图数据库解谜

搞清楚什么是图数据库，它能做什么，为什么它做的好？如何学用大厂们都选择了的 Nebula Graph？

古思为

DEVELOPER ADVOCATE @ vesoft

墨天轮 线上直播

Dec. 9th, 2021

古思为

- Nebula Graph 开发者布道师
- 程序员
- 开源信徒

wey-gu

wey_gu

siwei.io

Overview

- 什么是图？图数据库?
- 为什么需要图数据库?
- 到底图数据库可以做什么?
- 为什么大厂都在用的新一代开源图数据库：Nebula Graph

墨天轮

Overview

- 什么是图？图数据库?
- 为什么需要图数据库?
- 到底图数据库可以做什么?
- 为什么大厂都在用的新一代开源图数据库：Nebula Graph

wey-gu

wey_gu

siwei.io

墨天轮

图数据库简介

什么是图？ 什么是图数据库？ 为什么我们需要一个专门的数据库？

墨天轮

什么是图?

墨天轮

"以图结构、图语义来用点、边、属性来查询、表示存储数据的数据库

什么是图数据库

wikipedia.org/wiki/graph_database

了解更多关于 什么是图数据库

为什么需要图数据库?

传统数据库

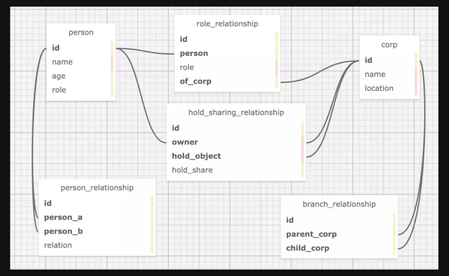
图数据库

墨天轮

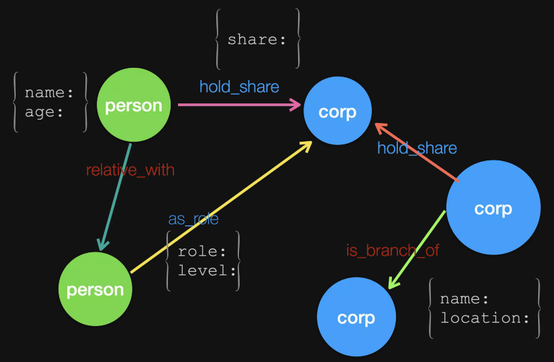
为什么需要图数据库?

传统数据库

图模型的结构



图数据库



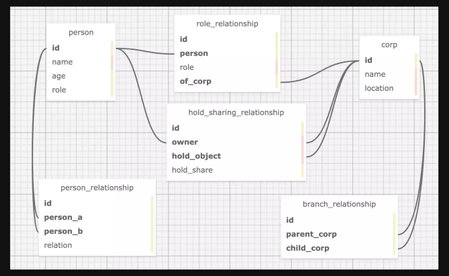
墨天轮

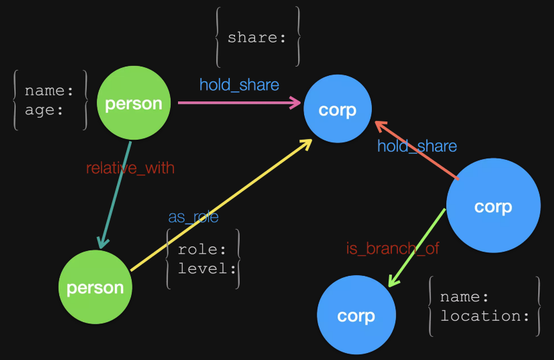
为什么需要图数据库?

传统数据库

图数据库

图模型的结构





图语义的查询

```
SELECT a.id, a.name, c.name
FROM player a
JOIN serve b ON a.id=b.player_id
JOIN team c ON c.id=b.team_id
WHERE c.name IN (SELECT c.name
FROM player a
JOIN serve b ON a.id=b.player_id
JOIN team c ON c.id=b.team_id
WHERE a.name = 'Tim Duncan')
```

```
GO FROM 100 OVER serve YIELD serve._dst AS Team | \
GO FROM $-.Team OVER serve REVERSELY YIELD $$.player.name;
```

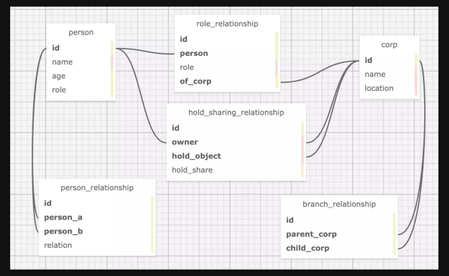
墨天轮

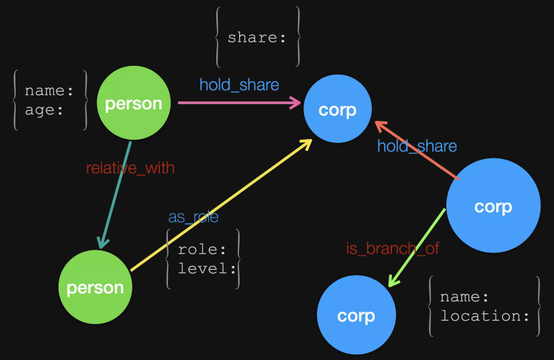
为什么需要图数据库?

传统数据库

图数据库

图模型的结构





图语义的查询

```
SELECT a.id, a.name, c.name
FROM player a
JOIN serve b ON a.id=b.player_id
JOIN team c ON c.id=b.team_id
WHERE c.name IN (SELECT c.name
FROM player a
JOIN serve b ON a.id=b.player_id
JOIN team c ON c.id=b.team_id
WHERE a.name = 'Tim Duncan')
```

```
GO FROM 100 OVER serve YIELD serve._dst AS Team | \
GO FROM $-.Team OVER serve REVERSELY YIELD $$.player.name;
```

性能

	主要应用场景	2-hop 延时 (~2.5K)	3-hop 延时 (~110K)	4-hop 延时 (~600K)
图数据库	关系遍历	0.01 秒	0.168 秒	1.36 秒
SQL数据库	信息检索	0.016 秒	30 秒	1544 秒

墨天轮

Nebula Graph!

如何发音：['nebjələ]， 它有哪些特点?

