Desenvolupament de sistema de documentació automàtica aplicant IA per APIs REST

Sergio Bolaños Casado

Resum— Aquest projecte té com a objectiu desenvolupar una eina que permeti la documentació automàtica d'una API REST a partir d'utilitzar un dels models d'Intel·ligència Artificial més coneguts en l'actualitat com és GPT, el model de llenguatge desenvolupat per OpenAI. En aquest document, es realitza una revisió de l'estat de l'art per tal de contextualitzar la temàtica i s'expliquen detalladament tots els aspectes relacionats amb el desenvolupament del projecte. S'inclouen tant els objectius, la metodologia i planificació proposades, realitzant una comparació entre la planificació realitzada a l'inici del projecte i la planificació amb les modificacions finals, i s'inclou tota la part d'implementació, que és on es troba el contingut més pràctic i on s'expliquen els diferents mòduls i components que té l'eina desenvolupada. Els resultats del projecte poden ser valuosos per entendre millor com aplicar la intel·ligència artificial en el camp de la documentació automàtica i contribuir així al progrés i a l'evolució d'un tòpic que s'està desenvolupant àmpliament.

Paraules clau— api, documentació automàtica, gpt, intel·ligència artificial

Abstract— The objective of this project is to develop a tool that allows the automatic documentation of a REST API using one of the best known Artificial Intelligence models currently available, such as GPT, the language model developed by OpenAI. In this document, a review of the state of the art is made to contextualize the subject and all the aspects related to the development of the project are explained in detail. It includes the objectives, the proposed methodology and planning, making a comparison between the planning made at the beginning of the project and the planning with the final modifications, and includes all the implementation part, which is where the most practical content is found and where the different modules and components of the developed tool are explained. The results of the project may be valuable to better understand how to apply artificial intelligence in the field of automatic documentation and thus contribute to the further progress and evolution of a topic that is developing significantly.

Index Terms— api, automatic documentation, gpt, artificial intelligence

1 Introducció - Context del treball

L'APARICIÓ de noves tecnologies d'intel·ligència artificial ha despertat l'interés de nombroses empreses que busquen automatitzar o incorporar aquestes noves tècniques en alguns dels seus processos, considerant-les com el futur dels pròxims anys. En aquest context, sorgeix la col·laboració amb Rosepetal S.L. [1], empresa amb la que s'ha col·laborat per la realització d'aquest projecte de final de grau

Rosepetal és una empresa relativament jove, fundada l'any 2021. Es tracta d'una filial de AIS Vision, una empresa dedicada al disseny de solucions aplicant visió artificial en entorns industrials. Rosepetal neix de la necessitat d'incorporar solucions tecnològiques basades en *deep learning* als diferents projectes, oferint així solucions de visió per computador basades en IA.

Inicialment, quan es proposa el projecte, l'empresa ja compta amb un sistema estàtic que mostra les diverses funcions de la seva API, amb la seva corresponent documentació i, en alguns casos, un terminal de proves. La proposta consisteix a desenvolupar una nova eina similar a l'esmentada anteriorment, però amb la capacitat de generar la documentació de manera automàtica utilitzant el model GPT. D'aquesta manera, es busca aconseguir una escalabilitat del sistema en afegir noves funcionalitats a l'API, assegurant al mateix temps una documentació de qualitat i un entorn interactiu de proves.

- *E-mail de contacte: sergio.bolanos@autonoma.cat*
- *Menció realitzada: Computació*
- *Treball tutoritzat per: Ernest Valveny Llobet (Departament de Ciències de la Computació)*
- *Curs 2022/23*

2 ESTAT DE L'ART

Aquest apartat té com a objectiu donar context sobre quina és la situació actual respecte la documentació automàtica d'APIs, destacant quins són els punts de vista i les eines més populars. També es tractarà la importancia dels models d'IA en aquest camp.

2.1 Enfocaments i eines destacades

La documentació automàtica d'APIs ha esdevingut un element essencial perquè els desenvolupadors puguin comprendre i utilitzar de manera eficaç una API. La qualitat d'aquesta documentació és un factor diferenciador en comparació amb l'elaboració de documentació manual, ja que aquesta pot resultar tediosa i sobretot propensa a errors.

Actualment, hi ha diverses perspectives que destaquen en relació a la documentació automàtica. Dos de les més rellevants són les següents.

2.1.1 Anàlisi de codi

Aquest primer punt de vista consisteix en l'anàlisi del codi font d'una API per extreure informació rellevant i generar la documentació corresponent. Això implica l'obtenció de detalls sobre els paràmetres de les funcions, els tipus de dades amb els que es treballen i els endpoints disponibles.

Dos de les eines que més destaquen en aquest àmbit són Swagger UI [2] i Apiary [3]. Ambdues eines són molt valuoses a l'hora de documentar APIs de manera precisa e interactiva. També faciliten el disseny i la col·laboració en la creació d'APIs, i ofereixen funcionalitats extres com la prova en temps real de les funcions i endpoints, permetent així verificar i validar el funcionament general.

Tot i que aquestes eines són de gran ajuda, també presenten alguns inconvenients. Al tractar-se d'eines relativament tècniques, la seva corba d'aprenentatge per algú que no ha treballat amb APIs pot ser complexa. Un altre punt negatiu a destacar seria la rigidesa a l'hora de treballar amb el codi de l'API. Perquè les eines funcionin correctament, el codi ha de seguir una estructura ben definida. Això pot comportar la necessitat de realitzar modificacions en el codi font existent.

Malgrat els inconvenients tècnics esmentats, aquestes eines no deixen de ser referents en el que es refereix a la documentació automàtica d'APIs basada en l'anàlisi de codi.

2.1.2 Processament de llenguatge natural (NLP)

El processament de llenguatge natural també ha adquirit gran rellevància en el que respecta a la generació automàtica de documentació d'APIs. Implica analitzar la descripció dels endpoints i utilitzar tècniques de processament de llenguatge natural per generar la documentació.

En aquest context, una de les eines més rellevants és OpenAI Codex [4]. Aquesta eina es tracta d'un model basat en IA, descendent més concretament del famós GPT-3. Permet interactuar amb el codi a través d'instruccions de llenguatge natural i generar codi i text en conseqüència a aquestes instruccions.

En resum, cadascun d'aquests punts de vista és molt vàlid en el que respecta a la generació automàtica de documentació. L'anàlisi de codi destaca sobretot per la seva precisió, però com s'ha comentat anteriorment, requereix que el codi sigui d'alta qualitat. D'altra banda, el processament de llenguatge natural és una perspectiva força interesant i prometedora, tenint cada cop més presencia en tot el que respecta a la intel·ligència artificial.

