Package 'gemini.R'

May 8, 2026

Title Interface for 'Google Gemini' API

Version 0.17.2

Maintainer Jinhwan Kim <hwanistic@gmail.com>

Description Provides a comprehensive interface for Google Gemini API,
enabling users to access and utilize Gemini Large Language Model (LLM) functionalities directly from R.
This package facilitates seamless integration with Google Gemini, allowing for advanced language processing,
text generation, and other AI-driven capabilities within the R environment.
For more information, please visit <https://ai.google.dev/docs/gemini_api_overview>.

License MIT + file LICENSE

Depends R (>= 4.1.0)

URL https://github.com/jhk0530/gemini.R

BugReports https://github.com/jhk0530/gemini.R/issues

Encoding UTF-8

Imports base64enc, cli, httr2, jsonlite, knitr, rstudioapi, tools

RoxygenNote 7.3.2

Suggests quarto, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

Config/Needs/website rmarkdown, quarto

NeedsCompilation no

Author Jinhwan Kim [aut, cre, cph] (ORCID:
<https://orcid.org/0009-0009-3217-2417>),
Maciej Nasinski [ctb]

Repository CRAN

Date/Publication 2025-09-01 10:30:02 UTC

Contents

`countTokens` *Count Tokens for Gemini Content (Including Images)*

Description

Calculates the token count for a given content, including text and image data, using the Vertex AI Gemini API.

Usage

```
countTokens(
  jsonkey = NULL,
  model_id = NULL,
  content = NULL,
  region = "us-central1"
)
```

Arguments

jsonkey	A path to JSON file containing the service account key from Vertex AI.
model_id	The ID of the Gemini model.
content	The content (text, image, or list of text/image parts) for which to count tokens. • For text, provide a string. • For images, provide a list with data (base64 encoded image) and mimeType (e.g., "image/png", "image/jpeg"). • For multiple content parts, provide a list where each element is either a text string or an image list.
region	The Google Cloud region where your Vertex AI resources are located (default is "us-central1"). See https://cloud.google.com/vertex-ai/docs/regions for available regions.

Value

A numeric value representing the token count of the content.

Examples

```
## Not run:
library(gemini.R)

# For text content
key_file <- "YOURAPIKEY.json"
model <- "2.0-flash"
token_count_text <- countTokens(
  jsonkey = key_file,
  model_id = model,
  content = "Hello, world!"
)
print(token_count_text)

# For image content (assuming 'image.jpg' is in your working directory)
image_data <- base64enc::base64encode("image.jpg")
image_content <- list(data = image_data, mimeType = "image/jpeg")
token_count_image <- countTokens(
  jsonkey = key_file,
  model_id = model,
  content = image_content
)
print(token_count_image)

# For multiple content parts (text and image)
content_parts <- list(
  list(text = "This is the first part."),
  list(data = image_data, mimeType = "image/jpeg"),
  list(text = "This is the last part")
)
token_count_parts <- countTokens(
```

```
  jsonkey = key_file,
  model_id = model,
  content = content_parts
)
print(token_count_parts)

## End(Not run)
```

---

gemini *Generate text from text with Gemini*

Description

Generate text from text with Gemini

Usage

```
gemini(
  prompt,
  model = "2.0-flash",
  temperature = 1,
  maxOutputTokens = 8192,
  topK = 40,
  topP = 0.95,
  seed = 1234,
  timeout = 60
)
```

Arguments

prompt	The prompt to generate text from
model	The model to use. Default is '2.0-flash'. see https://ai.google.dev/gemini-api/docs/models/gemini
temperature	The temperature to use. Default is 1 value should be between 0 and 2 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
maxOutputTokens	The maximum number of tokens to generate. Default is 8192 and 100 tokens correspond to roughly 60-80 words.
topK	The top-k value to use. Default is 40 value should be between 0 and 100 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
topP	The top-p value to use. Default is 0.95 value should be between 0 and 1 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
seed	The seed to use. Default is 1234 value should be integer see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
timeout	Request timeout in seconds. Default is 60.

