



# ***Aula 10: IDE+IA e APIs com Python+IA***

# Integrated Development Environment (IDE)

- Um ambiente com várias ferramentas integradas para

- Desenvolver,
- Testar,
- Debugar,
- Documentar,
- Empacotar / Implantar, entre outras.

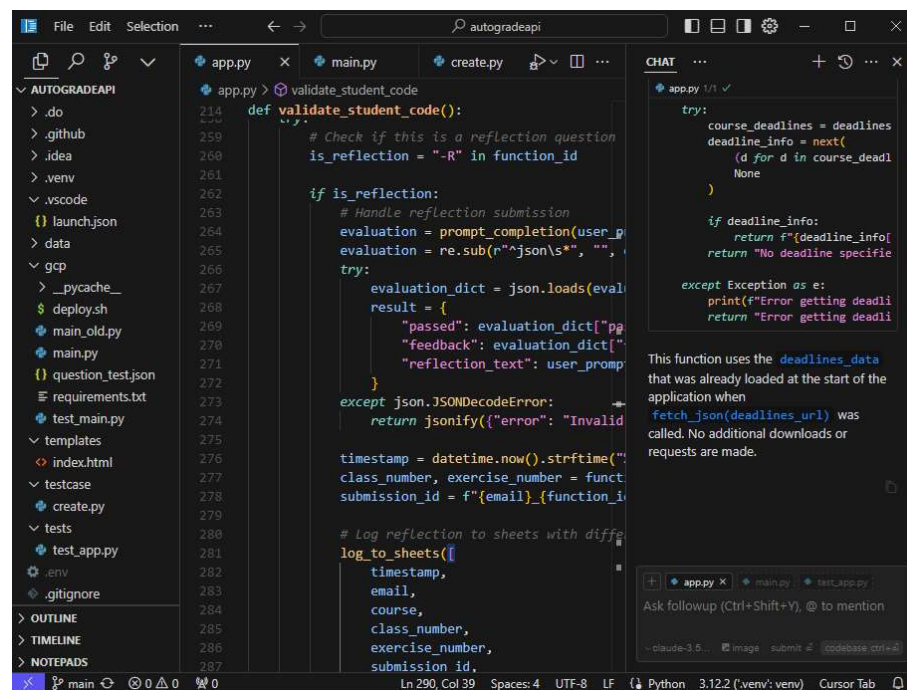
- Principais IDEs de Python

- Google Colab e Jupyter Lab (web)
- Cursor, VS Code, Pycharm

- PDCA de Dev/CICD com IA

- Disruptivo

✓ Automação está no core do desenv. de software



```
def validate_student_code():  
    # Check if this is a reflection question  
    is_reflection = "-R" in function_id  
  
    if is_reflection:  
        # Handle reflection submission  
        evaluation = prompt_completion(user_prompt, reflection_text)  
        evaluation = re.sub(r"^\s*", "", evaluation)  
        try:  
            evaluation_dict = json.loads(evaluation)  
            result = {  
                "passed": evaluation_dict["passed"],  
                "feedback": evaluation_dict["feedback"],  
                "reflection_text": user_prompt  
            }  
        except json.JSONDecodeError:  
            return jsonify({"error": "Invalid JSON"})  
  
        timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")  
        class_number, exercise_number = function_id.split("_")  
        submission_id = f"{email}_{function_id}_{timestamp}"  
  
        # Log reflection to sheets with diff  
        log_to_sheets([  
            timestamp,  
            email,  
            class_number,  
            exercise_number,  
            submission_id  
        ])
```

CHAT

```
try:  
    course_deadlines = deadlines  
    deadline_info = next(  
        (d for d in course_deadlines if d["class"] == class_number)  
    )  
  
    if deadline_info:  
        return f"{deadline_info['deadline']}  
    else:  
        return "No deadline specified"  
  
except Exception as e:  
    print(f"Error getting deadline info: {e}")  
    return "Error getting deadline info"
```

This function uses the `deadlines_data` that was already loaded at the start of the application when `fetch_json(deadlines_url)` was called. No additional downloads or requests are made.

## *Experiência de Usuário com o Cursor: excelente!*

- OpenSource baseado no VS Code
  - Performático (usa poucos recursos do PC)
- Atalhos importados da sua IDE anterior (VS Code ou Pycharm)
- Ambiente virtual reutilizado da IDE Anterior
- **Desenvolvimento iterativo com chatbot de IA**
  - Aplicação de solução com um clique
  - Execução de código no terminal com um clique
    - ✓ que resolve um problema no código (Ex. instalar uma biblioteca)
  - Refactoring de várias linhas de código semelhantes
- Autocomplete conveniente e oportuno
- Agente de IA?
  - Cria arquivos de dados

## *A versão free dá pra te fazer entender o benefício*

MONTHLY

YEARLY (SAVE 20%)

Hobby

Free

Includes

- ✓ Pro two-week trial
- ✓ 2000 completions
- ✓ 50 slow premium requests

DOWNLOAD

OTHERS

Pro

\$20 /month

Everything in Hobby, plus

- ✓ Unlimited completions
- ✓ 500 fast premium requests per month ?
- ✓ Unlimited slow premium requests
- ✓ 10 o1-mini uses per day

GET STARTED

Business

\$40 /user/month

Everything in Pro, plus

- ✓ Enforce privacy mode org-wide
- ✓ Centralized team billing
- ✓ Admin dashboard with usage stats
- ✓ SAML/OIDC SSO

GET STARTED

# ***Jeff Bezos API Mandate***

Anyone who doesn't do this will be fired.