En el context d'aquest projecte, s'utilitzarà una solució basada en el NLP, ja que un dels punts clau és la implementació de GPT per generar tota la documentació, i aquest model es basa principalment en tasques de processament de llenguatge natural.

2.2 Models d'IA

Els models d'intel·ligència artificial són algoritmes dissenyats per simular i replicar capacitats de raonament humanes. Entre les tasques més comunes que poden realitzar aquests models trobem la codificació i la representació del llenguatge.

En el camp de la codificació de text, BERT és un dels models més coneguts. Desenvolupat per Google, permet l'anàlisi del llenguatge de manera bidireccional, sent molt valuós en tasques de comprensió del llenguatge humà i la detecció de correu brossa, entre d'altres.

Tenint en compte el context d'aquest projecte, és necessari un model que sigui capaç de realitzar la tasca de representació i generació de text. Aquest és un dels motius pels quals s'ha triat GPT, un model àmpliament reconegut en l'actualitat.

2.2.1 Models GPT

Els models GPT [5] han guanyat gran notorietat en el camp de les intel·ligències artificials. Destaquen per la seva excepcional capacitat de generar respostes enfront qualsevol ordre.

Tant el GPT-3 com el recent GPT-4 són models que estan basats en *transformers* [6] (tipus de xaxa neuronal basada en l'autoatenció que te capacitat d'entendre el context semàntic d'una instrucció o frase) preentrenats amb una enorme quantitat d'informació procedent de llibres publicats, Wikipedia, milions de pàgines web i documents científics que es poden trobar per Internet.

Aquesta habilitat per generar respostes de manera coherent és el que els converteix en un component clau a l'hora de generar documentació, ja que ofereixen gran eficiència i sobretot gran qualitat en les seves respostes.

3 OBJECTIUS

Per assolir l'objectiu general de desenvolupar una eina capaç de documentar l'API de l'empresa a partir d'intel·ligència artificial, s'han desglossat els següent subobjectius:

1. Desenvolupar la interfície de l'eina, permetent generar tota la informació de les funcions de l'API.
2. Integrar un terminal per poder testejar cadascuna de les funcions de l'API, de manera que l'usuari tingui un entorn de probes interactiu.
3. Un cop desenvolupada tota l'eina amb el terminal integrat, integrar-la amb tota la interfície de l'empresa per poder realitzar crides als mètodes de l'API desde l'eina.

4 METODOLOGIA

Durant la realització d'aquest projecte, la metodologia triada ha sigut una metodologia de tipus *Agile*, més concretament *Scrumban* [7].

Aquest tipus de metodologia permet combinar elements de Scrum i Kanban per aconseguir una millor adaptabilitat i flexibilitat durant la realització del projecte, ja que combina els principis de planificació e iteració de Scrum amb la representació visual del treball de Kanban.

Els diferents elements i eines en els que es pot descomposar la metodologia emprada són els següents:

- **Sprints:** Cicles de treball de dues setmanes, amb un *sprint review* al finalitzar el període.
- **Daily meetings:** Reunions diàries per portar un seguiment continu del treball realitzat.
- **IDE (Integrated Development Environment):** L'IDE utilitzat en aquest cas ha sigut Visual Studio Code.
- **Control de versionat:** Per poder gestionar un correcte control de versions, s'ha utilitzat Github.
- **Trello**: Eina utilitzada per portar la gestió visual del flux de treball. D'aquesta manera, tots els membres de l'equip poden veure la càrrega de treball i les tasques assignades a cada integrant.

5 PLANIFICACIÓ

Aquest apartat pretén mostrar l'estructuració inicial de les tasques establertes a l'inici del projecte i realitzar una comparació amb les modificacions realitzades durant el desenvolupament.

Planificació inicial

Inicialment, la planificació estava desglosada tal com es mostra en la *Taula 1*:

Taula 1: Planficació Inicial

Tasca	Inici	Final
1 Recerca d'informació		
1.1 Aprenentatge Framework Vue.js	13/03/23	24/03/23
1.2 Consultar generadors automàtics	24/03/23	25/03/23
2 Anàlisi de requisits		
2.1 Plantejament d'objectius	27/03/23	
3 Disseny de l'estructura		
3.1 Divisió dels mòduls	27/03/23	
3.2 Definició de les característiques de cada mòdul	27/03/23	
4 Implementació		
4.1 Mòdul Generació de documentació	28/03/23	19/04/23
4.2 Mòdul Terminal Interactiu	24/04/23	10/05/23
4.3 Mòdul Integració amb el sistema	15/05/23	24/05/23
5 Validació		
5.1 Proves Mòdul Generació de documentació	20/04/23	21/04/23
5.2 Proves Mòdul Terminal Interactiu	11/05/23	12/05/23
5.3 Proves Mòdul Integració amb el sistema	25/05/23	26/05/23

Modificacions

A mesura que s'ha desenvolupat el projecte, certes tasques han resultat modificades en quant a les dates previstes. Això s'ha vist causat pel mòdul principal i del que depenien els altres, el Mòdul de Generació de Documentació.

La finalització d'aquest mòdul ha estat posposada al 28/04/23, tenint el següent impacte en els altres mòduls:

- Data inici Mòdul Terminal: 02/05/23
- Data inici Mòdul Integració: Descartat

Com es pot veure, el tercer dels mòduls ha sigut descartat de la realització del projecte, degut a que per terminis no era viable desenvolupar-ho.

Motius

Aquests són alguns dels motius que justifiquen l'endarreriment i les corresponents afectacions del projecte:

1. Temps addicional en realitzar una verificació exhaustiva dels resultats generats per GPT.
2. Incorporació de funcionalitats no proposades en un inici.
3. Realització de proves amb el nou model GPT-4.
4. Experimentació amb *fine-tuning*, procés que consisteix en entrenar el model amb conjunts de preguntes-respostes per tal de millorar els resultats.

6 IMPLEMENTACIÓ

En aquest apartat s'explica tota la part tècnica que constitueix el projecte. Es tractarà des de les especificacions tècniques a nivell de requisits fins a la implementació dels mòduls més importants.

6.1 Tecnologies utilitzades

Les tecnologies utilitzades pel desenvolupament web de l'eina han sigut Vue.js i Node.js:

- **Vue.js [8]:** Framework de Javascript utilitzat principalment per la creació de interfícies interactives. Moltes de les seves funcionalitats es basen en la reactivitat, es a dir, la possibilitat d'interactuar amb la pàgina web sense la necessitat que es torni a carregar el seu contingut. Com a característica destaca que els arxius Vue tenen en el mateix arxiu codi de HTML, Javascript i CSS.