Nebula Graph 介绍

一个可靠的分布式、线性扩容、性能高效的图数据库

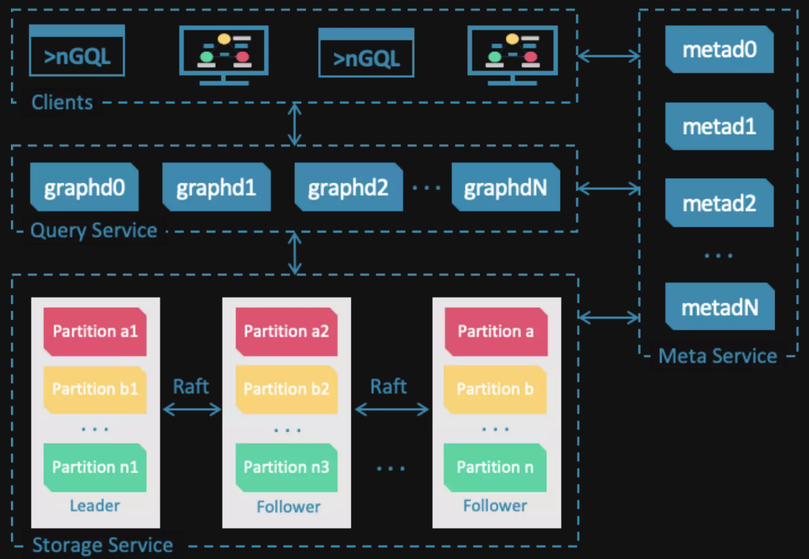
世界上唯一能够容纳千亿顶点和万亿条边，并提供毫秒级查询延时的图数据库解决方案

墨天轮

Nebula Graph 介绍

一个可靠的分布式、线性扩容、性能高效的图数据库

世界上唯一能够容纳千亿顶点和万亿条边，并提供毫秒级查询延时的图数据库解决方案

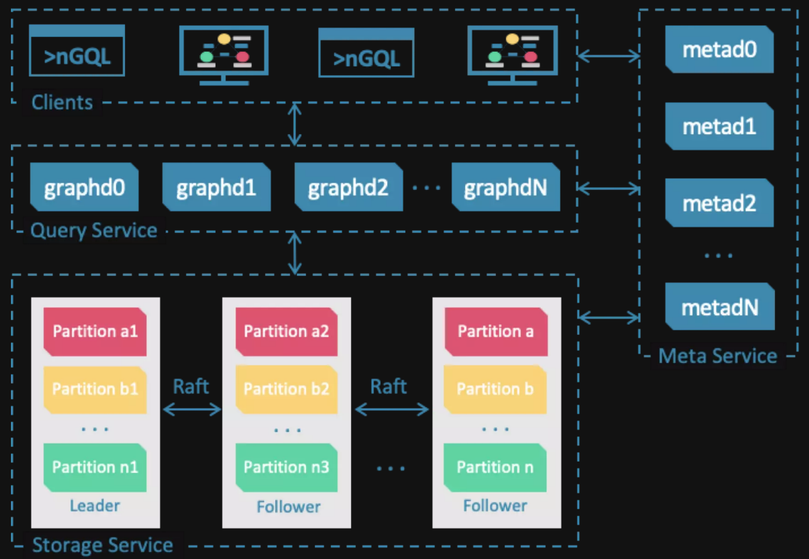


墨天轮

Nebula Graph 介绍

一个可靠的分布式、线性扩容、性能高效的图数据库

世界上唯一能够容纳千亿顶点和万亿条边，并提供毫秒级查询延时的图数据库解决方案



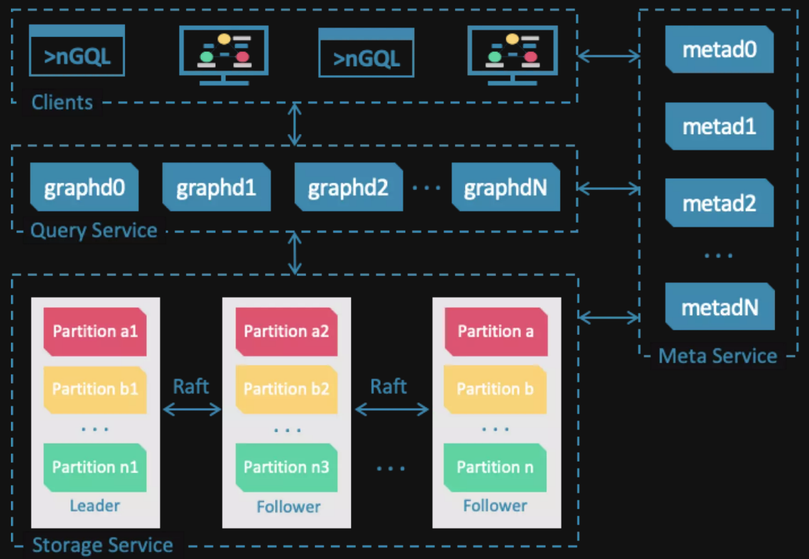
京东数科 · 美团 · 有赞 · 小红书 · WiFi万能钥匙 · Tencent 腾讯 · WeBank 微众银行 · vivo · 企查查 · 快手 · 知乎 · 360数科 · SUNING 苏宁科技 · BOSS直聘 · 网易 · Seeyii 视野金服 · 思知 · 半云科技 · 同盾科技 · 携程金融 · akulaku · 合合信息 · 中国移动 · oppo · 微信 · 众安保险 · 网易游戏 · Soul · ziroom自如 · HUAWEI

