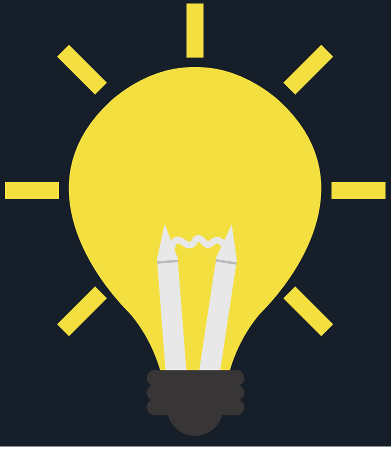
Production LLM Deployment

DS Pipelines in Minutes: An Example...

PDF Summary Pipeline

Using Seaplane and GPT3.5



seaplane

What is Seaplane

Don’t believe us, believe them!

"A straightforward platform for data science deployments without getting caught up in configuration tasks.A data scientist's dream 🤩"

Kjeld Oostra AI Engineer @ Entropical

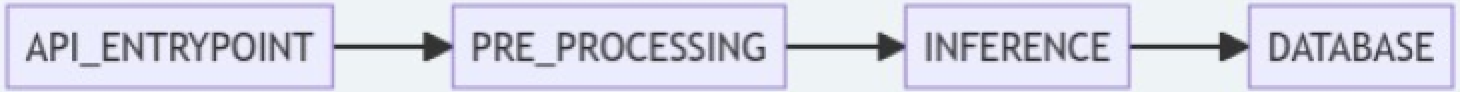
A powerful Python SDK to take your data science experiments to production in minutes without having to wrangle any cloud infrastructure.

Seaplane is jam packed with functionality to enable data scientist to do their best work. Such as built-in SQL, Vector-Store, Object Store, Large models and much more!

PDF Summary Application

- An API entry point available at /demo-input
- A pre-processing step to download, extract the text from our PDF and create the prompt for GPT-3.5
- An inference step to create the summary using GPT-3.5
- A database step to store the result in Seaplane's managed global SQL database

Directed Acyclic Graph



How to build an app on Seaplane

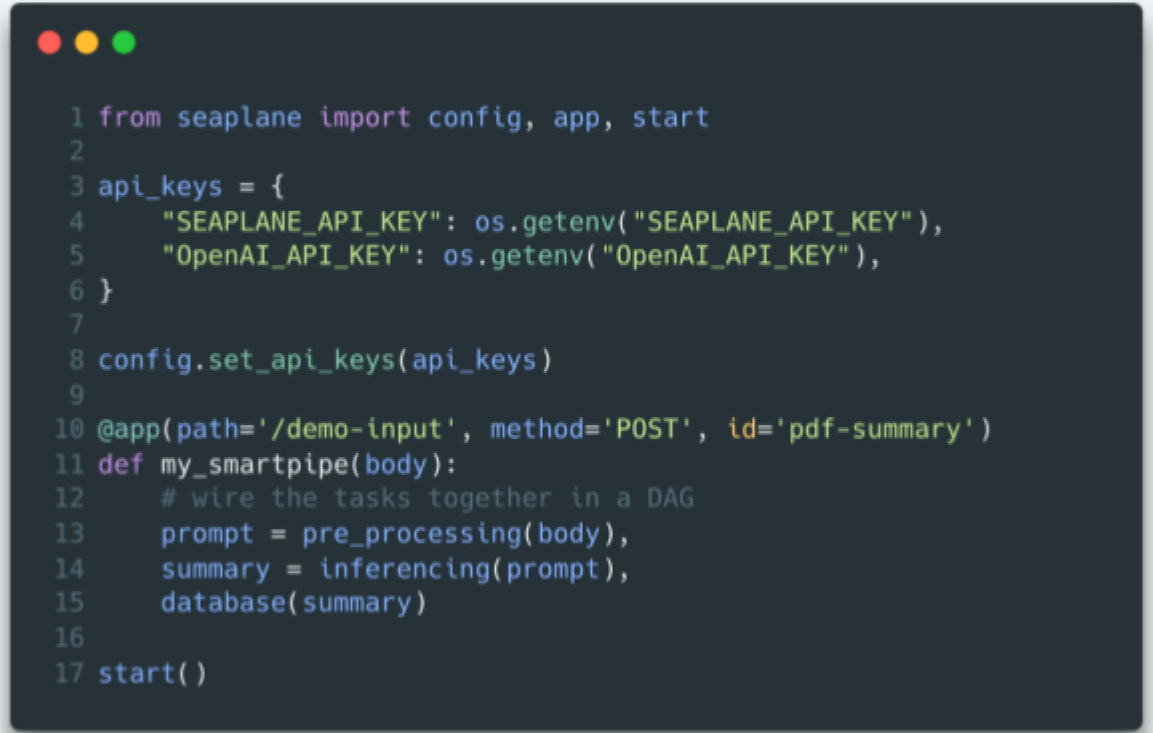
- Applications @app - HTTP, Kafka, Cron job entry points
- Tasks @task - Individually scalable containers
- Task Resources
    - SQL
    - Vector-DB
    - Object Store
    - LLMs
    - Much more!