Value

Generated text or image

See Also

https://ai.google.dev/docs/gemini_api_overview#text_input

Examples

```
## Not run:
library(gemini.R)
setAPI("YOUR_API_KEY")
gemini("Explain dplyr's mutate function")

## End(Not run)
```

`gemini.vertex` *Generate text from text with Gemini Vertex API*

Description

Generate text from text with Gemini Vertex API

Usage

```
gemini.vertex(
  prompt = NULL,
  tokens = NULL,
  temperature = 1,
  maxOutputTokens = 8192,
  topK = 40,
  topP = 0.95,
  seed = 1234,
  timeout = 60,
  labels = NULL
)
```

Arguments

`prompt`	A character string containing the prompt for the Gemini model.
`tokens`	A list containing the API URL and key from token.vertex() function.
`temperature`	The temperature to use. Default is 1 value should be between 0 and 2 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
`maxOutputTokens`	The maximum number of tokens to generate. Default is 8192 and 100 tokens correspond to roughly 60-80 words.

topK	The top-k value to use. Default is 40 value should be between 0 and 100 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
topP	The top-p value to use. Default is 0.95 value should be between 0 and 1 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
seed	The seed to use. Default is 1234 value should be integer see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
timeout	Request timeout in seconds. Default is 60.
labels	(Optional) A named list for custom metadata labels. Example: `list(team = "research", env = "test")`.

Value

A character string containing the generated text.

See Also

https://ai.google.dev/docs/gemini_api_overview#text_input

Examples

```
## Not run:
# token should be created before this. using the token.vertex() function
prompt <- "What is sachins Jersey number?"
gemini.vertex(prompt, tokens)
gemini.vertex(prompt, tokens, labels = list(team = "research", env = "test"))

## End(Not run)
```

---

gemini_audio *Analyze audio using Gemini*

Description

This function sends audio to the Gemini API and returns a text description.

Usage

```
gemini_audio(
  audio = NULL,
  prompt = "Describe this audio",
  model = "2.0-flash",
  temperature = 1,
  maxOutputTokens = 8192,
  topK = 40,
  topP = 0.95,
  seed = 1234
)
```

Arguments

audio	Path to the audio file (default: uses a sample file). Must be an MP3.
prompt	A string describing what to do with the audio.
model	The model to use. Options are "2.0-flash", "2.0-flash-lite", "2.5-pro-exp-03-25". Default is '2.0-flash' see https://ai.google.dev/gemini-api/docs/models/gemini
temperature	The temperature to use. Default is 1 value should be between 0 and 2 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
maxOutputTokens	The maximum number of tokens to generate. Default is 8192 and 100 tokens correspond to roughly 60-80 words.
topK	The top-k value to use. Default is 40 value should be between 0 and 100 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
topP	The top-p value to use. Default is 0.95 value should be between 0 and 1 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
seed	The seed to use. Default is 1234 value should be integer see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters

Details

The API key is now sent via the HTTP header `x-goog-api-key` instead of as a URL query parameter.

Value

A character vector containing the Gemini API's response.

Examples

```
## Not run:
library(gemini.R)
setAPI("YOUR_API_KEY")
gemini_audio(audio = "YOUR_AUDIO_FILE")

## End(Not run)
```

---

gemini_audio.vertex *Analyze Audio using Gemini Vertex API*

Description

This function sends audio to the Gemini API and returns a text description.

Usage

```
gemini_audio.vertex(
  audio = NULL,
  prompt = "Describe this audio",
  tokens = NULL,
  temperature = 1,
  maxOutputTokens = 8192,
  topK = 40,
  topP = 0.95,
  seed = 1234
)
```

Arguments

audio	Path to the audio file (character string). only supports "mp3".
prompt	A prompt to guide the Gemini API's analysis (character string, defaults to "Describe this audio").
tokens	A list containing the API URL and key from token.vertex() function.
temperature	The temperature to use. Default is 1 value should be between 0 and 2 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
maxOutputTokens	The maximum number of tokens to generate. Default is 8192 and 100 tokens correspond to roughly 60-80 words.
topK	The top-k value to use. Default is 40 value should be between 0 and 100 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
topP	The top-p value to use. Default is 0.95 value should be between 0 and 1 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
seed	The seed to use. Default is 1234 value should be integer see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters

Value

A character vector containing the Gemini API's description of the audio.