墨天轮

Nebula Graph 介绍

一个可靠的分布式、线性扩容、性能高效的图数据库

世界上唯一能够容纳千亿顶点和万亿条边，并提供毫秒级查询延时的图数据库解决方案



了解更多 >>>

文档：Nebula 架构

官网：用户案例

墨天轮

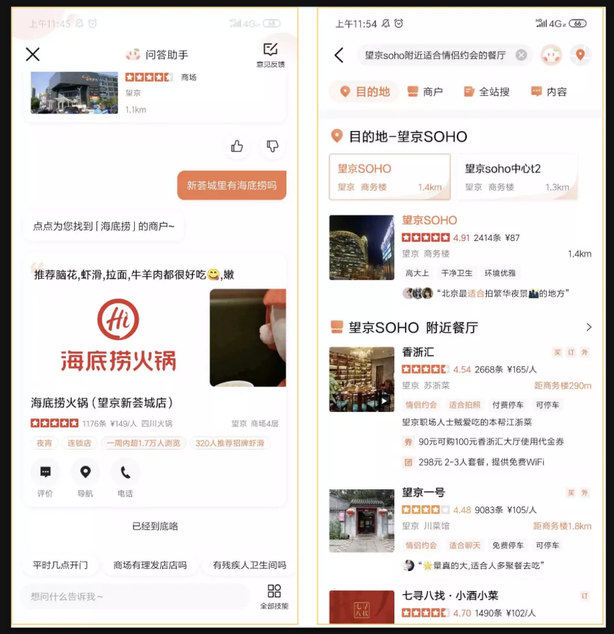
一个贴切的小 Demo

关于漫威电影宇宙（MCU）与姚明的故事

墨天轮

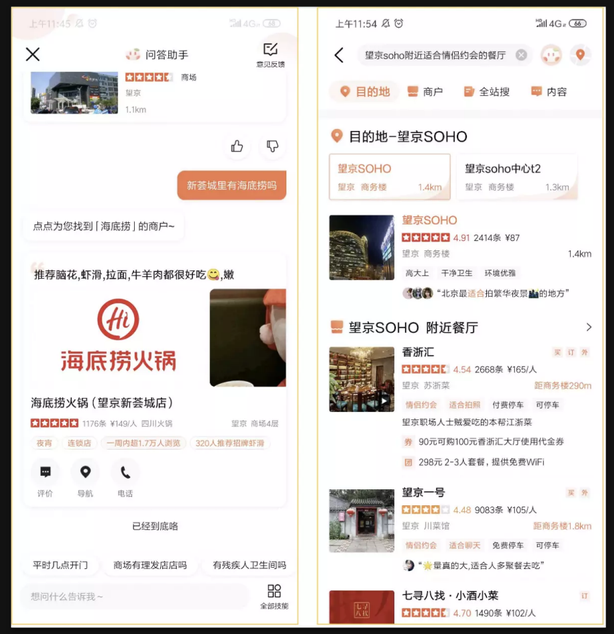
图数据库的应用场景

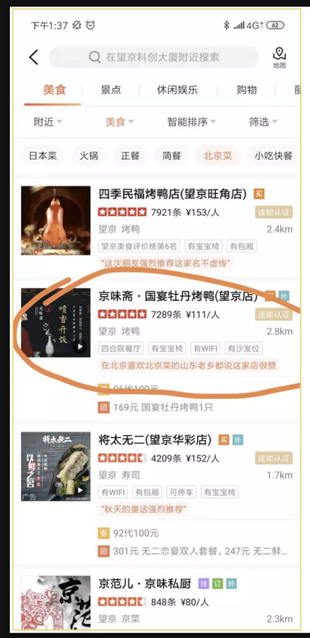
图数据库的应用场景

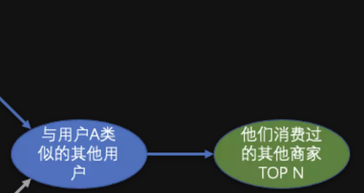


墨天轮

图数据库的应用场景

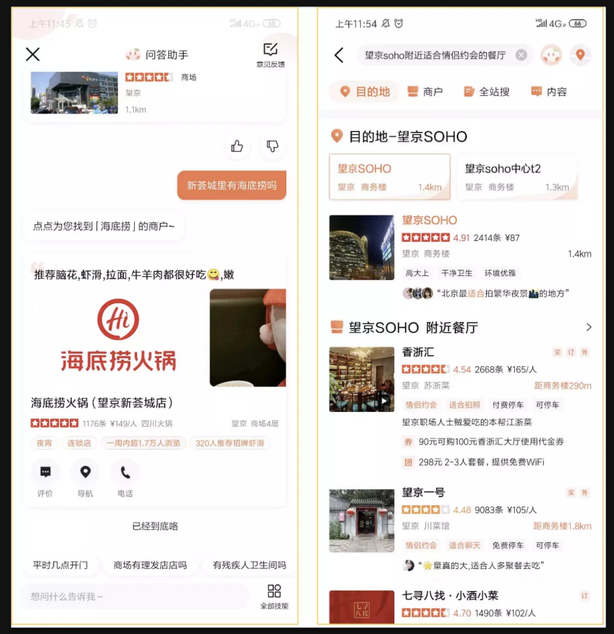


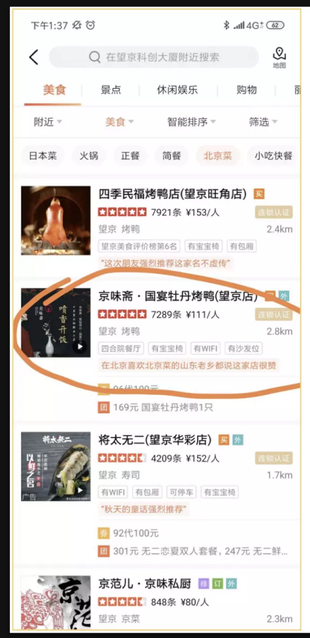


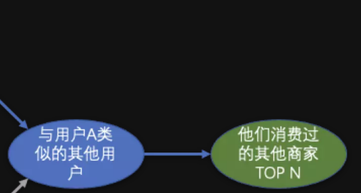


墨天轮

图数据库的应用场景