Let’s Start Building

seaplane

The App

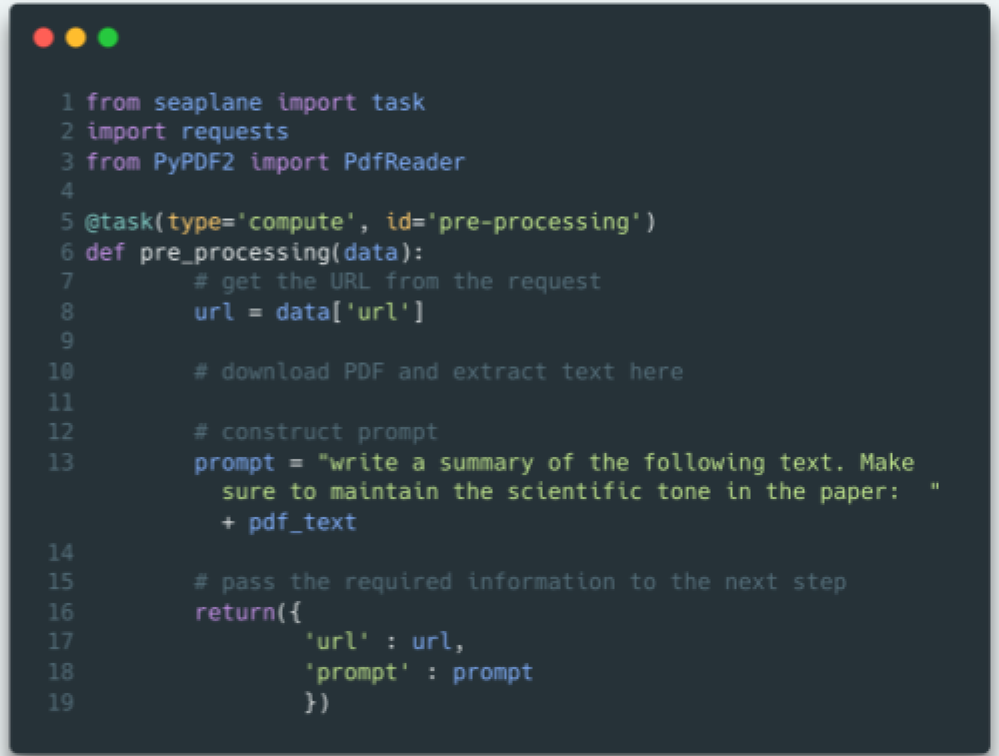
- Creates the HTTP endpoint
- Creates the DAG i.e the flow of the data
- Starts the application



```python
from seaplane import config, app, start

api_keys = {
    "SEAPLANE_API_KEY": os.getenv("SEAPLANE_API_KEY"),
    "OpenAI_API_KEY": os.getenv("OpenAI_API_KEY"),
}

config.set_api_keys(api_keys)

@app(path='/demo-input', method='POST', id='pdf-summary')
def my_smartpipe(body):
    # wire the tasks together in a DAG
    prompt = pre_processing(body),
    summary = inferencing(prompt),
    database(summary)

start()
```