## *Jeff Bezos API Mandate*

- "All teams will henceforth expose their data and functionality through service interfaces"
- "Teams must communicate with each other through these interfaces"
- "No other form of interprocess communication allowed: no direct linking, no direct reads of another team's data store, no shared-memory model, no back-doors whatsoever"
- "It doesn't matter what technology they use. HTTP, Corba, Pubsub, custom protocols—doesn't matter"
- "All service interfaces must be designed from the ground up to be externalizable"

# *APIs são Protocolos*

- Definição de Protocolo (sentido amplo)
  - conjunto **regras**, procedimentos e **convenções** que
    - ✓ determinam como uma **comunicação** ou interação deve ocorrer entre **duas** ou mais partes.
- Principais funções de um protocolo
  - Padronizar interações
  - Evitar mal-entendidos
  - Garantir que todas as partes envolvidas saibam exatamente como proceder
  - Facilitar a comunicação efetiva
  - Manter a ordem e previsibilidade nas interações



# *Exemplos de Protocolos*

- **Diplomacia**
  - Regras de etiqueta e procedimentos formais que governam como chefes de estado e diplomatas devem interagir,
    - ✓ como ordem de precedência, formas de tratamento e procedimentos em cerimônias oficiais.
- **Medicina**
  - Sequência padronizada de procedimentos médicos para diagnosticar ou tratar determinada condição,
    - ✓ garantindo que todos os profissionais sigam os mesmos passos e padrões de segurança
- **Comunicação**
  - Desde protocolos sociais (como cumprimentar alguém) até protocolos de comunicação técnica (como cartas formais ou documentos legais),
    - ✓ são convenções que estabelecem como a informação deve ser estruturada e transmitida.
- **Tecnologia**
  - Conjuntos de regras que definem como dados devem ser transmitidos entre dispositivos ou sistemas,
    - ✓ incluindo formato dos dados, sequência de mensagens, tratamento de erros e requisitos de segurança.

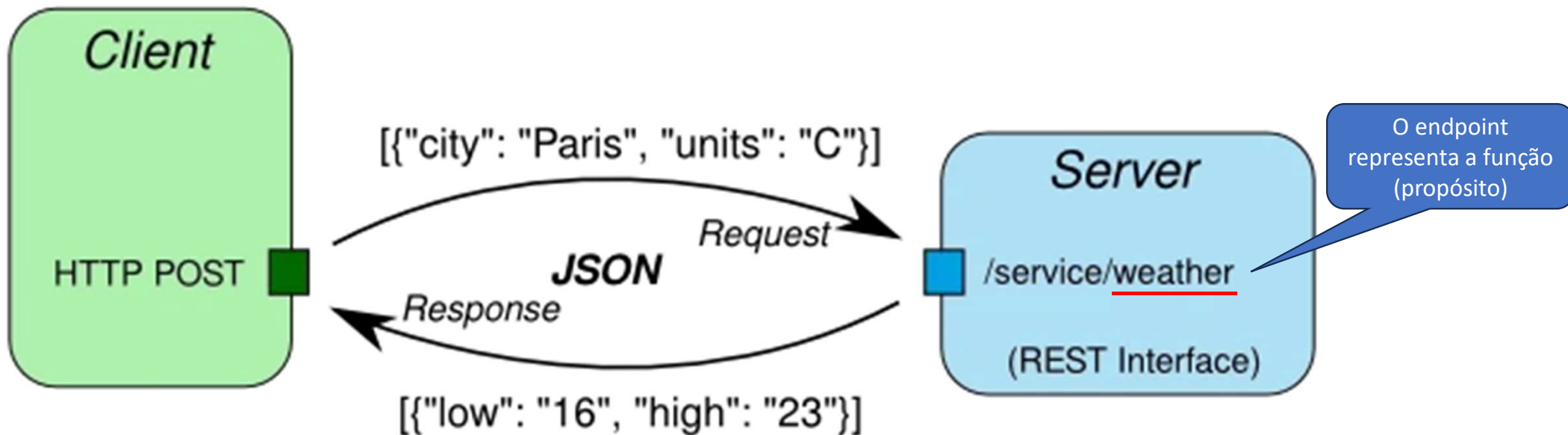


# *API “esconde” a complexidade*

- E proporciona uma comunicação direta e funcional
  - Funcional = orientada a funções (ações)
- Analogia com Restaurante
  - Garçom => API
  - Cliente => Aplicação (que acessa um dado)
  - Cozinha => Sistema (complexo)
  - Cardápio => Documentação da API
- Você faz o pedido em um formato específico
  - "Quero o prato X, ponto médio, sem cebola"
  - O garçom anota e transmite o pedido à cozinha usando a linguagem/formato que eles entendem
  - A cozinha prepara e envia o prato de volta seguindo padrões estabelecidos
  - O garçom entrega o resultado final na sua mesa