---

gemini_chat *Multi-turn conversations (chat)*

Description

Generate text from text with Gemini

Usage

```
gemini_chat(
  prompt,
  history = list(),
  model = "2.0-flash",
  temperature = 1,
  maxOutputTokens = 8192,
  topK = 40,
  topP = 0.95,
  seed = 1234
)
```

Arguments

prompt	The prompt to generate text from
history	history object to keep track of the conversation
model	The model to use. Options are "2.0-flash", "2.0-flash-lite", "2.5-pro-exp-03-25". Default is '2.0-flash' see https://ai.google.dev/gemini-api/docs/models/gemini
temperature	The temperature to use. Default is 1 value should be between 0 and 2 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
maxOutputTokens	The maximum number of tokens to generate. Default is 8192 and 100 tokens correspond to roughly 60-80 words.
topK	The top-k value to use. Default is 40 value should be between 0 and 100 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
topP	The top-p value to use. Default is 0.95 value should be between 0 and 1 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
seed	The seed to use. Default is 1234 value should be integer see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters

Value

Generated text

See Also

https://ai.google.dev/docs/gemini_api_overview#chat

Examples

```
## Not run:
library(gemini.R)
setAPI("YOUR_API_KEY")

chats <- gemini_chat("Pretend you're a snowman and stay in character for each")
print(chats$outputs)
```

```
chats <- gemini_chat("What's your favorite season of the year?", chats$history)
print(chats$outputs)

chats <- gemini_chat("How do you think about summer?", chats$history)
print(chats$outputs)

## End(Not run)
```

`gemini_docs` *Summarize or analyze one or more local documents using Gemini API*

Description

Summarize, compare, or analyze the content of one or more local documents (PDF, TXT, HTML, etc.) using the Gemini API.

Usage

```
gemini_docs(
  pdf_path,
  prompt,
  type = "PDF",
  model = "2.5-flash",
  api_key = Sys.getenv("GEMINI_API_KEY"),
  large = FALSE,
  local = FALSE
)
```

Arguments

`pdf_path`	Path(s) to the local file(s). Can be a character vector.
`prompt`	The prompt to send to Gemini (e.g., "Summarize these documents").
`type`	File type. One of "PDF", "JavaScript", "Python", "TXT", "HTML", "CSS", "Markdown", "CSV", "XML", "RTF". Default is "PDF".
`model`	The model to use. Default is '2.5-flash'. see https://ai.google.dev/gemini-api/docs/models/gemini
`api_key`	Gemini API key. Defaults to `Sys.getenv("GEMINI_API_KEY")`. The API key is sent via the HTTP header `x-goog-api-key`.
`large`	Logical. If `TRUE`, use the file upload API for large files (only one file supported). Default is `FALSE`.
`local`	Logical. If `TRUE`, treat `pdf_path` as a local file path. If `FALSE`, download from URL. Default is `FALSE`.

Details

This function encodes one or more local files, sends them along with a prompt to the Gemini API, and returns the generated summary or response.

Value

The summary or response text from Gemini.

See Also

https://ai.google.dev/gemini-api/docs/document-processing?lang=rest

Examples

```
## Not run:
gemini_docs(
  pdf_path = c("doc1.pdf", "doc2.pdf"),
  prompt = "Compare these documents",
  type = "PDF",
  model = "2.5-flash"
)

## End(Not run)
```

---

gemini_docs.vertex *Summarize or analyze documents using Vertex AI Gemini*

Description

Summarize, compare, or analyze the content of one or more documents (PDF, TXT, HTML, etc.) using Vertex AI Gemini.