Preprocessing @task

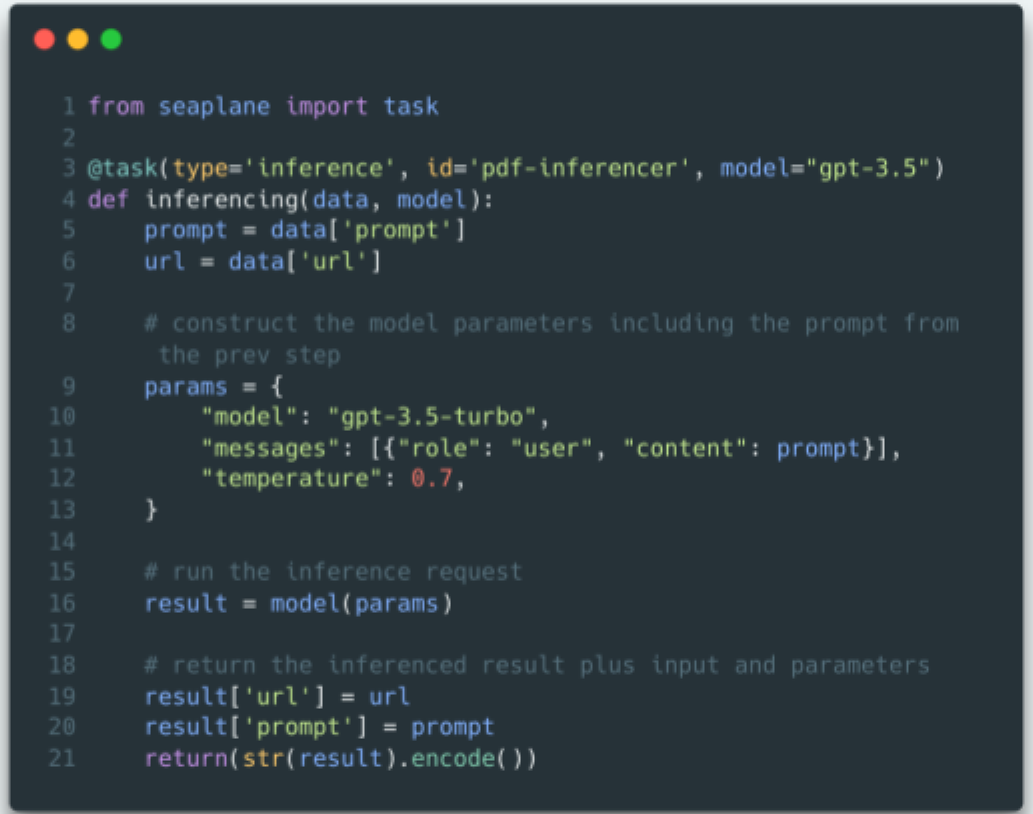
- Downloads PDF
- Extract Text
- Create Prompt



```python
from seaplane import task
import requests
from PyPDF2 import PdfReader

@task(type='compute', id='pre-processing')
def pre_processing(data):
        # get the URL from the request
        url = data['url']

        # download PDF and extract text here

        # construct prompt
        prompt = "write a summary of the following text. Make
          sure to maintain the scientific tone in the paper:  "
          + pdf_text

        # pass the required information to the next step
        return({
                'url' : url,
                'prompt' : prompt
                })
```

Inference @task

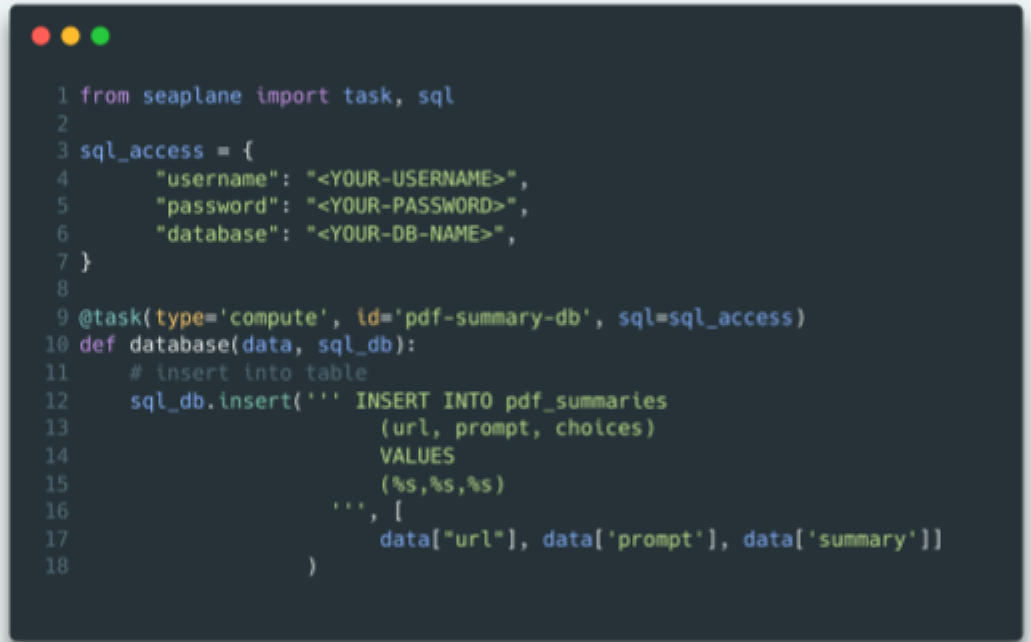
- Runs the inference
- OpenAI and other LLMs available through the SDK
- Local models such as MPT-30B, Bloom, Bloomz - Shared resource hosted by Seaplane.



```python
from seaplane import task

@task(type='inference', id='pdf-inferencer', model="gpt-3.5")
def inferencing(data, model):
    prompt = data['prompt']
    url = data['url']

    # construct the model parameters including the prompt from
     the prev step
    params = {
        "model": "gpt-3.5-turbo",
        "messages": [{"role": "user", "content": prompt}],
        "temperature": 0.7,
    }

    # run the inference request
    result = model(params)

    # return the inferenced result plus input and parameters
    result['url'] = url
    result['prompt'] = prompt
    return(str(result).encode())
```