# Porque API é Orientada a Funções

**JSON / REST / HTTP**



# *API simples em Python*

```
from flask import Flask
```

```
app = Flask(__name__)
```

```
@app.route('/')
```

```
def hello_world():
```

```
    return {'message': 'Hello World'}
```

```
if __name__ == '__main__':
```

```
    app.run(debug=True)
```

## *API simples em Python (prompt)*

Crie um código da API mais simples possível de se criar em flask com apenas um endpoint que retorna um "hello world".

Quais foram as unidades básicas de conhecimento (vocabulário) essenciais nesse prompt?

## *Próximos Prompts*

- Crie um ambiente virtual do python para esse projeto
- Instale o pacote flask
- Crie um comando curl pra testar a API
  - `curl http://localhost:5000/`

```
PS C:\Users\alex_\PycharmProjects\mba_enap> curl http://localhost:5000/

StatusCode      : 200
StatusDescription : OK
Content         : {
                  "message": "Hello World"
                }
```

# *Algumas soluções para disponibilizar uma API*

- Python/Flask (ou FastAPI) hospedado na nuvem
  - 3 linhas de código e 10min para fazer o deploy (implantação)
    - ✓ Demanda mais tempo e conhecimento
      - Você precisa gerir a base de dados que a API consulta/escreve
    - ✓ Proporciona mais flexibilidade para requisitos específicos
- Soluções em plataformas Low Code (Fastgen, Linx)
- Serverless function (Google Cloud Function e AWS Lambda)
- Application Builder (com acesso via API)
  - **Airtable**
    - ✓ Cria aplicações razoavelmente simples (no code) com as quais você pode interagir via API

# *Criar uma Google Cloud Function*

- Prompt

- crie o código de uma google cloud function e me ajude a implanta-la na nuvem do google
  - ✓ Baixe e instale o [Google Cloud SDK](#)
  - ✓ Execute gcloud init para inicializar e fazer login



# *Gastos com Google Cloud Functions nessa disciplina*

January 21 – February 5, 2025 (total cost) ?

R\$0.59

includes -R\$11.82 in credits

— ?

R\$0.58 over January 5 – 20, 2025

January 21 – February 5, 2025 (forecasted total cost) ?

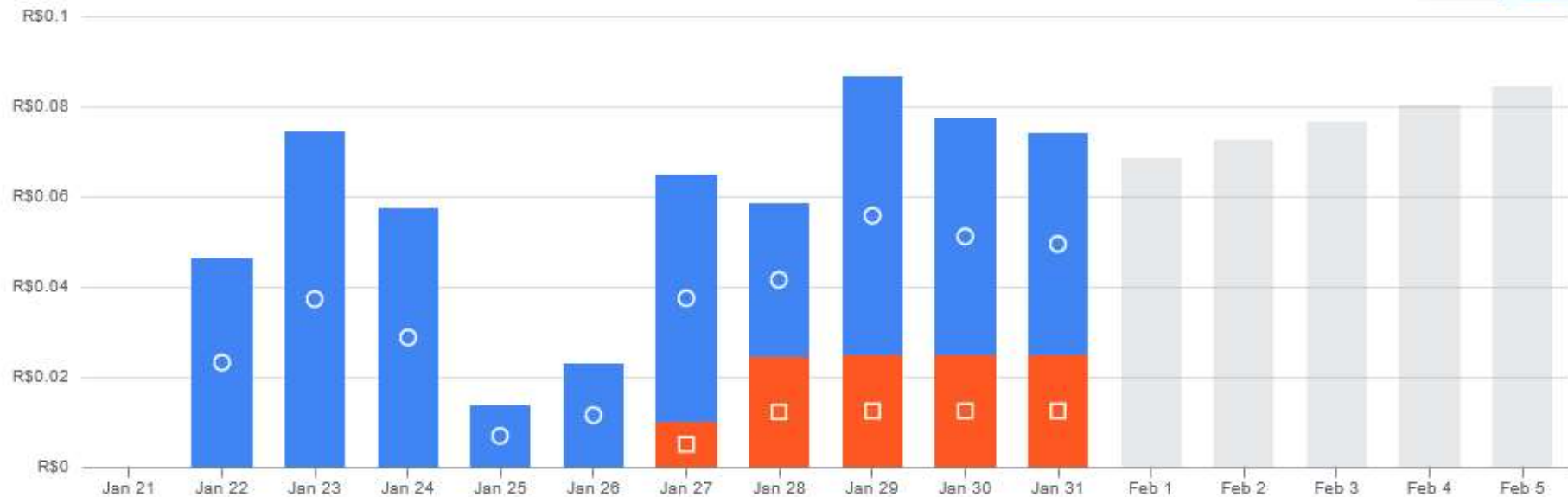
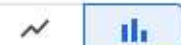
R\$0.90

includes -R\$16.72 in credits

— ?