Usage

```
gemini_docs.vertex(
  file_uri,
  prompt,
  mime_type = "application/pdf",
  tokens = NULL,
  temperature = 1,
  maxOutputTokens = 8192,
  topK = 40,
  topP = 0.95,
  seed = 1234
)
```

Arguments

file_uri	The URI(s) or URL(s) of the file(s) to include in the prompt. Accepts Cloud Storage URI (gs://...), HTTP(S) URL, or YouTube video URL.
prompt	The text instructions to include in the prompt.

mime_type	The media type of the file (e.g., "application/pdf", "text/plain").
tokens	A list containing the API URL and key from token.vertex() function.
temperature	The temperature to use. Default is 1.
maxOutputTokens	The maximum number of tokens to generate. Default is 8192.
topK	The top-k value to use. Default is 40.
topP	The top-p value to use. Default is 0.95.
seed	The seed to use. Default is 1234.

Value

The summary or response text from Gemini Vertex.

See Also

https://cloud.google.com/vertex-ai/docs/generative-ai/multimodal/send-request-document

Examples

```
## Not run:
tokens <- token.vertex()
gemini_docs.vertex(
  file_uri = "gs://cloud-samples-data/generative-ai/pdf/2403.05530.pdf",
  prompt = "Summarize this document.",
  mime_type = "application/pdf",
  tokens = tokens
)

## End(Not run)
```

---

gemini_garden *Interact with Vertex AI Model Garden*

Description

This function sends a PDF file to the Vertex AI Model Garden (Mistral model) for processing, such as OCR. The PDF is encoded as base64 and sent to the rawPredict endpoint. The function is designed for future extension to support other document types and tasks.

Usage

```
gemini_garden(token, project_id, pdf_path)
```

Arguments

token	Token object (e.g., from token.vertex()) containing the access token, region, and model_id.
project_id	Google Cloud project ID.
pdf_path	Path to the PDF file to be processed.

Details

The PDF file is read and encoded as base64, then sent to the Vertex AI rawPredict endpoint for processing using a Mistral model. This function is structured for future extension to support other document types and model tasks available in Vertex AI Model Garden.

For more information about available models, endpoints, and supported tasks, see Vertex AI Model Garden documentation.

Value

A parsed list containing the results from the Vertex AI API (e.g., OCR results).

See Also

https://cloud.google.com/vertex-ai/generative-ai/docs/model-garden/explore-models

Examples

```
## Not run:
# Issue a token using token.vertex() first
my_token <- token.vertex(
  jsonkey = "your-service-account.json",
  region = "us-central1",
  model_id = "mistral-ocr-2505"
)
result <- gemini_garden(
  token = my_token,
  project_id = "your-project-id",
  pdf_path = "sample.pdf"
)
print(result)

## End(Not run)
```

---

gemini_image *Generate text from text and image with Gemini*

Description

Generate text from text and image with Gemini

Usage

```
gemini_image(
  image = NULL,
  prompt = "Explain this image",
  model = "2.0-flash",
  temperature = 1,
  maxOutputTokens = 8192,
  topK = 40,
  topP = 0.95,
  seed = 1234,
  type = "png"
)
```

Arguments

image	The image to generate text
prompt	The prompt to generate text, Default is "Explain this image"
model	The model to use. Options are "2.0-flash", "2.0-flash-lite", "2.5-pro-exp-03-25". Default is '2.0-flash' see https://ai.google.dev/gemini-api/docs/models/gemini
temperature	The temperature to use. Default is 1 value should be between 0 and 2 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
maxOutputTokens	The maximum number of tokens to generate. Default is 8192 and 100 tokens correspond to roughly 60-80 words.
topK	The top-k value to use. Default is 40 value should be between 0 and 100 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
topP	The top-p value to use. Default is 0.95 value should be between 0 and 1 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
seed	The seed to use. Default is 1234 value should be integer see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
type	The type of image. Options are 'png', 'jpeg', 'webp', 'heic', 'heif'. Default is 'png'

Details

The API key is now sent via the `x-goog-api-key` HTTP header instead of as a URL query parameter.