- **Node.js [9]:** Entorn d'execució de Javascript, utilitzat principalment a la part del servidor. És el que permet la transmissió de les dades a la interfície creada amb Vue.

A partir de la combinació d'aquestes dues tecnologies es pot crear una aplicació web que sigui completa i escalable.

6.2 Estructura de l'API

L'estructura d'una API determina la forma en la que aquesta està organitzada i dissenyada per facilitar la interacció dels sistemes que l'utilitzen.

El que es pretén amb aquesta secció és mostrar i explicar com està estructurada l'API de Rosepetal amb la qual s'ha treballat. Aquesta es divideix en diferents arxius Javascript, on cadascun representa un controlador amb una sèrie de funcions. Aquesta pràctica és típica quan es vol mantenir una estructura organitzada i modular.

A la *Fig. 1* es mostra l'estructura d'arxius amb una breu descripció de les funcionalitats que incorpora cadascun.

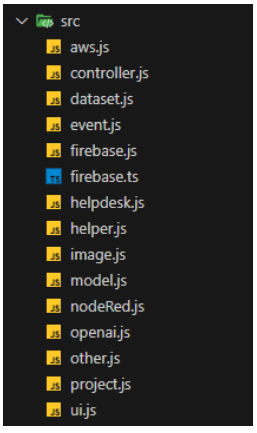


Fig. 1: Estructura arxius API Rosepetal

- **aws.js:** Conté les funcionalitats per treballar amb els serveis d'Amazon Web Services.

- **controller.js:** S'utilitza per carregar les diferents pàgines / seccions de la web de Rosepetal.

- **dataset.js:** Serveix per realitzar modificacions dels diferents datasets que es troben guardats a la base de Firebase.

- **event.js:** Totes les modificacions, ja sigui de datasets o imatges, queden guardades en un registre de la base de Firebase per tenir constància de tot el que s'ha realitzat. Aquest controlador implementa funcionalitats per treballar amb aquests canvis.

- **firebase.js:** Incorpora les funcionalitats per configurar l'entorn de Firebase i així permetre des de altres controladors el poder treballar sobre la base de dades.

- **firebase.ts:** Conté les mateixes funcionalitats que firebase.js, però en una versió de Typescript.

- **helpdesk.js:** Aquest es el controlador que s'ha creat destinat a l'eina desenvolupada. Conté les funcionalitats bàsiques d'aquesta.

- **helper.js:** Té funcionalitats bàsiques com poden ser el transformar un *string* a un *Uint8Array,* o bé posar en majúscules tot un *string*.

- **image.js:** Funcionalitats destinades al tractament de les imatges i les metadades d'aquestes.

- **model.js:** Permet fer les accions bàsiques com la creació, modificació i entrenament dels models d'intel·ligència artificial que es desenvolupen.

- **nodeRed.js:** Node-RED [10] és una eina que permet la programació visual de fluxos de treball. Aquest controlador incorpora les funcionalitats necessàries.

- **openai.js:** Controlador mitjançant el qual es realitzen totes les peticions a l'API de OpenAI per treballar amb els models GPT.

- **other.js:** Similar a helper.js, incorpora funcionalitats auxiliars com pot ser *getIcons* per obtenir els diferents icones de la web.

- **project.js:** S'utilitza per obtenir informació general, com pot ser la versió de l'API.

- **ui.js:** Incorpora funcionalitats per carregar els elements visuals de la interfície.

6.3 GPT

OpenAI [11] és una destacada empresa en l'àmbit de la intel·ligència artificial que ha desenvolupat diferents models amb bastant renom en l'actualitat.

Aquest apartat es centrarà sobretot en el model GPT, el qual s'ha utilitzat per la generació automàtica de la documentació de l'eina. El que es pretén és proporcionar una visió completa i detallada dels aspectes més rellevants.

6.3.1 Versions

El primer aspecte a destacar del model és el versionat. El fet de treballar amb una tecnologia que està en constant millora comporta la possibilitat que a mesura que es desenvolupa l'eina, apareixin noves característiques que s'han de tenir en compte ja que poden ser útils pel treball que s'està realitzant.

Durant el projecte, s'han probat els següents models:

- gpt-3.5-turbo
- gpt-4
- gpt-3.5-turbo-16k-0613
- gpt-4-0613

A continuació es realitza una comparativa entre els models que han sortit en la mateixa *release date*, per així explicar la justificació de la tria de models en cada moment durant el projecte.

És necessari destacar que la comparativa no es basa en valors quantitatius, sinó en una opinió empírica fonamentada durant aquests mesos de treball amb els models.

gpt-3.5-turbo | gpt-4

Taula 2: Comparació models GPT 1

Model	Valoració
gpt-3.5-turbo	• Respostes més ràpides • Tokens: 4.096
gpt-4	• Millor comprensió del context de les peticions • Tokens: 8.192

gpt-3.5-turbo-16k-0613 | gpt-4-0613

Taula 3: Comparació models GPT 2

Model	Valoració
gpt-3.5-turbo-16k-0613	• Respostes més ràpides • Tokens: 16.384 • Incorpora *function calling*
gpt-4-0613	• Millor comprensió del context de les peticions • Tokens: 8.192 • Incorpora *funcion calling*

Valoració final

Tenint en compte les comparacions realitzades, el model triat finalment ha sigut l'última versió del model gpt-3.5, gpt-3.5-turbo-16k-0613.

Encara que el model gpt-4 acostuma a funcionar millor en situacions que poden arribar a ser complexes, la valoració que s'obté és que no hi ha prou justificació per triar-ho en lloc de la versió del gpt-3.5.

És important tenir en compte el context en aquest cas, ja que les peticions que es realitzen en algunes situacions requereixen molts tokens. Aquest és un dels principals motius per triar la versió del gpt-3.5, ja que aquest duplica la quantitat de tokens que pot rebre respecte el gpt-4.

6.3.2 Peticions

El segon aspecte a destacar és com es realitzen les peticions a OpenAI. Mitjançant aquestes peticions és com s'interactiua amb els models GPT.