Database @task

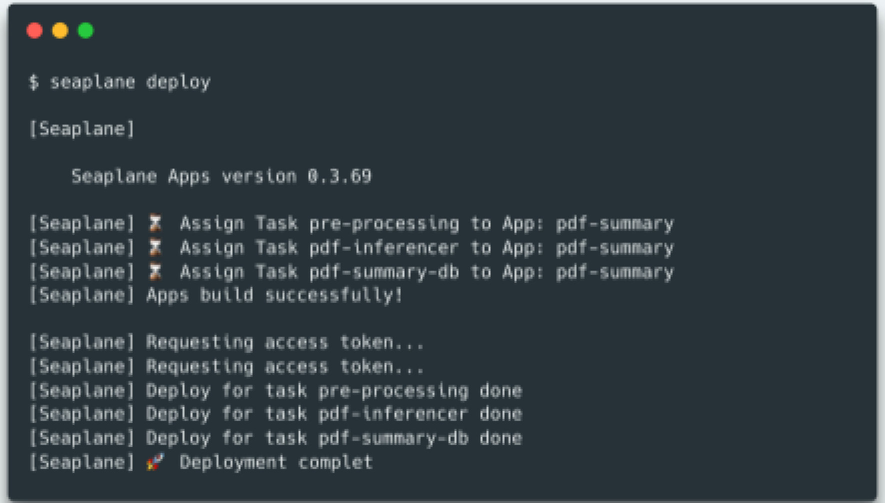
- Stores the result in Seaplane SQL
- DB connection managed by SDK