R\$0.90 over January 5 – 20, 2025

☐ Show cumulative



# Criar uma Google Cloud Function

- Prompt

- crie o código de uma google cloud function e me ajude a implanta-la na nuvem do google

*main.py*

```
from flask import jsonify
def hello_world(request):
    return jsonify({'message': 'Hello World'})
```

-----

requirements.txt

```
flask>=2.0.0
```

-----

```
$> gcloud functions deploy hello_world --runtime python312 --trigger-http --allow-unauthenticated
```

```
$ curl https://us-central1-autograde-314802.cloudfunctions.net/hello_world
{"message":"Hello World"}
```

# *Acessando a API de uma aplicação do Airtable*

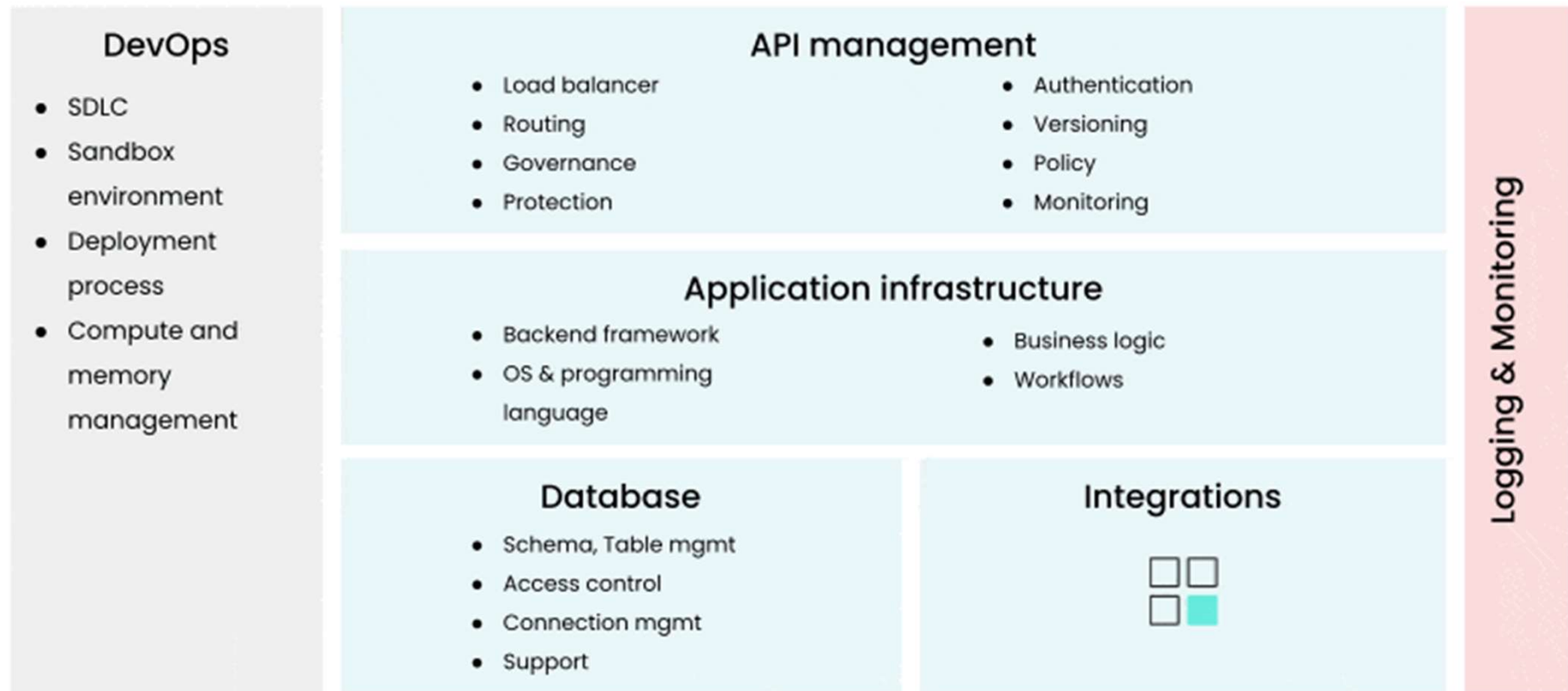
- Prompt

- create a very simple code to read data from airtable API
  - ✓ Criou o Código python
  - ✓ Criou um arquivo de variáveis de ambiente (.env)
    - Para guardar o token da API de forma Segura
  - ✓ Recomendou os pacotes de bibliotecas python
    - pip install pyairtable python-dotenv
  - ✓ Ensinou como descobrir o Table Name e o Base ID
- Tive dificuldade com a versão do pacote da biblioteca pyairtable
  - ✓ Que a perplexity ajudou a resolver recomendando a versão 1.5