A la *Fig. 2* es mostra un exemple de com es realitza una petició a l'API de OpenAI.

```
const messages = [
    { role: 'system',   content: 'Interactua com una calculadora' },
    { role: 'user',     content: 'Quant és 2 + 2?' },
];

openai.createChatCompletion({
    model:          'gpt-3.5-turbo-16k-0613',
    messages:       messages,
    temperature:    temperature || 0,
    n:              1,
})
```

Fig. 2: Petició bàsica OpenAI

Els elements que es mostren a la imatge són:

- **messages:** Array que conté le petició que es vol realitzar. Com es pot observar, el *role* denominat 'system' indica a OpenAI com es vol que es comporti el sistema. El *role* 'user' indica la petició que es vol realitzar.
- **model:** *String* que indica a OpenAI el model que es vol utilitzar, en aquest cas l'última versió del gpt-3.5.
- **temperature:** Paràmetre opcional que estableix el 'grau de creativitat' en les respostes de GPT. Aquest paràmetre pot tenir valors entre 0 i 1.
- **n:** Número de respostes que es vol obtenir amb la petició realitzada.

A l'apartat A.1 de l'apèndix es mostren algunes de les peticions que s'han realitzar per obtenir le documentació de les diferents funcions.

Function Calling
Una de les novetats més recent incorporada a algunes de les últimes versions del model GPT com es mostra a la *Taula 3*, es el *function calling* [12].

Aquesta nova funcionalitat incorpora la capacitat de cridar a les funcions de la pròpia API a partir de realitzar una petició a l'API de OpenAI.

El format de les peticions en aquest cas es veu afectat respecte el format estàndard d'una petició a GPT, el qual s'ha mostrat anteriorment.

Per realitzar la petició, s'inclou un arxiu en format JSON mitjançant el qual s'indica a GPT les funcions que té l'API. A la *Fig. 3* es mostra com es veuria modificada una de les peticions.

```
openai.createChatCompletion({
    model:          'gpt-3.5-turbo-16k-0613',
    messages:       messages,
    temperature:    temperature || 0,
    functions:      functions   || undefined,
    function_call:  functions ? 'auto' : undefined
})
```

Fig. 3: Petició function calling OpenAI

Com es pot observar, hi ha dos nous paràmetres:

- **functions**: Paràmetre que conté el JSON en cas de que s'indiqui aquest. Sinó, s'assigna *undefined* com a valor.

- **function_call:** Paràmetre que pren com a valor 'auto' per triar entre les funcions indicades, aquella que compleix amb la petició indicada al *role* 'user' explicat anteriorment.

6.3.3 Context
El tercer i últim aspecte a tenir en compte a l'hora de treballar amb GPT és el context. Quan s'utilitza el model a partir de l'API de OpenAI, aquest no és conscient de les converssacions que ha tingut prèviament. Per aquest motiu, és necessari mantenir un històric de les peticions que es realitzen.

El format triat per guardar l'històric es mostra a la imatge Fig. 4.

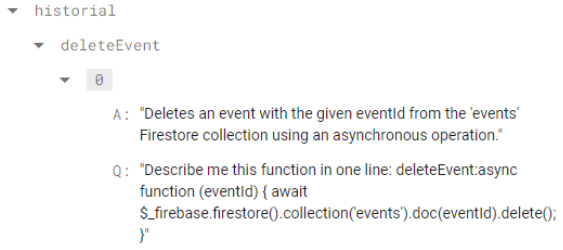


Fig. 4: Exemple històric

D'aquesta manera, per cada controlador de l'API de Rosepetal, s'implementa un històric que registra totes les preguntes i respostes realitzades per cada funció del controlador en qüestió.

Aquesta funcionalitat permet que en cada petició que es vulgui realitzar a GPT, es pugui obtenir la conversació corresponent i concatenarla a l'array *messages* mencionat a l'apartat 6.3.2.

6.4 Firebase
Firebase [13] és una plataforma digital que permet, entre altres moltes funcionalitats, tenir una base de dades en temps real.

Aquest és un component essencial en el funcionament general de tota l'eina. La seva importància resideix en la possibilitat de guardar tota la informació que es genera de les peticions de OpenAI, com si es tractés d'una memòria caché.

El que s'aconsegueix amb això és:

1. Estalviar crides a l'API de OpenAI, i per tant estalviar temps d'obtenció de la informació de les funcions.
2. Mantenir la mateixa documentació sempre i quan no es vulgui modificar aquesta, ja que no té sentit generar documentació nova cada vegada que s'utilitza l'eina.

A la següent imatge es mostra com quedaria guardada la informació d'una funció a la base de Firebase:

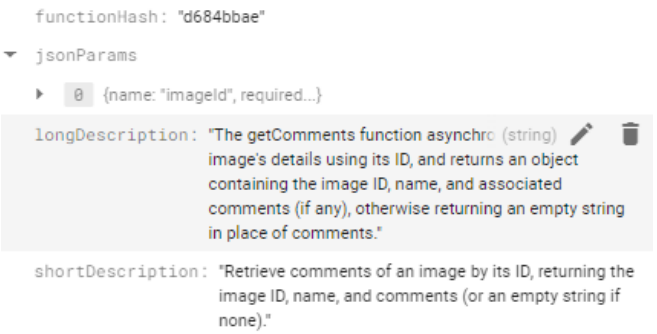


Fig. 5: Exemple informació Firebase

7 Resultats
Aquest apartat té com objectiu mostrar cadascuna de les parts desenvolupades de l'eina.

Primer de tot, es mostraran les funcionalitats que són una novetat respecte l'eina que hi havia anteriorment. A continuació, es realitzarà una comparació d'ambdues eines, analitzant els resultats que proporcionen cadascuna d'aquestes.

7.1. Noves funcionalitats

Comprovació de canvis

Aquesta primera funcionalitat permet tenir un control dels canvis que es puguin realitzar a les funcions de l'API i així avisar a l'usuari. A la *Fig. 6* es mostra l'avís que apareixeria en cas d'alguna modificació.

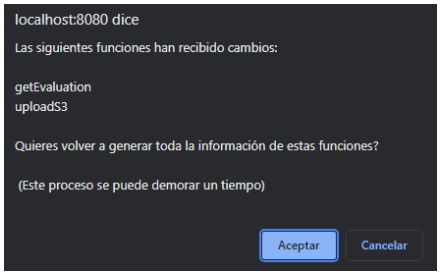


Fig. 6: Avís modificació funcions

Com es pot veure, l'usuari tindria l'opció de tornar a generar la informació de les funcions modificades.

Aquesta implementació s'incorpora mitjançant el hash de cadascuna de les funcions. Quan es genera per primer cop la documentació, també es genera el hash de la funció corresponent i s'emmagatzema a Firebase. La segona vegada que es carrega la documentació, es compara si el hash de les funcions continua sent el mateix amb el que hi ha guardat.

Modificació del sistema

Aquesta segona funcionalitat permet a l'usuari indicar al sistema com vol que es comporti a l'hora de resoldre les peticions.