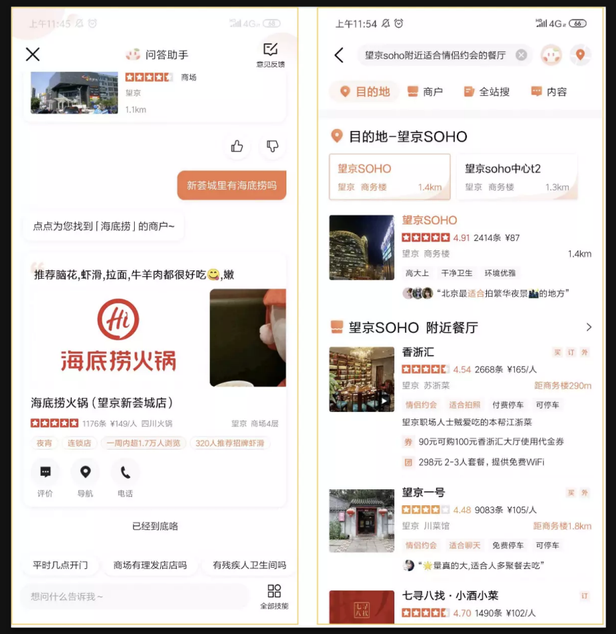




tech.meituan.com/2021/04/01/nebula-graph-practice-in-meituan.html

墨天轮

图数据库的应用场景



典型场景

社交网络(meta)	风险控制	公共安全	知识图谱	机器学习
生化制药	物联网	区块链	数据血缘	智能运维

tech.meituan.com/2021/04/01/nebula-graph-practice-in-meituan.html

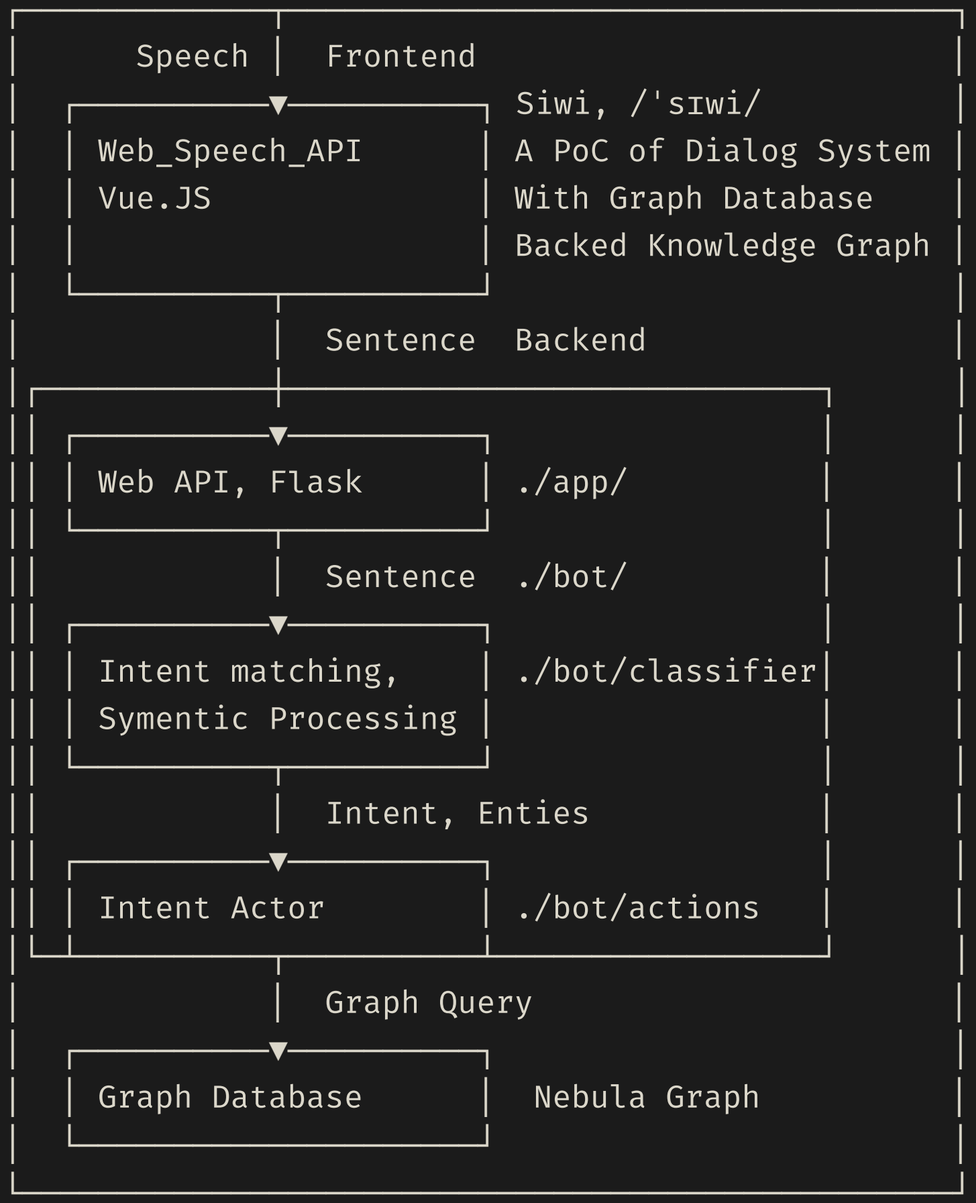
墨天轮

Siwi

Siwi (/ˈsɪwi/)，一个知识图谱驱动的智能语音助手

墨天轮

Arch

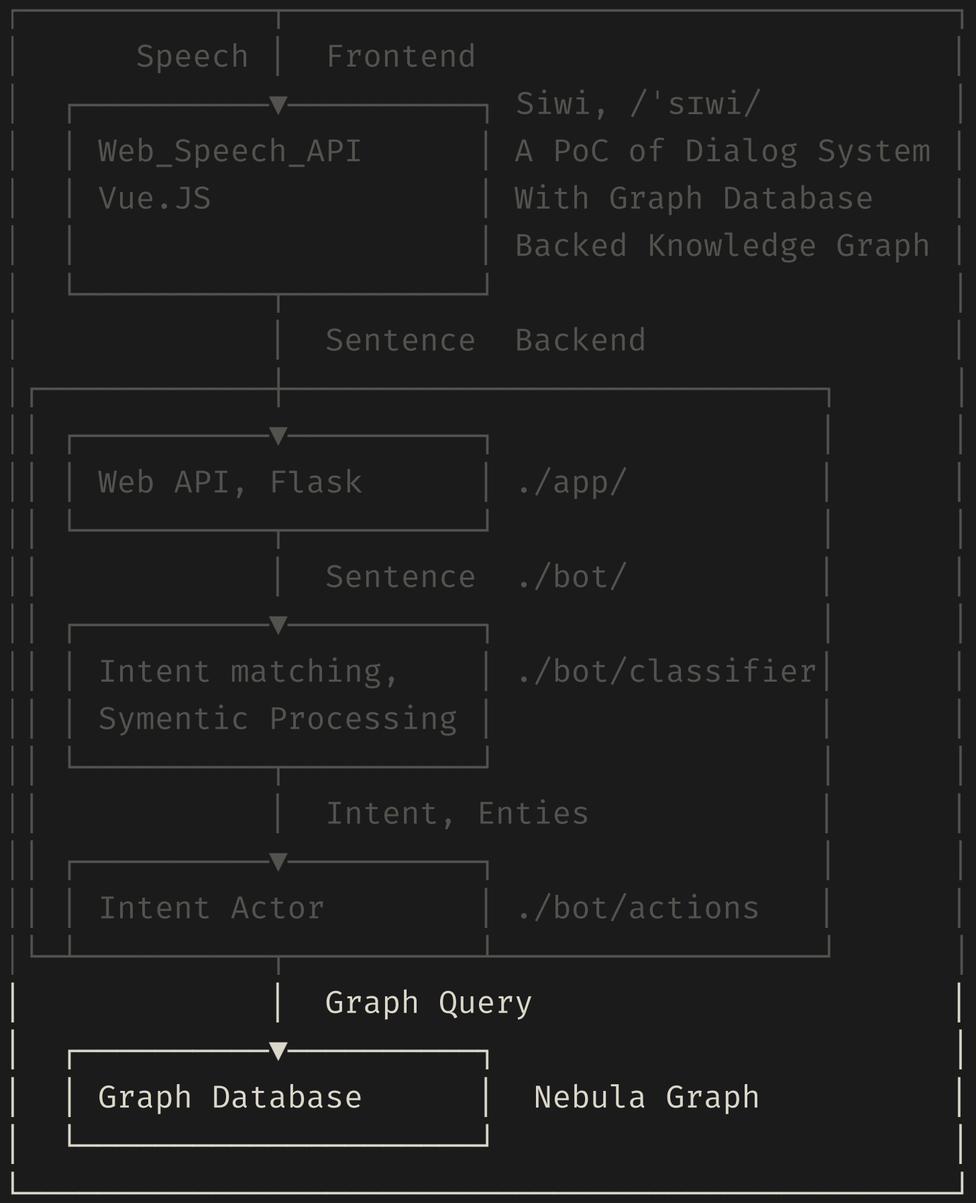


Code

```
.
├── README.md
├── src
│   ├── siwi                  # Siwi-API Backend
│   │   ├── app               # Web Server, take HTTP req > ca
│   │   └── bot               # Bot API
│   │       ├── actions       # Take Intent, Slots, Query KG h
│   │       ├── bot           # Entrypoint of the Bot API
│   │       ├── classifier    # Symentic Parse, Intent Match,
│   │       └── test          # Example Data as equivalent/moc
│   └── siwi_frontend         # Browser End
│       ├── README.md
│       ├── package.json
│       └── src
│           ├── App.vue       # Listen to user and pass Qs to
│           └── main.js
└── wsgi.py
```