- Código fonte

# *Escopo de Desenv. e Manutenção de API*

Total effort to release and maintain an API



## *Zapier (Automação de Workflow)*

- Integra mais de 7mil aplicações web e serviços (APIs)
  - Você conecta ela em fluxos razoavelmente complexos
  - No Code
  - Serverless
- Implementa fluxos de transformação digital
- Pode criar fluxos com prompts de IA
- Executa uma etapa especificada em Python (ou javascript)

## *Exemplo de Fluxo com Zapier (1)*

- Preencher um formulário num site
- Entrar numa fila de backoffice ([Pipefy](#), por exemplo)
  - Automatizada ou não
- Gerar um PDF com dados pessoais preenchidos
  - Exemplo: um contrato ou um termo de ciência
- Enviar PDF para coletar assinatura dos envolvidos
  - Envio de email e coleta de assinatura com docusign

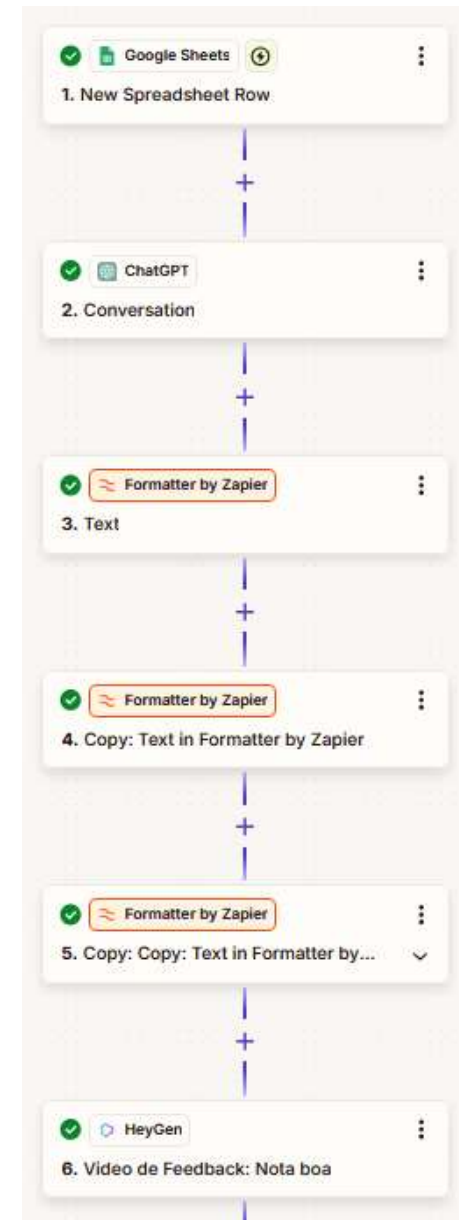
## *Exemplo de Fluxo com Zapier (2)*

- Coletar dados de uma interação online de um usuário
- Criar roteiro de vídeo com o ChatGPT
  - Comunicação personalizada (hiper-personalização)
- Gerar vídeo com avatar
  - Heygen ou Synthesia
- Mostrar para o usuário
  - Ou enviar por email



## Exemplo de Fluxo com Zapier (2)

- Heygen
  - 10 Créditos/mês no plano Free
- Synthesia
  - Sem plano gratuito para API
- Resultado final
  - Vídeo



## *Exercício 10.1 (5 pontos)*

- Crie uma Google Cloud Function
  - Que receba um prompt textual;
  - Envie para um LLM de texto (como o ChatGPT, Gemini ou Claude); e
    - ✓ O Google oferece para a API do Gemini 60 requisições por minuto gratuitamente
  - Retorne o resultado do prompt num formato json.
- Formulário de Entrega
  - Um URL no Github para o seu código fonte
  - Uma URL de Cloud Function sem autenticação
  - Uma explicação sobre o que perguntar para a função que você desenvolveu.

## *Alguns prompts utilizados para criar essa disciplina*

- Repositórios do Autograder: [1](#) e [2](#) (open source)
- Quote and cite the main AI researchers about
  - whether people should or not learn how to program computers
- Create python code for an API with rate limit
  - Create all the dependencies that I need to **deploy** to digital ocean.
    - ✓ Outro prompt incluindo: rate limit por endereço IP
- How to specify a testcase in text and then instantiate a python code that checks whether the implementation offered by a student is correct or not.
  - ok. this looks nice. but now I want something else. I want both the function name and the implementation specified as text. So that students send me their implementation and my automation code test them against a set of test cases.

