Package 'PatientGenerator'

May 7, 2026

Type Package

Title Generator of Synthetic Patient Data for the OMOP Common Data Model

Version 0.1.4

Description Tools to generate synthetic patient-level test datasets in the Observational Medical Outcomes Partnership (OMOP) Common Data Model (CDM). Includes a chat-driven generator backed by large language models and an interactive 'shiny' designer for editing CDM test sets.

URL https://github.com/mi-erasmusmc/PatientGenerator, https://mi-erasmusmc.github.io/PatientGenerator/

BugReports https://github.com/mi-erasmusmc/PatientGenerator/issues

License Apache License (>= 2)

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports bslib, checkmate, data.table, DBI, dplyr, DT, duckdb, ellmer, glue, httr2, jsonlite, r2d3, R6, shiny, stringr, testthat

Suggests CDMConnector, CohortCharacteristics, CohortConstructor, covr, ggplot2, knitr, rmarkdown, TestGenerator

VignetteBuilder knitr

Config/Needs/website pkgdown

Config/testthat/edition 3

NeedsCompilation no

Author Cesar Barboza [aut, cre] (ORCID: <https://orcid.org/0009-0002-4453-3071>), Ger Inberg [aut] (ORCID: <https://orcid.org/0000-0001-8993-8748>), Adam Black [aut] (ORCID: <https://orcid.org/0000-0001-5576-8701>)

Maintainer Cesar Barboza <c.barboza@mi-erasmusmc.nl>

Repository CRAN

Date/Publication 2026-05-04 19:10:02 UTC

Contents

`availableModels` `availableModels()` *If the API key is valid in the system, returns available models to the user from the LLM provider.*

Description

OpenAI is the only one currently supported.

Usage

```
availableModels()
```

Value

A string list with the id of a vailable models.

`conceptSearchModule` *Concept search Shiny module*

Description

Reusable UI and server for searching the vocabulary (Hecate) and selecting a concept. Shows a button that opens a modal with text input, search button, and results in a DT table. Optional callback when a concept is selected.

Usage

```
conceptSearchUI(id, buttonLabel = "Concept search")

conceptSearchServer(id, onConceptSelected = NULL, placeholderText = "")
```

Arguments

id	Module namespace id.
buttonLabel	Label for the trigger button (default "Concept search").
onConceptSelected	Optional function of one argument conceptId called when user selects a concept; modal is closed after.
placeholderText	Character string used as placeholder text in the search input field.

Functions

- conceptSearchUI(): UI for the concept search: a single button that opens the search modal.
- conceptSearchServer(): Server for the concept search modal (text input, search, DT, close).

---

getTestSets *Useful to list available datasets in the testCases folder*

Description

Useful to list available datasets in the testCases folder

Usage

```
getTestSets(path = NULL)
```

Arguments

path	Optional directory containing JSON test sets. If NULL, the package resolves a default path with testthat integration.

Value

A list()

hecateClient *Create a Hecate API client*

Description

Create a Hecate API client

Usage

```
hecateClient(baseUrl = NULL, timeoutMs = NULL, apiKey = NULL)
```

Arguments

baseUrl	Base URL of the Hecate API (default from config).
timeoutMs	Timeout in milliseconds (default from config).
apiKey	Optional API key for authorization (default from config).

Value

A client object with class `hecate_client`.

hecateSearch *Search Hecate concepts and return results as a data frame*

Description

Search Hecate concepts and return results as a data frame

Usage

```
hecateSearch(
  query,
  vocabularyId = NULL,
  standardConcept = NULL,
  domainId = NULL,
  conceptClassId = NULL,
  limit = 20,
  client = hecateClient()
)
```

Arguments

query	Character(1); search query (required).
vocabularyId	Character(1) or NULL; optional vocabulary filter (comma-separated).
standardConcept	Character(1) or NULL; e.g. "S" (Standard), "C" (Classification).
domainId	Character(1) or NULL; optional domain filter (comma-separated).
conceptClassId	Character(1) or NULL; optional concept class filter.
limit	Integer(1); max results (default 20, max 150).
client	Hecate client (default hecateClient()).

Value

Data frame of search results, or NULL if an error occurred (API error or bad response shape).

Examples

```
## Not run:
# Simple search
df <- hecateSearch("diabetes")

# Search with filters
df <- hecateSearch("hypertension", domainId = "Condition", limit = 10)

## End(Not run)
```

---

nullOr *Null coalescing operator*

Description

Null coalescing operator

Usage

```
x %||% y
```

Arguments

x	First value (any type).
y	Fallback value when x is NULL.