wey-gu/**nebula-siwi**

Arch

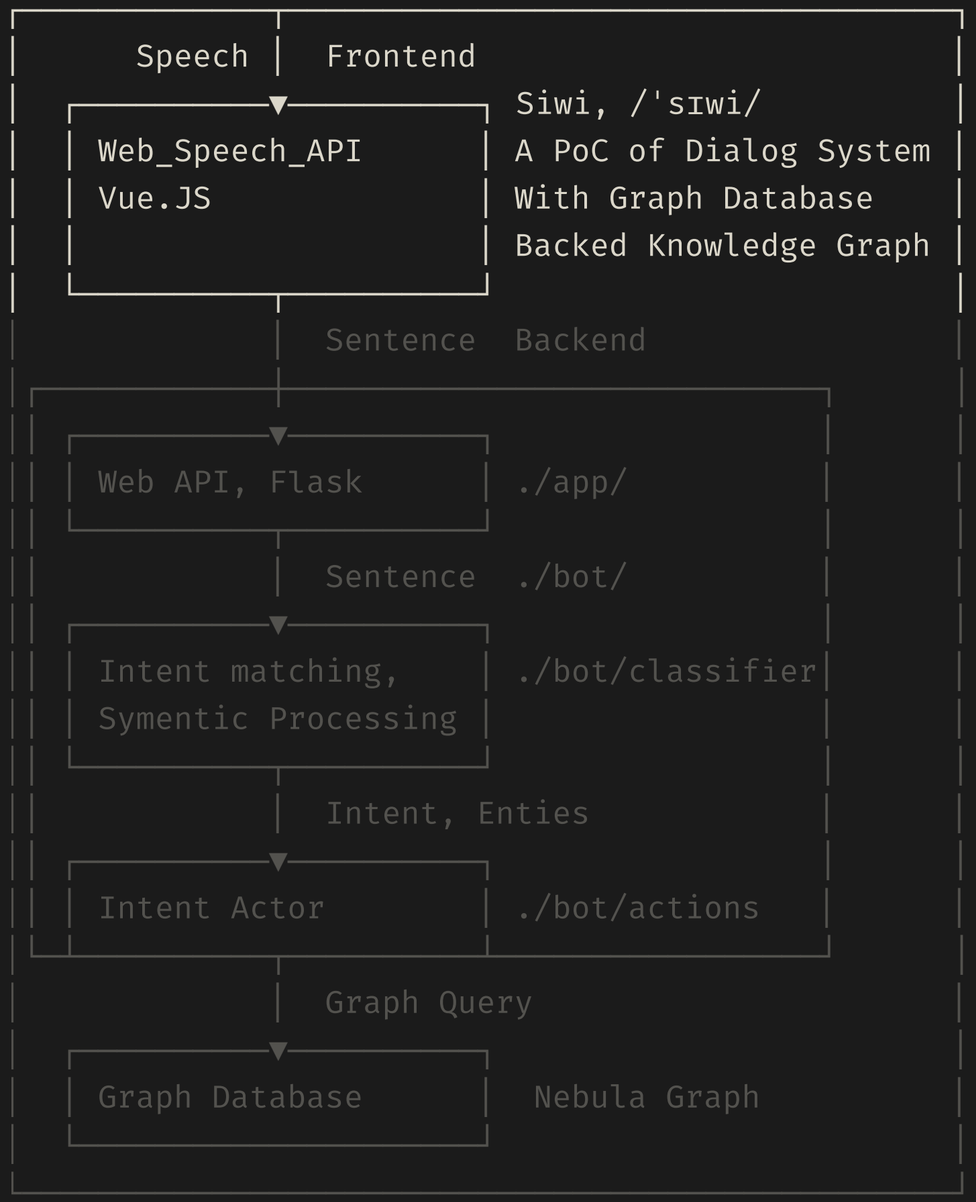


Code

```
.
├── README.md
├── src
│   ├── siwi                  # Siwi-API Backend
│   │   ├── app               # Web Server, take HTTP req > ca
│   │   └── bot               # Bot API
│   │       ├── actions       # Take Intent, Slots, Query KG h
│   │       ├── bot           # Entrypoint of the Bot API
│   │       ├── classifier    # Symentic Parse, Intent Match,
│   │       └── test          # Example Data as equivalent/moc
│   └── siwi_frontend         # Browser End
│       ├── README.md
│       ├── package.json
│       └── src
│           ├── App.vue       # Listen to user and pass Qs to
│           └── main.js
└── wsgi.py
```

wey-gu/**nebula-siwi**

Arch

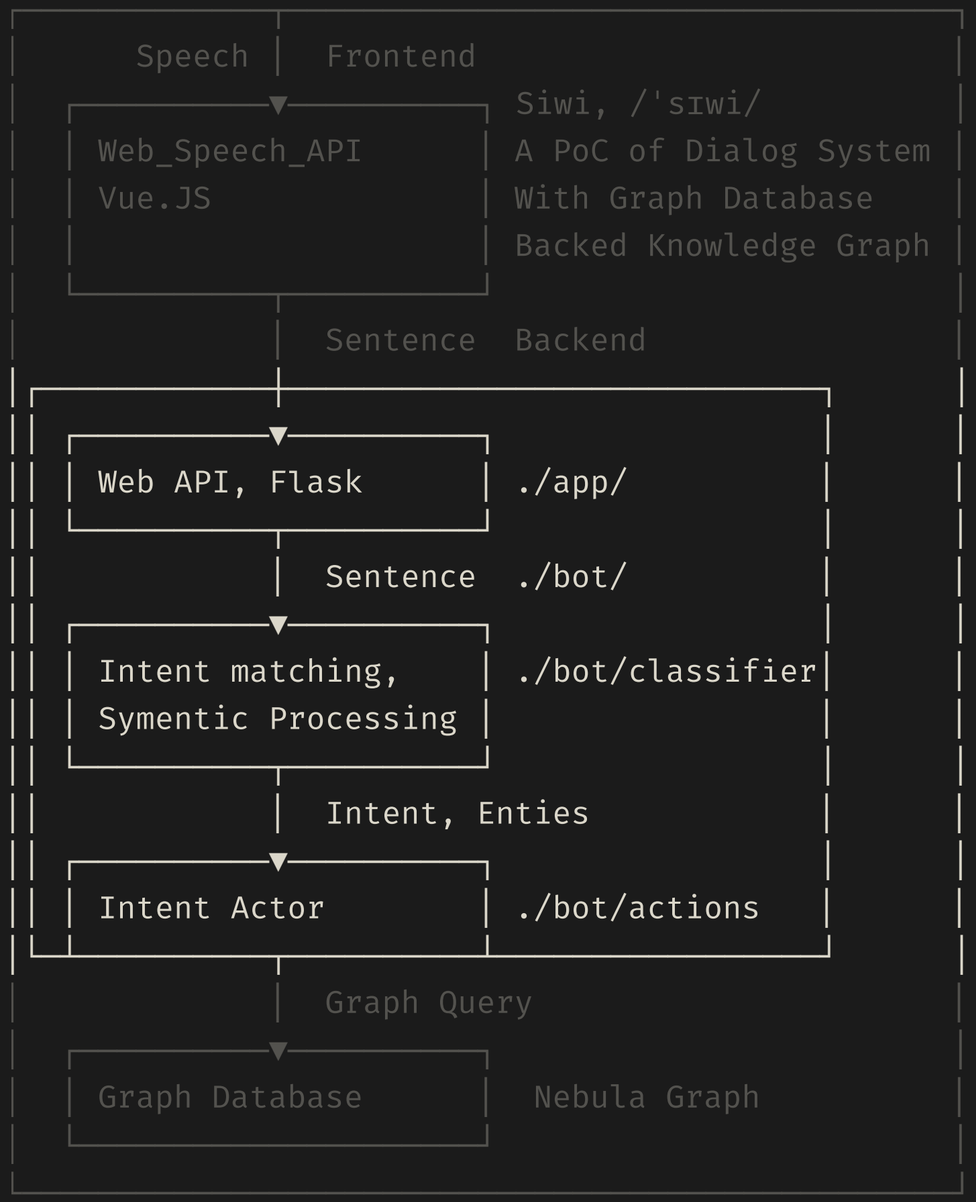


Code

```
.
├── README.md
├── src
│   ├── siwi                  # Siwi-API Backend
│   │   ├── app               # Web Server, take HTTP req > ca
│   │   └── bot               # Bot API
│   │       ├── actions       # Take Intent, Slots, Query KG h
│   │       ├── bot           # Entrypoint of the Bot API
│   │       ├── classifier    # Symentic Parse, Intent Match,
│   │       └── test          # Example Data as equivalent/moc
│   └── siwi_frontend         # Browser End
│       ├── README.md
│       ├── package.json
│       └── src
│           ├── App.vue       # Listen to user and pass Qs to
│           └── main.js
└── wsgi.py
```