## *Alguns prompts utilizados para criar essa disciplina*

- How to specify a testcase in text and then instantiate a python code that checks whether the implementation offered by a student is correct or not.
  - ok. this looks nice. but now I want something else. I want both the function name and the implementation specified as text. So that students send me their implementation and my automation code test them against a set of test cases.
    - ✓ Execute the Code: Use `exec()` to dynamically define the function.
      - `exec(full_code, {})`
- Implement a flask api endpoint that receives such a json with student's answer for validation and calls this `test_student_code` function

## *Alguns prompts utilizados para criar essa disciplina*

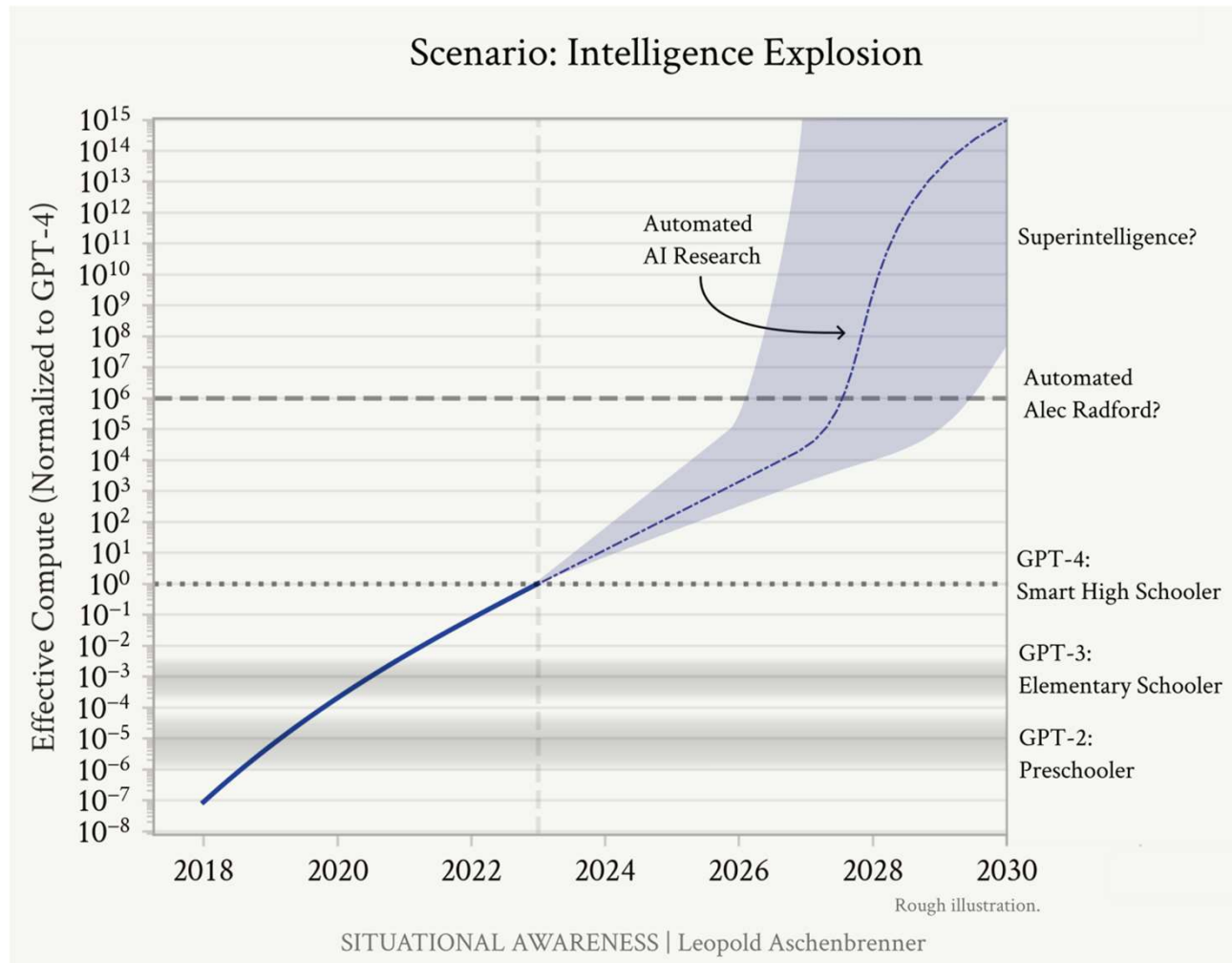
- I want you to adapt the code so that each test case is instantiated regarding its function id as specified in the test\_cases list. Consider this example:

```
"""test_cases = [  
    {"input": (3, 5), "expected": 8, id: "AAA"},  
    {"input": (10, 20), "expected": 30, id: "AAA"},  
    {"input": (-1, 1), "expected": 0, id: "AAA"},  
]"""
```

## *Alguns prompts utilizados para criar essa disciplina*

- I'm going to run user code in my container... I'd like to perform safety and security tests prior to execution of user code. Could you help me this checking procedure?
  - Gerou um Código que implementou 3 tipos distintos de medidas de segurança.
- Are there any web service that I can send a code to and it runs a python code for me and returns the output?
  - Sugeriu 6 opções e seus trade-offs.
    - ✓ Escolhi o Google Cloud Functions
      - Confiável, barato, escalável