Cas pràctic

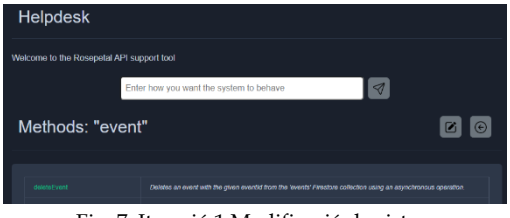


Fig. 7: Iteració 1 Modificació de sistema

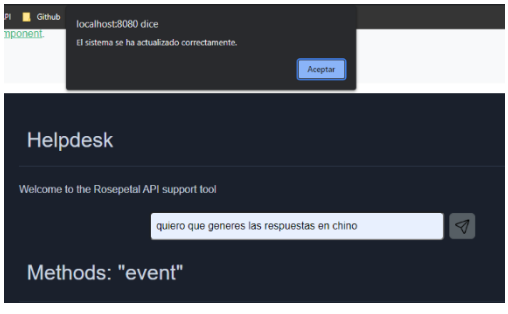


Fig. 8: Iteració 2 Modificació de sistema

Fig. 9: Iteració 3 Modificació de sistema

Eina d'edició

Aquesta tercera i última nova funcionalitat permet modificar diferents apartats dins de l'eina.

En el cas de les descripcions generades, permet tornar a generar-les en cas que l'usuari ho desitji. En la següent imatge es mostra el botó per cambiar al mode edició.

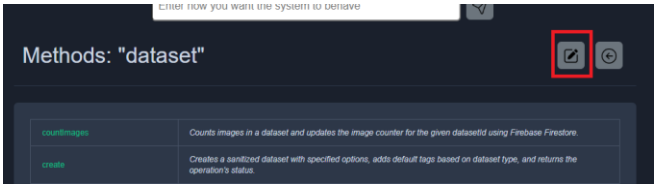


Fig. 10: Botó Mode edició

Un cop seleccionat el mode, com es mostra a la següent figura, s'habilita la possibilitat de tornar a carregar les descripcions de les funcions i guardar-les en cas de que les descripcions generades siguin més adequades.

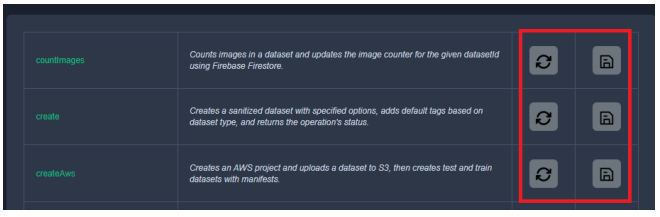


Fig. 11: Regenerar descripcions

Aquesta eina també permet modificar les taules dels paràmetres mostrades quan es selecciona una funció, com es mostra a continuació.

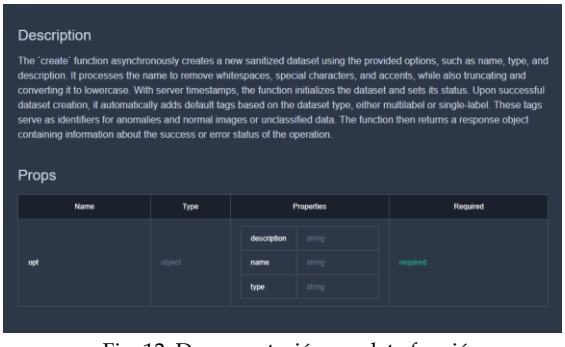


Fig. 12: Documentació completa funció

Al seleccionar el mode edició, es pot modificar completament la taula corresponent als paràmetres de la funció. Això és molt útil quan la taula generada per GPT és incorrecte i es vol modificar manualment.

Fig. 13: Modificació taula paràmetres

7.2. Comparació de resultats

Per realitzar la comparació d'una manera més adient, s'han afegit les imatges a l'Apèndix A.2, per així poder mostrar d'una manera més adequada la documentació que genera cadascuna de les eines, així com imatges dels dos terminals integrats.

A partir de les imatges de l'Apèndix, es poden obtenir les següents observacions:

1. L'escalabilitat és un dels punts més diferenciadors entre eines, pel fet que la versió antiga contenia informació estàtica de les funcions. Això suposava un problema a l'hora d'afegir noves funcions a l'API de Rosepetal, ja que s'havia de documentar de manera manual. La versió nova de l'eina soluciona de manera satisfactòria aquest inconvenient. Això es pot apreciar en les imatges 1 i 2 de l'apartat mencionat de l'apèndix.

2. Les descripcions breus generades per l'eina desenvolupada proporcionen més quantitat d'informació en comparació a les descripcions breus de l'eina antiga.

3. Les descripcions completes proporcionades per la nova eina ajuden a entendre millor què és el que realitzen les funcions.

4. Les taules de paràmetres generades per la nova eina ajuden més a la comprensió dels diferents paràmetres que rep la funció.

5. Els terminals d'ambdues eines incorporen funcionalitats similiars, ja que no hi ha intervenció de GPT en aquest cas.

8 Conclusions

Malgrat no haver assolit tots els objectius proposats en un inici, sí que es pot afirmar que el funcionament de l'eina desenvolupada és l'esperat i dona uns resultats que són satisfactoris en comparació a l'eina que hi havia abans. En addició a aquesta valoració, s'han extret les següents conclusions:

1. El projecte ha demostrat la viabilitat i utilitat de l'ús del model GPT en el context específic del domini de l'aplicació.

2. S'ha demostrat que la generació automàtica de documentació mitjançant l'ús de GPT pot estalviar temps i recursos significatius en comparació als mètodes tradicionals.

3. S'ha identificat la importància de portar un manteniment i revisió de les respostes generades pel model, ja que, encara que és molt potent, pot introduir errors o generar informació inexacta.

4. S'ha reconegut que, malgrat els avenços del model, aquest té dificultats per interpretar adequadament certes funcions, sobretot si aquestes manquen d'un bon context i una bona estructura.

Basant-nos en aquest informe i en la realització del projecte es pot determinar una conclusió final i que pot resumir molts dels punts vists al llarg del treball. La contínua evolució d'aquest tipus de model no deixa de meravellar al món amb la capacitat que pot arribar a assolir. No obstant això, s'ha de ser conscient que no tot el que proporcionen com a resultat és perfecte i que en casos com aquest, on el resultat determina la utilitat de l'eina, és necessària, encara que sigui mínima, una intervenció humana.