Value

Generated text

See Also

https://ai.google.dev/docs/gemini_api_overview#text_image_input

Examples

```
## Not run:
library(gemini.R)
setAPI("YOUR_API_KEY")
gemini_image(image = system.file("docs/reference/figures/image.png", package = "gemini.R"))

## End(Not run)
```

`gemini_image.vertex` *Generate text from text and image with Gemini Vertex API*

Description

Generate text from text and image with Gemini Vertex API

Usage

```
gemini_image.vertex(
  image = NULL,
  prompt = "Explain this image",
  type = "png",
  tokens = NULL,
  temperature = 1,
  maxOutputTokens = 8192,
  topK = 40,
  topP = 0.95,
  seed = 1234
)
```

Arguments

`image`	The image to generate text
`prompt`	A character string specifying the prompt to use with the image. Defaults to "Explain this image".
`type`	A character string specifying the image type ("png", "jpeg", "webp", "heic", "heif"). Defaults to "png".
`tokens`	A list containing the API URL and key from token.vertex() function.
`temperature`	The temperature to use. Default is 1 value should be between 0 and 2 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
`maxOutputTokens`	The maximum number of tokens to generate. Default is 8192 and 100 tokens correspond to roughly 60-80 words.
`topK`	The top-k value to use. Default is 40 value should be between 0 and 100 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters

topP	The top-p value to use. Default is 0.95 value should be between 0 and 1 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
seed	The seed to use. Default is 1234 value should be integer see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters

Value

A character vector containing Gemini's description of the image.

`gemini_narrative` *Generate narrative description for an input using Gemini*

Description

Generate a narrative description for a given input (e.g., table, figure) by converting it to a suitable format and sending it to Gemini.

Usage

```
gemini_narrative(input, type = "table", prompt = NULL, ...)
```

Arguments

input	Input object. For type = "table", a data.frame should be provided.
type	Type of input. Default is "table". (e.g., "table", "figure")
prompt	Optional prompt string to guide the narrative generation. If NULL, a default prompt is used.
...	Additional arguments passed to gemini()

Value

Narrative description generated by Gemini

See Also

gemini

Examples

```
## Not run:
# Example data.frame
table_data <- data.frame(
  sort = c(1, 1, 2, 2, 2, 3, 3, 3, 4, 4),
  category = c(
    "Gender", "Gender", "Age Group", "Age Group", "Age Group",
    "Race", "Race", "Race", "Ethnicity", "Ethnicity"
  ),
  type = c(
```

```
    "F", "M", "<65", ">80", "65-80",
    "AMERICAN INDIAN OR ALASKA NATIVE", "BLACK OR AFRICAN AMERICAN", "WHITE",
    "HISPANIC OR LATINO", "NOT HISPANIC OR LATINO"
  ),
  Placebo = c(53, 33, 14, 30, 42, NA, 8, 78, 3, 83),
  Xanomeline_High_Dose = c(40, 44, 11, 18, 55, 1, 9, 74, 3, 81),
  Xanomeline_Low_Dose = c(50, 34, 8, 29, 47, NA, 6, 78, 6, 78),
  stringsAsFactors = FALSE # Do not convert strings to factors
)
gemini_narrative(table_data)

## End(Not run)
```

`gemini_search` *Generate text with real-time information using Google Search (Grounding)*

Description

Generate text responses that include up-to-date information from Google Search

Usage

```
gemini_search(
  prompt,
  temperature = 1,
  maxOutputTokens = 8192,
  topK = 40,
  topP = 0.95,
  seed = 1234
)
```

Arguments

`prompt`	The prompt or question requiring real-time information
`temperature`	The temperature to use. Default is 1 value should be between 0 and 2 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
`maxOutputTokens`	The maximum number of tokens to generate. Default is 8192 and 100 tokens correspond to roughly 60-80 words.
`topK`	The top-k value to use. Default is 40 value should be between 0 and 100 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
`topP`	The top-p value to use. Default is 0.95 value should be between 0 and 1 see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters
`seed`	The seed to use. Default is 1234 value should be integer see https://ai.google.dev/gemini-api/docs/models/generative-models#model-parameters

Details

The API key is now sent via the HTTP header `x-goog-api-key` instead of as a URL query parameter.