Value

x if not NULL, otherwise y.

patientChat `patientChat()` *generates synthetic patients in the OMOP-CDM using an LLM API.*

Description

Requires an OPEN_AI_KEY in ~/.Renviron. After that just sent a prompt and save() the results. The JSON file can be used as an OMOP-CDM patient test set.

Details

Accepts a prompt as input. Produces a test set using a structured JSON schema. Utilizes tools such as CodelistGenerator or Hecate to look up concept IDs. Accepts subsequent prompts to modify existing test sets that the LLM uses as context.

This class allows testing patient sets created by the LLM, prompt engineering, integration of search tools and functionality, and creating a set of patients to test analytical packages.

Value

A JSON response that includes: the natural language answer from the LLM and a JSON with test set patients in accordance to the provided schema.

Public fields

`chat` An ellmer chat instance

`json_schema_path` JSON schema to output structured results

`response` Ouput from the LLM

`codelist` A codelist with details to search for concepts ids

Methods

Public methods:

- `patientChat$new()`
- `patientChat$prompt()`
- `patientChat$json_response()`
- `patientChat$output()`
- `patientChat$retrieveCodelist()`
- `patientChat$save()`
- `patientChat$availableModels()`
- `patientChat$clone()`

Method `new()`**:** Create a new chat to create JSON test sets for OMOP-CDM.

Usage:

```
patientChat$new(
  system_prompt = NULL,
  model = "gpt-5.4",
  jsonSchemaPath = NULL,
  echo = c("none", "output", "all"),
  codelist_data = NULL
)
```

Arguments:

`system_prompt` Initial system prompt to impose behaviour to the LLM

`model` Such as "gpt-5.3". For a complete list, call patientChat$availableModels()

`jsonSchemaPath` The JSON schema to structure output from LLM

`echo` How the output will be displayed in the console

`codelist_data` A codelist with details to search for concepts ids

Returns: A new `Person` object.

Method `prompt()`**:** Prompt to request data from LLM API

Usage:

```
patientChat$prompt(prompt)
```

Arguments:

`prompt` A query in character.

Method `json_response()`**:** Output in JSON format

Usage:

```
patientChat$json_response()
```

Method `output()`**:** Returns the chat object

Usage:

```
patientChat$output()
```

Method `retrieveCodelist()`**:** Retrieves and filters data from codelist_data

Usage:

```
patientChat$retrieveCodelist(concept_label = "Stage 1", domain = "Measurement")
```

Arguments:

`concept_label` Filters the concept_name in the codelist with details

`domain` Filters the domain in the codelist with details.

Method `save()`**:** Saves the JSON test set to disk.

Usage:

```
patientChat$save(name = "patient-chat-test", path = NULL)
```

Arguments:

`name` Name of the file

`path` To save the file. If NULL, the package first tries `testthat::test_path("testCases")`, then checks `options(PatientGenerator.testSetDir = "...")`, and finally falls back to the package user data directory.

Method `availableModels()`**:** Retrieves available models from the LLM API.

Usage:

```
patientChat$availableModels()
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
patientChat$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
## Not run:
generator <- patientChat$new()
generator$prompt("Give me 5 patients")
generator$save("my_test")

## End(Not run)
```

`patientChatNaive` `patientChatNaive()` *is a grapper for the ellmer package to send prompts and send the output test set to an LLM. Requires a valid API key.*

Description

Priorities: - Accepts a prompt as an input. - Produces a test set in accordance to the provided JSON schema. - Utilizes tools such as CodelistGenerator or Hecate to look up for functions. - Accepts a subsequent prompt with a test set that the LLM has to use as a context

One function for this tasks allow us to: - Test the test sets created by the LLM. - Test prompt engineering. - Test integration of tools functionality. - Allow us to create fast a small set of patients to test analytical packages.

Usage

```
patientChatNaive(
  prompt = "### Give me a sample of five patients",
  model = "gpt-5.2",
  jsonSchemaPath = NULL
)
```

Arguments

`prompt`	A prompt to the LLM, in character or JSON response.
`model`	The model used by the LLM. Currently only OpenAI models are accepted.
`jsonSchemaPath`	Path to a JSON schema used to structure the response.

Value

A JSON response that includes: the natural language answer from the LLM and a JSON with test set patients in accordance to the provided schema.

`patientDesigner` `patientDesigner()` *is a visual interface based on D3 to construct test datasets for the OMOP-CDM*

Description

`patientDesigner()` is a visual interface based on D3 to construct test datasets for the OMOP-CDM

Usage

```
patientDesigner(path = NULL)
```

Arguments

`path`	Optional folder containing JSON test sets. If NULL, default path resolution keeps testthat integration.

Value

A Shiny app

Index