9 Possibles Millores

Com a últim apartat es presenten possibles ampliacions o millores que es podrien incorporar a l'eina desenvolupada amb l'objectiu de millorar-ne les funcionalitats i l'experiència d'ús.

1. Finalitzar la implementació de l'eina amb la interfície de Rosepetal, tal com s'indica en el tercer dels objectius.

2. Afegir un xat interactiu amb GPT per poder fer preguntes sobre les funcions que es mostren. Això permetrà obtenir informació dinàmica e interactiva en lloc de mostrar només informació estàtica.

3. Habilitar la capacitat d'executar comandes al terminal per fer-ho sentir com un terminal real.

AGRAÏMENTS

Vull agrair al meu tutor durant aquest Treball de Fi de Grau, Ernest Valveny, pel seguiment realitzat, i també a l'empresa Rosepetal per donar-me l'oportunitat de realitzar aquest projecte amb la seva col·laboració.

BIBLIOGRAFIA

[1] «AIS VISION - SOBRE NOSOTROS». https://www.rosepetal.ai/sobrenosotros (accedido 29 de junio de 2023).

[2] Chakray, «SWAGGER y SWAGGER UI: ¿Qué es y por qué es imprescindible en APIS?», Chakray, 27 de enero de 2017. https://www.chakray.com/es/swagger-y-swagger-ui-por-que-es-imprescindible-para-tus-apis/ (accedido 26 de marzo de 2023).

[3] H. C. Romero, «Documenta tu API con Apiary», Adictos al trabajo, 17 de agosto de 2017. https://www.adictosaltrabajo.com/2017/08/17/documenta-tu-api-con-apiary/ (accedido 26 de marzo de 2023).

[4] «OpenAI Codex». https://openai.com/blog/openai-codex (accedido 28 de mayo de 2023).

[5] «Evolución de los modelos GPT: Un recorrido por las innovaciones en Procesamiento del Lenguaje Natural», 14 de abril de 2023. https://wwwhatsnew.com/2023/04/14/evolucion-de-los-modelos-gpt-un-recorrido-por-las-innovaciones-en-procesamiento-del-lenguaje-natural/ (accedido 28 de mayo de 2023).

[6] «Uso de redes neuronales tipo transformer: Novedades en Wolfram Language 12». https://www.wolfram.com/language/12/neural-network-framework/use-transformer-neural-nets.html.es (accedido 28 de junio de 2023).

[7] Asana, «Scrumban: lo mejor de dos metodologías ágiles [2022] • Asana», Asana. https://asana.com/es/resources/scrumban (accedido 1 de julio de 2023).

[8] «Introducción — Vue.js». https://es.vuejs.org/v2/guide/ (accedido 28 de junio de 2023).

[9] «Acerca», Node.js. https://nodejs.org/[...path] (accedido 28 de junio de 2023).

[10] «Node-RED». https://nodered.org/ (accedido 29 de junio de 2023).

[11] «OpenAI». https://openai.com/ (accedido 29 de junio de 2023).

[12] «Function calling and other API updates». https://openai.com/blog/function-calling-and-other-api-updates (accedido 29 de junio de 2023).

[13] «Qué es Firebase: funcionalidades, ventajas y conclusiones», DIGITAL55, 17 de mayo de 2020. https://digital55.com/blog/que-es-firebase-funcionalidades-ventajas-conclusiones/ (accedido 1 de julio de 2023).

APÈNDIX

A1. Format peticions

```
case 'shortDescription-1':
    return {
        id: 'shortDescription',
        message: `Describe me this function in a very short way, with only one line: \n\n${methodF}`,
        params: {
            max_tokens: 50,
        }
    };
case 'shortDescription-2':
    return {
        id: 'shortDescription',
        message: `Give me another version of the short description of the function.`,
        params: {
            max_tokens: 50,
            temperature: 0.5,
        }
    };
case 'longDescription-1':
    return {
        id: 'longDescription',
        message: `Extend the description of the function a little further. Don't explain the operation step by step, just give some more general information.`,
        params: {
            max_tokens: 200,
        }
    };
```

Fig 1: Contingut de les peticions realitzades per obtener la informació de les funcions