# ***SITUATIONAL AWARENESS (fonte) - Leopold Aschenbrenner***





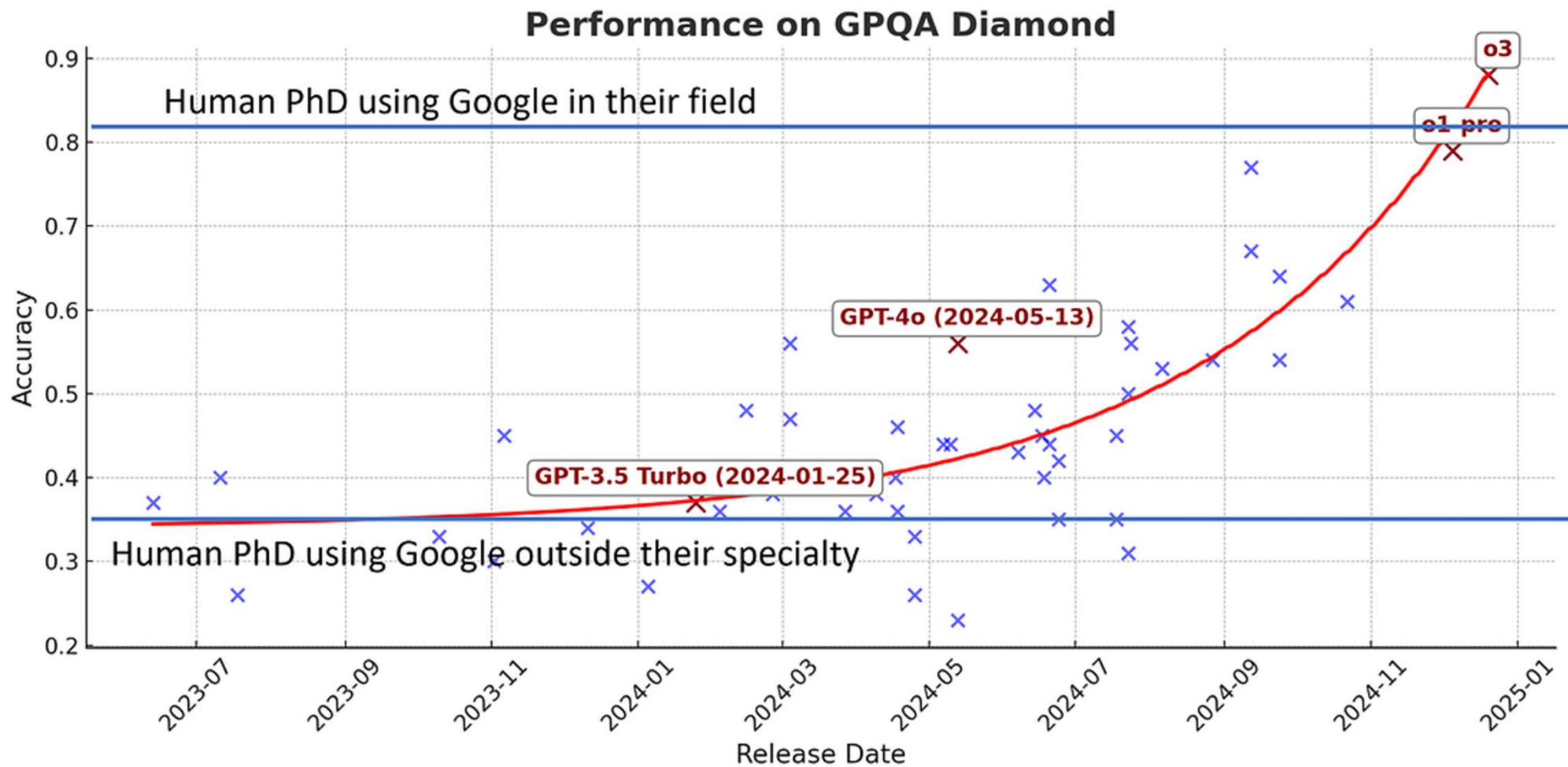
# *Capital will matter more than ever after AGI*

- Substituição do trabalho humano pelo capital:
  - A AGI permitirá que o capital (como centros de dados executando software) substitua amplamente o trabalho humano, reduzindo a necessidade de mão de obra humana.
- Aumento do poder do capital:
  - Com a substituição do trabalho humano, a capacidade de adquirir resultados no mundo real através do capital aumentará significativamente.
- Diminuição do poder humano sem capital:
  - A capacidade dos humanos de exercer influência no mundo real diminuirá drasticamente sem acesso ao capital, pois:
    - ✓ Instituições como estados e empresas terão menos incentivos para se preocuparem com os humanos.
    - ✓ Será mais difícil para os indivíduos alcançarem resultados excepcionais sem recursos iniciais significativos.
- Improvável implementação de medidas igualitárias radicais:
  - Mudanças radicais para equalizar a distribuição de riqueza são improváveis, o que pode levar à ampliação e perpetuação de desequilíbrios de poder existentes.

