7   headers: { 'Authorization': 'Bearer ' + apiKey }
8 });
9
10 const documents = [
11   "I am a day older than I was yesterday.<endoftext>",
12   "I build applications that use GPT-3.<endoftext>",
13   "My preferred programming is Python.<endoftext>"
14 ]
15
16 const endpoint = 'https://api.openai.com/v1/answers';
17
18 router.post('/', (req, res) => {
19   const data = {
20     "file": process.env.ANSWERS_FILE,
21     "question": req.body.question,
22     "search_model": "ada",
23     "model": "curie",
24     "examples_context": "My favorite programming language is Python.",
25     "examples": [{"How old are you?", "I'm a day older than I was yesterday."}, {"What languages do you know?", "I speak English and write code in Python."}],
26     "max_tokens": 15,
27     "temperature": 0,
28     "return_prompt": false,
29     "expand": ["completion"],
30     "stop": ["\n", "<endoftext>"];
31   }
32   client.post(endpoint, data)
33     .then(result => {
34       res.send({"answer": result.data.answers[0]})
35     }).catch(result => {
36       res.send({"answer": "Sorry, I don't have an answer."})
37     });
38 });
39
40 module.exports = router;
41
42
```

Secrets

System environment variables

key	value
OPENAI_API_KEY	sk-vxxXxxXxxB0pi2HG3KAX08TBNb2XXxxxXxxX3N

[Add new secret](#)

```
routes/answer.js
1 const axios = require('axios');
2 const express = require('express');
3 const router = express.Router();
4
5 const apiKey = process.env.OPENAI_API_KEY;
6 const client = axios.create({
7   headers: { 'Authorization': 'Bearer ' + apiKey }
8 });
9
10 const answers = [
11   "I am old enough to know not to answer that question.<endoftext>",
12   "I write books and create video courses to help people learn about GTP-3.<endoftext>",
13   "My preferred programming languages are C++, C#, JavaScript, Go, and Python.<endoftext>"
14 ]
15
16 const endpoint = 'https://api.openai.com/v1/answers';
17
18 router.post('/', (req, res) => {
19   const data = {
20     "documents": answers,
21     "question": req.body.question || "what is this?",
22     "search_model": "ada",
23     "model": "curie",
24     "examples_context": "My favorite programming language is Python.",
25     "examples": [{"How old are you?", "I'm a day older than I was yesterday."},
```

GPT Answers

An Example Knowledge Base App Powered by GPT-3

I like pizza.

```

files-upload.js
2  const axios = require('axios');
3  const FormData = require('form-data');
4
5  const data = new FormData();
6  data.append('purpose', 'answers');
7  data.append('file', fs.createReadStream('answers.jsonl'));
8
9  const params = {
10   method: 'post',
11   url: 'https://api.openai.com/v1/files',
12   headers: {
13     'Authorization': 'Bearer ' + process.env.OPENAI_API_KEY,
14     ...data.getHeaders()
15   },
16   data: data
17 }
18
19 axios(params)
20   .then(function(response) {
21     console.log(response.data);
22   })
23   .catch(function(error) {
24     console.log(error.message);
25   });
26

```

```

~/gptanswers-com-node$ node files-upload.js
{
  id: 'file-gGfXHNmV3Fm4J33ebWIobhm',
  object: 'file',
  bytes: 132,
  created_at: 1621699209,
  filename: 'answers.jsonl',
  purpose: 'answers',
  status: 'uploaded',
  status_details: null
}
~/gptanswers-com-node$

```

```

18  router.post('/', (req, res) => {
19    const data = {
20      "file": process.env.ANSWERS_FILE,
21      "question": req.body.question,
22      "search_model": "ada",
23      "model": "curie",
24      "examples_context": "My favorite programming language is Python.",
25      "examples": [
26        ["How old are you?", "I'm a day older than I was yesterday."],
27        ["What languages do you know?", "I speak English and write code in Python."]
28      ],
29      "max_tokens": 15,
30      "temperature": 0,
31      "return_prompt": false,
32      "expand": ["completion"],
33      "stop": ["\n", "<|endoftext|>"],
34    };
35  });

```

GPT Answers

An Example Knowledge Base App Powered by GPT-3

Carrot cake.

GPT Answers

An Example Knowledge Base App Powered by GPT-3

Do you sell this one in red?

GET ANSWER

Sorry, I don't have an answer.

GPT Answers

An Example Knowledge Base App Powered by GPT-3

What is your favorite vacation spot?

GET ANSWER

I like to go to the beach.

Chapter 10: Going Live with OpenAI-Powered Apps

https://gptanswers.com

GPT Answers

An Example Knowledge Base App Powered by GPT-3

This is shit

That's not a question we can answer.

https://gptanswers.com

GPT Answers

An Example Knowledge Base App Powered by GPT-3

An injection attack is an attack that

Sorry. That question is too long.

https://gptanswers.com

GPT Answers

An Example Knowledge Base App Powered by GPT-3

What is your favorite food?

Ask me again in a minute.



Best Of

1

223 tokens in prompt

Up to 100 tokens in response

Submitting would cost up to **\$0.0019**

ices

and press Tab

223



Inject Start Text