A2. Comparació resultats

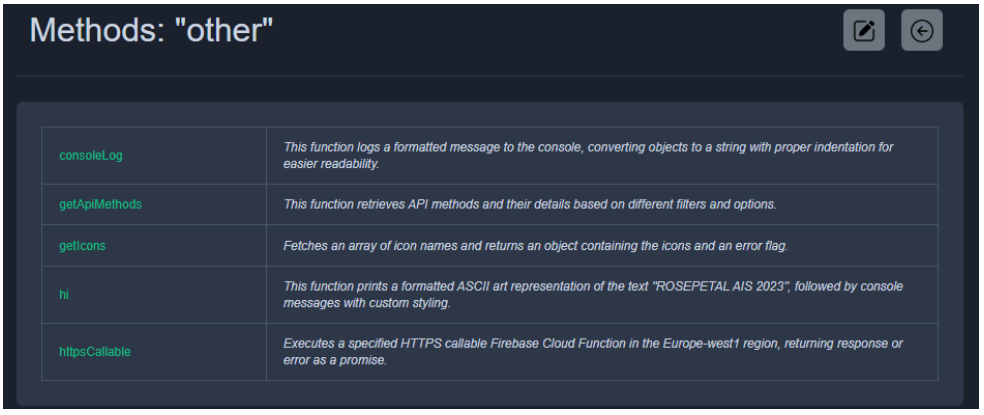


Fig 1: Exemple 1 Descripcions breus generades per la nova versió de l'eina

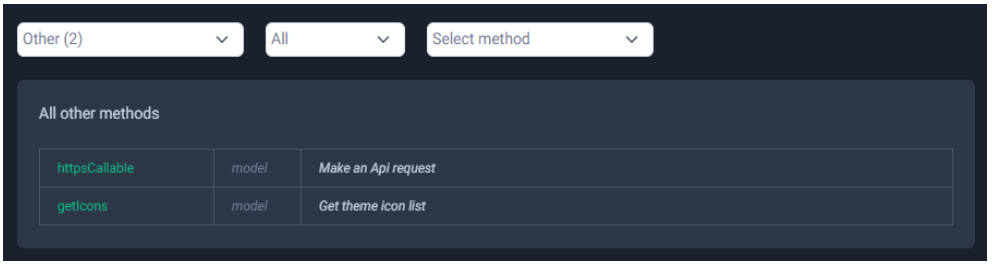


Fig 2: Exemple 1 Descripcions breus generades per l'antiga versió de l'eina

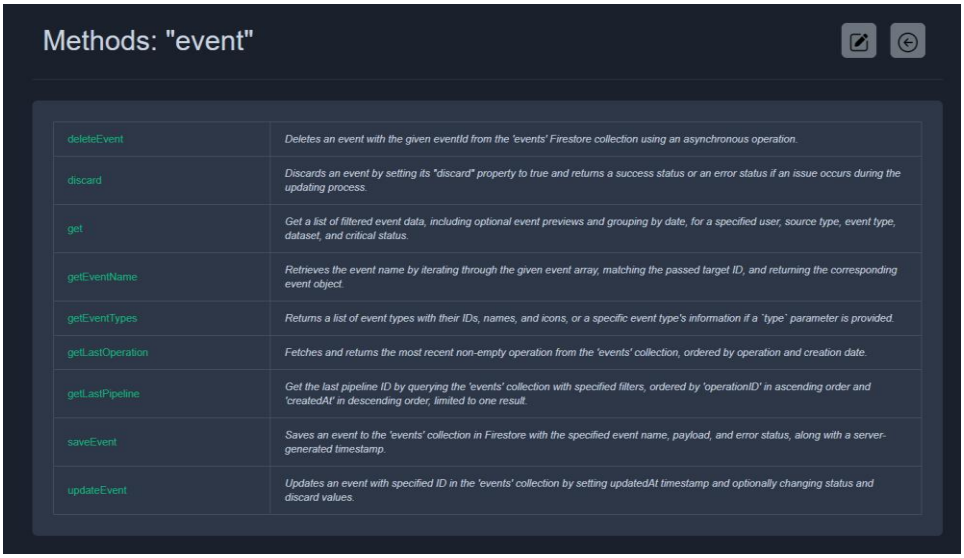


Fig 3: Exemple 2 Descripcions breus generades per la nova versió de l'eina

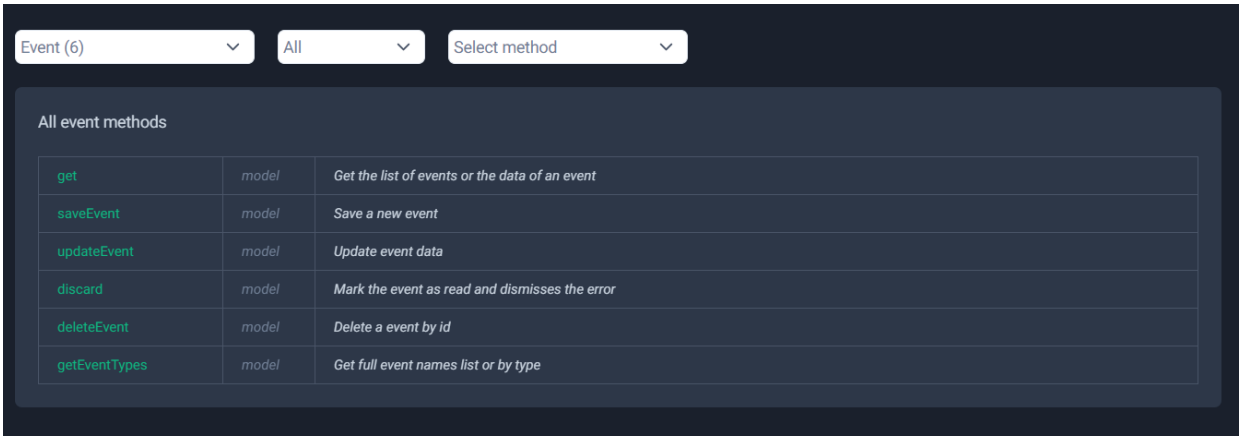
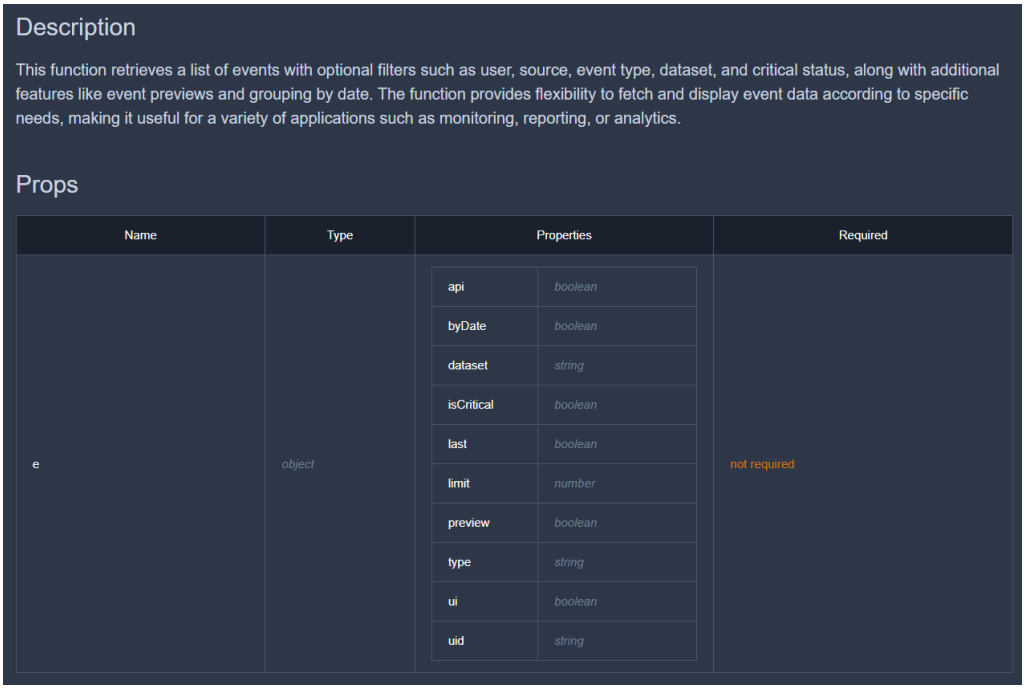