wey-gu/**nebula-siwi**

Arch

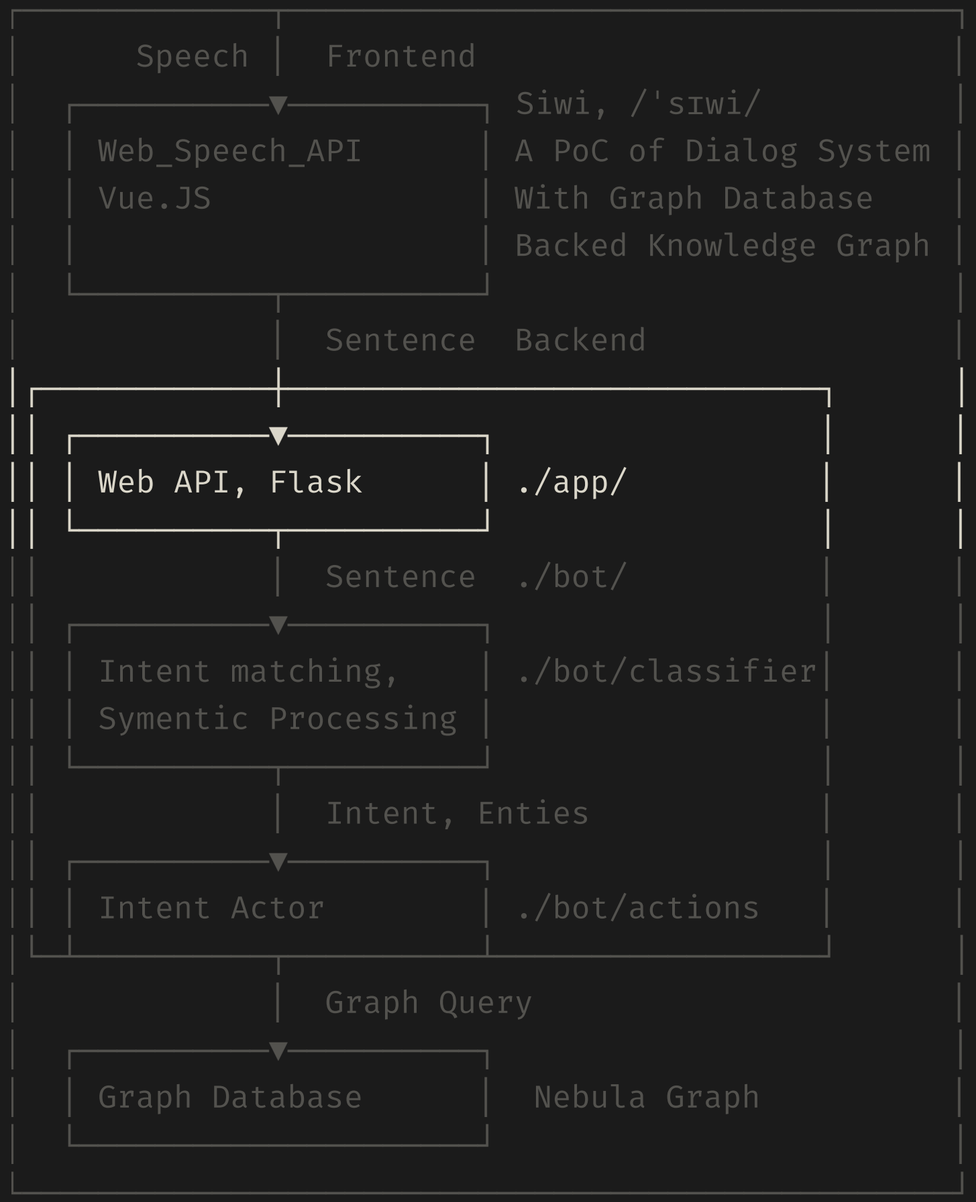


Code

```
.
├── README.md
├── src
│   ├── siwi                  # Siwi-API Backend
│   │   ├── app               # Web Server, take HTTP req > ca
│   │   └── bot               # Bot API
│   │       ├── actions       # Take Intent, Slots, Query KG h
│   │       ├── bot           # Entrypoint of the Bot API
│   │       ├── classifier    # Symentic Parse, Intent Match,
│   │       └── test          # Example Data as equivalent/moc
│   └── siwi_frontend         # Browser End
│       ├── README.md
│       ├── package.json
│       └── src
│           ├── App.vue       # Listen to user and pass Qs to
│           └── main.js
└── wsgi.py
```