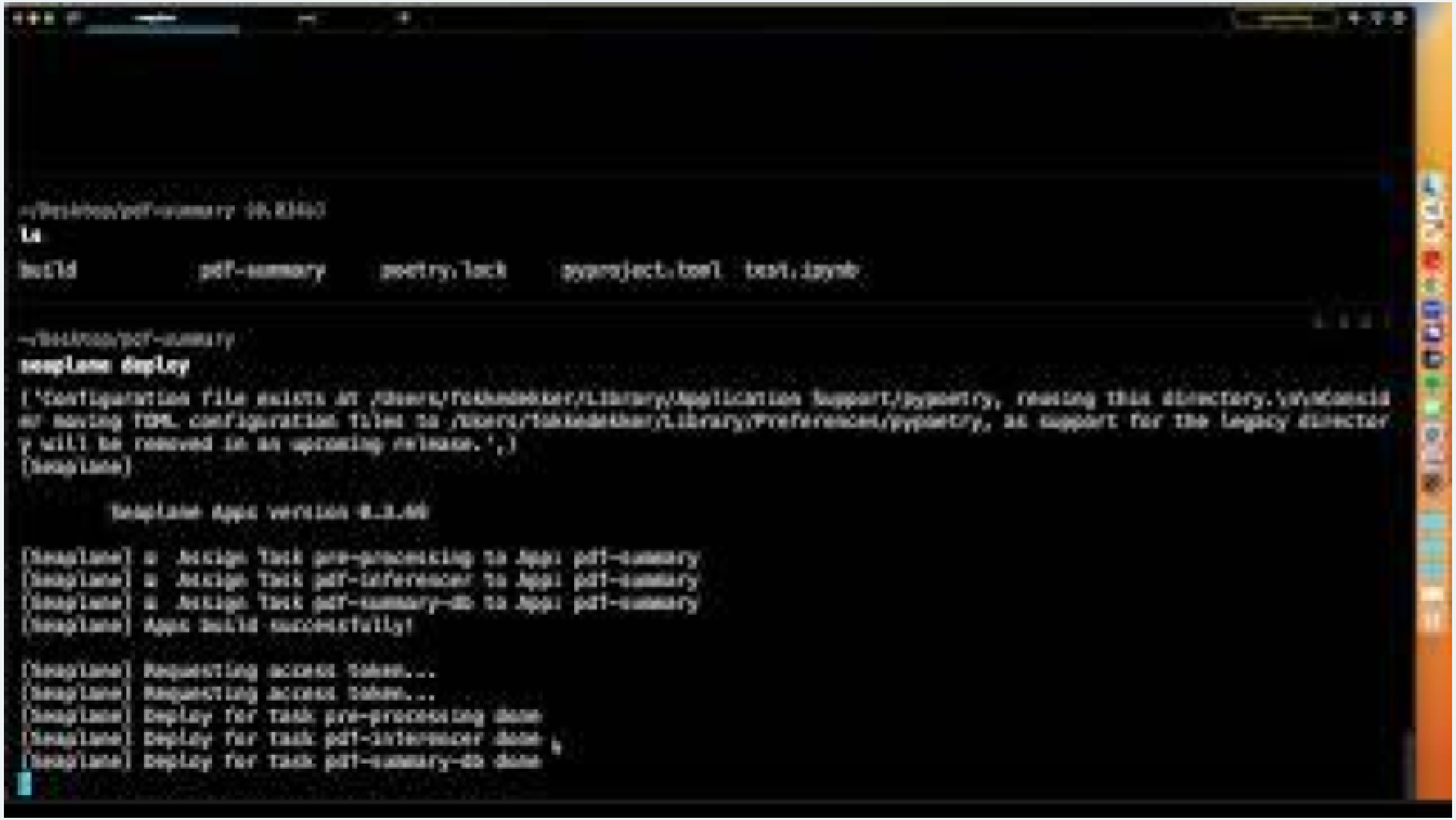
```python
from seaplane import task, sql

sql_access = {
        "username": "<YOUR-USERNAME>",
        "password": "<YOUR-PASSWORD>",
        "database": "<YOUR-DB-NAME>",
}

@task(type='compute', id='pdf-summary-db', sql=sql_access)
def database(data, sql_db):
    # insert into table
    sql_db.insert(''' INSERT INTO pdf_summaries
                        (url, prompt, choices)
                        VALUES
                        (%s,%s,%s)
                    ''', [
                        data["url"], data['prompt'], data['summary']]
                 )
```

Deploying

- Set up API gateway
- Deployed 3 containers
- Wired them all together
- Set up and configured database
- Scalable serverless multi-cloud infrastructure



```
$ seaplane deploy

[Seaplane]

    Seaplane Apps version 0.3.69

[Seaplane] ⏳ Assign Task pre-processing to App: pdf-summary
[Seaplane] ⏳ Assign Task pdf-inferencer to App: pdf-summary
[Seaplane] ⏳ Assign Task pdf-summary-db to App: pdf-summary
[Seaplane] Apps build successfully!

[Seaplane] Requesting access token...
[Seaplane] Requesting access token...
[Seaplane] Deploy for task pre-processing done
[Seaplane] Deploy for task pdf-inferencer done
[Seaplane] Deploy for task pdf-summary-db done
[Seaplane] 🚀 Deployment complet
```



```
~/Desktop/pdf-summary [illegible]
ls
build           pdf-summary     poetry.lock     pyproject.toml  test.ipynb

~/Desktop/pdf-summary
seaplane deploy
('Configuration file exists at /Users/[illegible]/Library/Application Support/pypoetry, reusing this directory.\n\nConsid
er moving TOML configuration files to /Users/[illegible]/Library/Preferences/pypoetry, as support for the legacy director
y will be removed in an upcoming release.',)
[Seaplane]

        Seaplane Apps version [illegible]

[Seaplane] ⚙  Assign Task pre-processing to App: pdf-summary
[Seaplane] ⚙  Assign Task pdf-inferencer to App: pdf-summary
[Seaplane] ⚙  Assign Task pdf-summary-db to App: pdf-summary
[Seaplane] Apps build successfully!

[Seaplane] Requesting access token...
[Seaplane] Requesting access token...
[Seaplane] Deploy for task pre-processing done
[Seaplane] Deploy for task pdf-inferencer done
[Seaplane] Deploy for task pdf-summary-db done
```

https://www.youtube.com/watch?v=M4nhkC1E2Uw

Sign up for the beta today and we will double your credits*

seaplane.io/join-the-seaplane-beta

*some restrictions apply