## *Capital will matter more than ever after AGI*

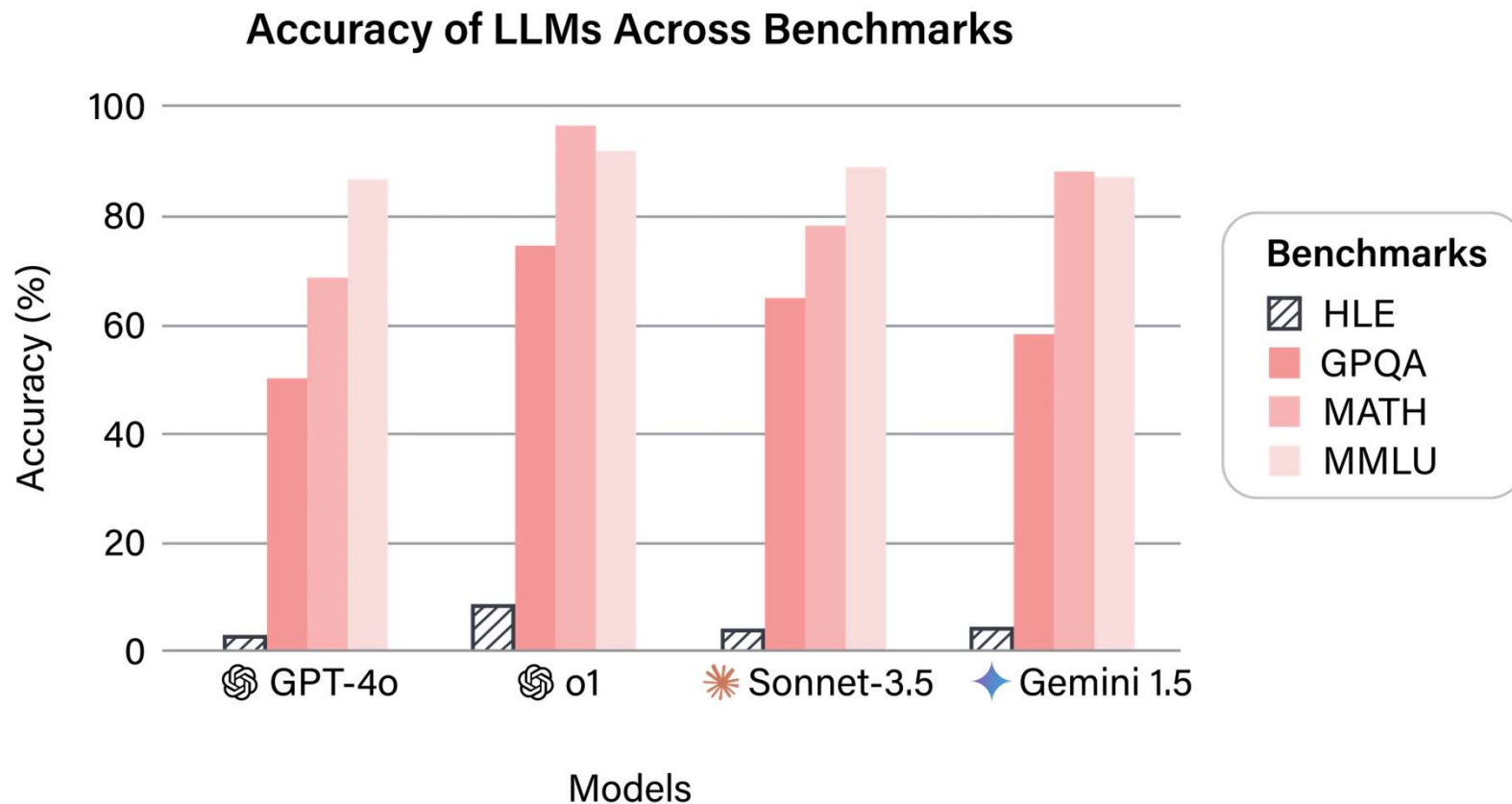
- Possível estagnação social
  - A sociedade pode se tornar permanentemente estática, com as atuais disparidades de poder sendo amplificadas e tornando-se imutáveis.
- Risco de desvalorização do trabalho humano
  - Mesmo que medidas como a Renda Básica Universal (UBI) sejam implementadas para garantir o conforto material, a desvalorização do trabalho humano pode reduzir a relevância e a agência dos indivíduos na sociedade.
- Importância contínua do dinheiro
  - Embora algumas pessoas acreditem que o dinheiro perderá relevância após a AGI, o artigo argumenta que ele continuará sendo uma ferramenta essencial para a alocação de recursos, a menos que uma única entidade de IA controle toda a economia.

# Melhoria exponencial



## Humanity's Last Exam (HLE)

- Benchmarks anteriores não são mais apropriados



## *Humanity's Last Exam (HLE)*

- HLE foi lançado em
  - 20 de Janeiro 2025
- Nova ferramenta da OpenAI: Deep Research
  - Usa o modelo o3
- Histórico do desempenho recente
  - Deep Research: 26.6% accuracy
  - o3-mini-high: 13% accuracy
  - o3-mini: 10.5% accuracy
  - o1: 8.3% accuracy<sup>1</sup>

***Pagar ou não pagar por um modelo  
de IA?***