Value

Generated text with real-time information from Google Search

See Also

https://ai.google.dev/gemini-api/docs/google-search

Examples

```
## Not run:
library(gemini.R)
setAPI("YOUR_API_KEY")
gemini_search("What is the current Google stock price?")

## End(Not run)
```

---

gemini_structured *Generate structured response from Gemini*

Description

Returns a structured (JSON) response from the Gemini API.

Usage

```
gemini_structured(
  prompt,
  schema,
  model = "2.5-flash",
  temperature = 1,
  maxOutputTokens = 8192,
  topK = 40,
  topP = 0.95,
  seed = 1234,
  timeout = 60
)
```

Arguments

`prompt`	The prompt (question) to send to the model.
`schema`	JSON schema (as a list) for the expected response.
`model`	Model to use. Default is '2.5-flash'.

temperature	Sampling temperature. Default is 1.
maxOutputTokens	Maximum number of output tokens. Default is 8192.
topK	Top-k value. Default is 40.
topP	Top-p value. Default is 0.95.
seed	Random seed. Default is 1234.
timeout	Request timeout in seconds. Default is 60.

Value

A structured list (parsed JSON).

Examples

```
## Not run:
schema <- list(
  type = "ARRAY",
  items = list(
    type = "OBJECT",
    properties = list(
      recipeName = list(type = "STRING"),
      ingredients = list(
        type = "ARRAY",
        items = list(type = "STRING")
      )
    ),
    propertyOrdering = c("recipeName", "ingredients")
  )
)
gemini_structured(
  "List a few popular cookie recipes, and include the amounts of ingredients.",
  schema
)

## End(Not run)
```

---

gemma *Generate text from text with Gemma*

Description

Generate text from text with Gemma with Gemini API

Usage

```
gemma(prompt, model = "gemma-3-1b-it", api_key = NULL, timeout = 60)
```

Arguments

prompt	The prompt to generate text from
model	The model to use. Default is 'gemma-3-1b-it'. see https://ai.google.dev/gemma/docs/get_started#models-list
api_key	Your API key. If NULL, uses GEMINI_API_KEY environment variable.
timeout	Request timeout in seconds. Default is 60.

Value

Generated text

Examples

```
## Not run:
gemma("Roses are red...")

## End(Not run)
```

---

gen_docs *Generate Roxygen Documentation*

Description

Generates Roxygen2 documentation for an R function based on the currently selected code.

Usage

```
gen_docs(prompt = NULL)
```

Arguments

prompt	A character string specifying additional instructions for the LLM. Defaults to a prompt requesting Roxygen2 documentation without the original code.

Value

Invisibly returns the generated documentation string, but primarily inserts the text into the RStudio console.

Examples

```
## Not run:
# Select your function code in the editor, then run:
gen_docs()

# For custom instructions:
gen_docs("Generate minimal Roxygen docs for this function")
```

```
## End(Not run)
```

---

gen_tests *Generate Unit Tests for R Functions*

Description

Generates unit test code for an R function using the Gemini AI model.

Usage

```
gen_tests(prompt = NULL)
```

Arguments

prompt	A character string specifying additional instructions for the Gemini model. If NULL, a default prompt requesting unit tests is used.

Value

Invisibly returns the generated test code, but primarily inserts it into the RStudio console.