Fig 4: Exemple 2 Descripcions breus generades per l'antiga versió de l'eina



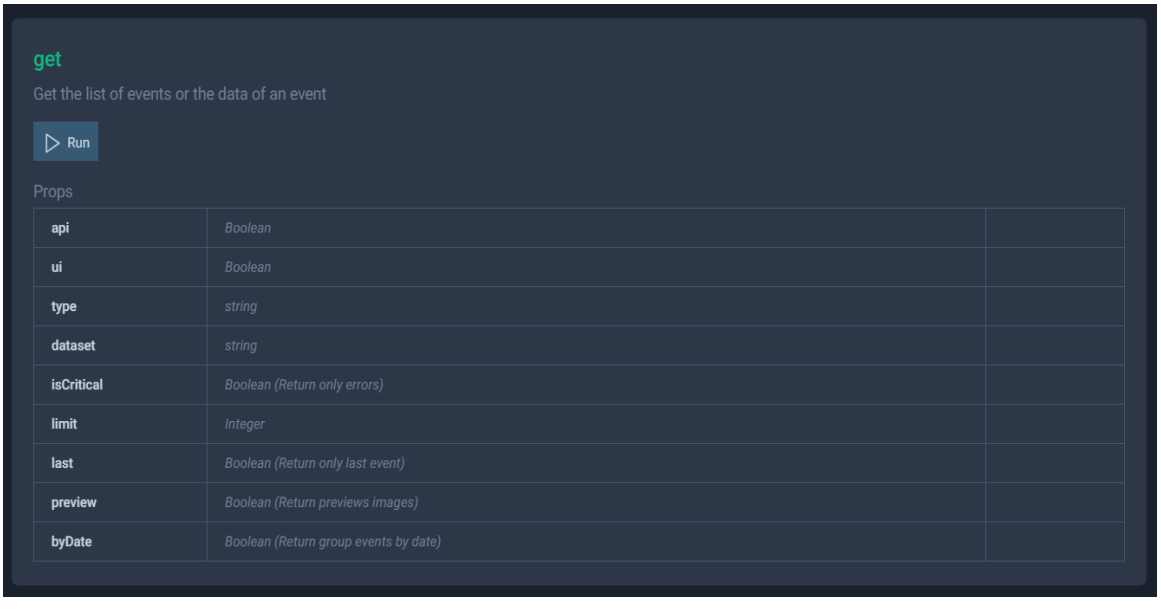
Description

This function retrieves a list of events with optional filters such as user, source, event type, dataset, and critical status, along with additional features like event previews and grouping by date. The function provides flexibility to fetch and display event data according to specific needs, making it useful for a variety of applications such as monitoring, reporting, or analytics.

Props

Name	Type	Properties	Required
e	*object*	api *boolean* byDate *boolean* dataset *string* isCritical *boolean* last *boolean* limit *number* preview *boolean* type *string* ui *boolean* uid *string*	not required

Fig 5: Descripcions completes i taula de paràmetres generades per la nova versió de l'eina



get

Get the list of events or the data of an event

▷ Run

Props

api	*Boolean*	
ui	*Boolean*	
type	*string*	
dataset	*string*	
isCritical	*Boolean (Return only errors)*	
limit	*Integer*	
last	*Boolean (Return only last event)*	
preview	*Boolean (Return previews images)*	
byDate	*Boolean (Return group events by date)*	

Fig 6: Descripcions completes i taula de paràmetres generades per l'antiga versió de l'eina

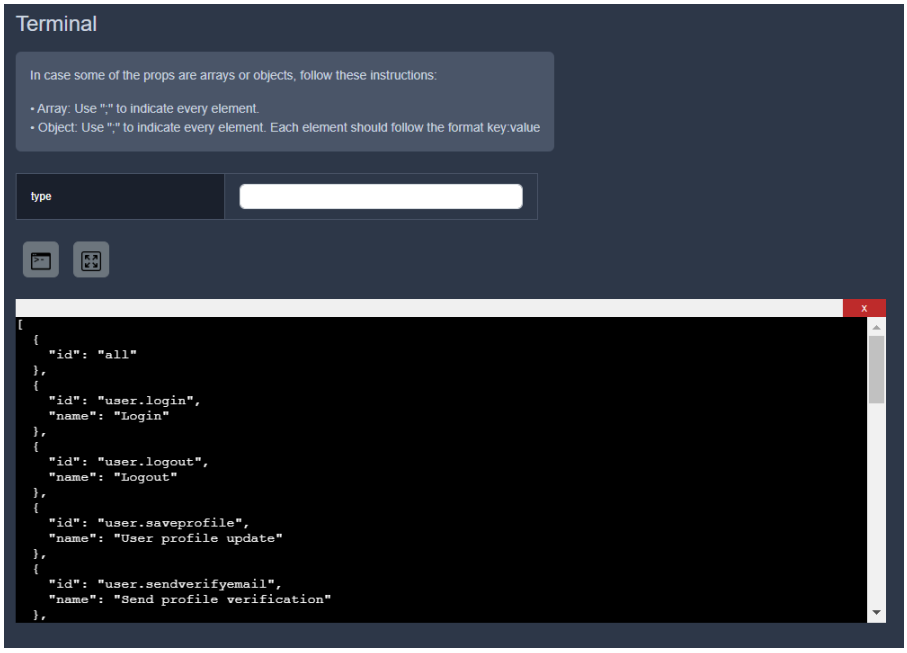


Fig 7: Terminal integrat versió actual de l'eina

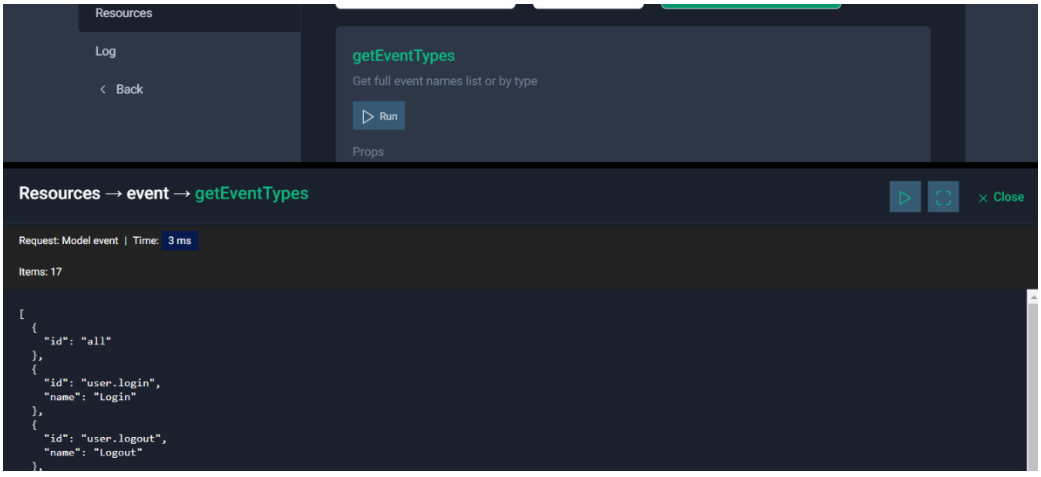


Fig 8: Terminal integrat versió antiga de l'eina