Examples

```
## Not run:
# Select your function code in the editor, then run:
gen_tests()

# For custom instructions:
gen_tests("Generate comprehensive testthat tests with edge cases")

## End(Not run)
```

---

nano_banana *Generate, edit, or transfer images using Gemini API*

Description

Generate a new image, edit an existing image, or transfer styles/content between two images using the Gemini API (aka Nano Banana). This function supports image generation from text, image editing with a prompt and a base image, and image transfer between two images.

Usage

```
nano_banana(
  prompt,
  type = "generate",
  img_path = NULL,
  img_path2 = NULL,
  output_path
)
```

Arguments

prompt	Character. The prompt describing the image to generate or edit.
type	Character. The type of operation: "generate" (text-to-image), "edit" (image editing), "transfer" (image-to-image).
img_path	Character. Path to the input image PNG file.
img_path2	Character. Path to the second input image PNG file.
output_path	Character. The filename to save the result image.

Value

The path to the saved image file, or NULL if an error occurred.

Examples

```
## Not run:
# Generate an image from text
prompt <- "Create a picture of a nano banana dish in a fancy restaurant with a Gemini theme"
nano_banana(prompt, output_path = "gemini-native-image.png")

# Edit an image with a prompt (continued from generate)
prompt <- paste(
  "Create a picture of my cat eating a nano-banana",
  "in a restaurant under the Gemini constellation"
)
nano_banana(
  prompt,
  type = "edit",
  img_path = "gemini-native-image.png",
  output_path = "edited_image.png"
)

# Transfer style/content between two images
prompt <- paste(
  "Take the blue floral dress from the first image",
  "and let the woman from the second image wear it."
)
nano_banana(
  prompt,
  type = "transfer",
  img_path = "dress.png",
```

```
  img_path2 = "model.png",
  output_path = "transferred_image.png"
)

## End(Not run)
```

---

setAPI *Set Gemini API Key*

Description

Sets the Gemini API key as an environment variable for use in API calls.

Usage

```
setAPI(api_key)
```

Arguments

api_key	A character string containing your Gemini API key.

Value

No return value, called for side effects.

Note

Please be aware you have to agree to the terms of service of the API provider. Any app that uses the API key is subject to the terms of service. Also, please be aware that the API key is a sensitive information.

See Also

https://makersuite.google.com/app/apikey

Examples

```
## Not run:
setAPI("YOUR_API_KEY")

## End(Not run)
```

setEnv *Store API key in local environment file*

Description

Saves the API key to a local .Renviron file for persistent access across R sessions

Usage

```
setEnv(api_key, overwrite = TRUE, install_message = TRUE)
```

Arguments

api_key	The API key to store
overwrite	Whether to overwrite the existing API key if already present in .Renviron (default: TRUE)
install_message	Whether to display a message about how to use the API (default: TRUE)

Value

No return value, called for side effects.

See Also

setAPI which sets the API key for the current session only

Examples

```
## Not run:
setEnv("your_api_key")

## End(Not run)
```

token.vertex *Generate Gemini Access Token and Endpoint URL*

Description

Generates an access token for the Gemini model and constructs the corresponding endpoint URL.

Usage

```
token.vertex(
  jsonkey = NULL,
  model_id = NULL,
  expTime = 3600,
  region = "us-central1"
)
```

Arguments

jsonkey	A path to JSON file containing the service account key from Vertex AI.
model_id	The ID of the Gemini model. This will be prepended with "gemini-".
expTime	The expiration time of the access token in seconds (default is 3600 seconds, or 1 hour).
region	The Google Cloud region where your Vertex AI resources are located (default is "us-central1"). See https://cloud.google.com/vertex-ai/docs/general/locations for available regions.

Value

A list containing:

key	The generated access token.
url	The endpoint URL for the Gemini model.

Examples

```
## Not run:
library(gemini.R)
tokens <- token.vertex(jsonkey = "YOURAPIKEY.json", model_id = "2.5-flash")

# Specify a different region
tokens <- token.vertex(jsonkey = "YOURAPIKEY.json", model_id = "2.5-flash", region = "europe-west4")

## End(Not run)
```

Index