wey-gu/**nebula-siwi**

Arch

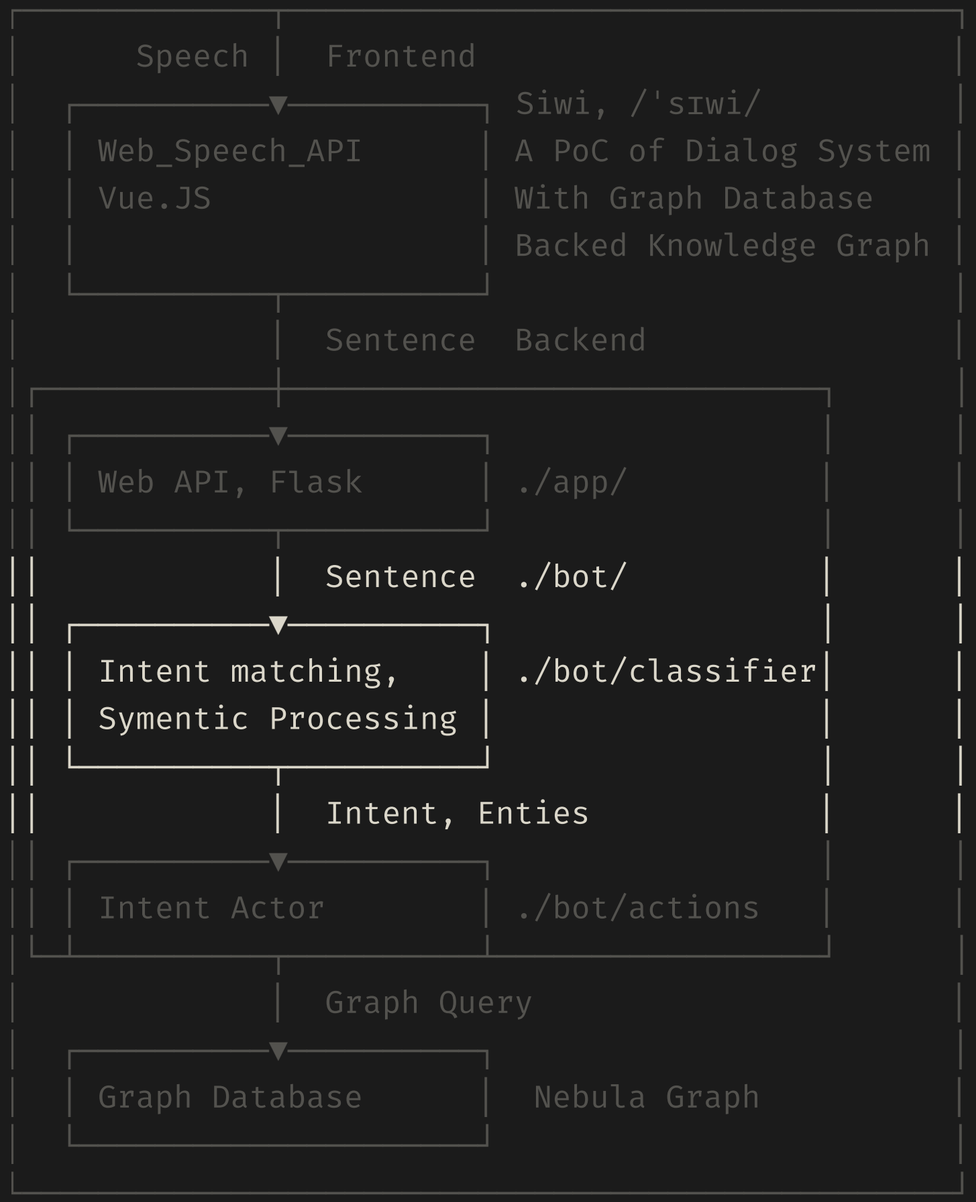


Code

```
.
├── README.md
├── src
│   ├── siwi                  # Siwi-API Backend
│   │   ├── app               # Web Server, take HTTP req > ca
│   │   └── bot               # Bot API
│   │       ├── actions       # Take Intent, Slots, Query KG h
│   │       ├── bot           # Entrypoint of the Bot API
│   │       ├── classifier    # Symentic Parse, Intent Match,
│   │       └── test          # Example Data as equivalent/moc
│   └── siwi_frontend         # Browser End
│       ├── README.md
│       ├── package.json
│       └── src
│           ├── App.vue       # Listen to user and pass Qs to
│           └── main.js
└── wsgi.py
```

wey-gu/**nebula-siwi**

Arch

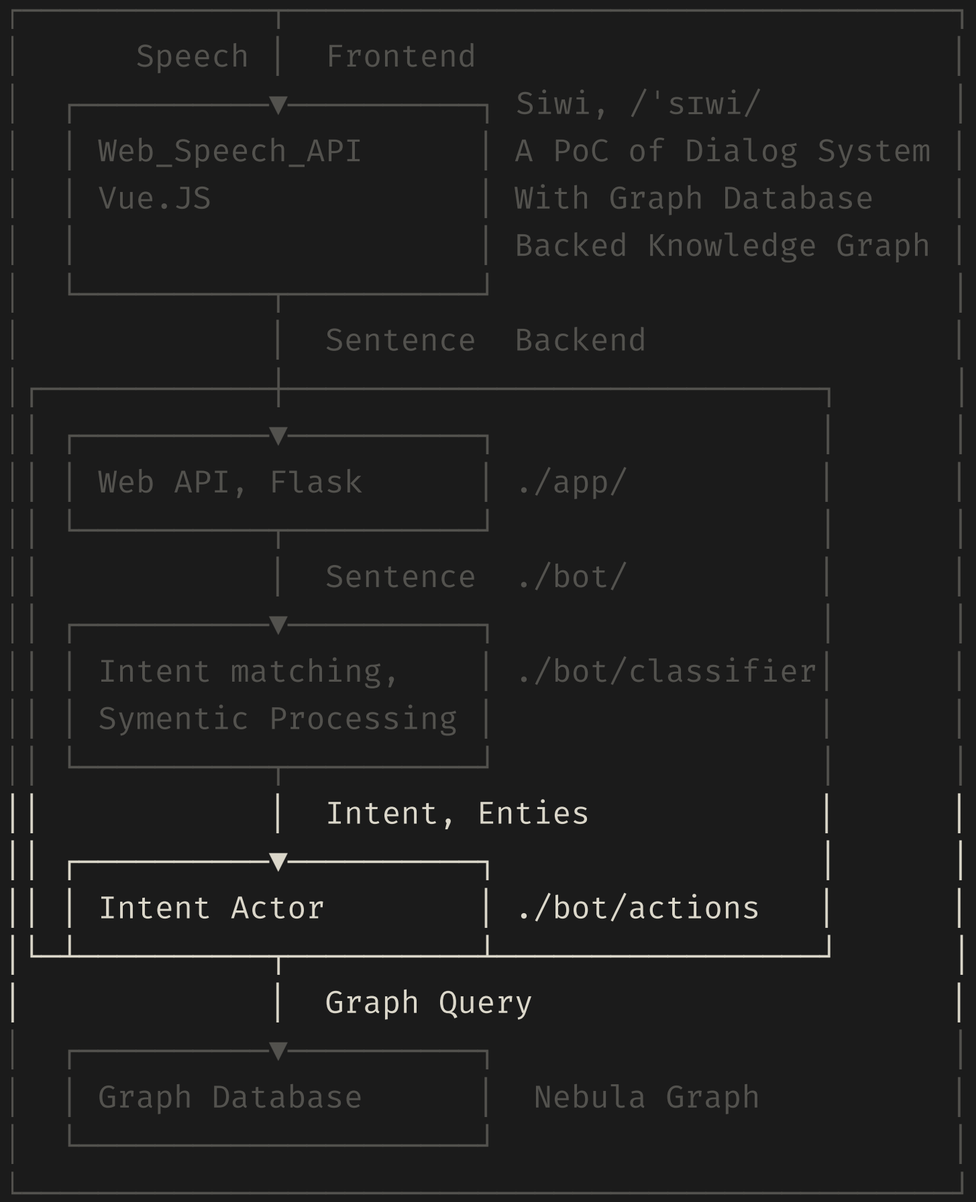


Code

```
.
├── README.md
├── src
│   ├── siwi                  # Siwi-API Backend
│   │   ├── app               # Web Server, take HTTP req > ca
│   │   └── bot               # Bot API
│   │       ├── actions       # Take Intent, Slots, Query KG h
│   │       ├── bot           # Entrypoint of the Bot API
│   │       ├── classifier    # Symentic Parse, Intent Match,
│   │       └── test          # Example Data as equivalent/moc
│   └── siwi_frontend         # Browser End
│       ├── README.md
│       ├── package.json
│       └── src
│           ├── App.vue       # Listen to user and pass Qs to
│           └── main.js
└── wsgi.py
```

wey-gu/**nebula-siwi**

Arch

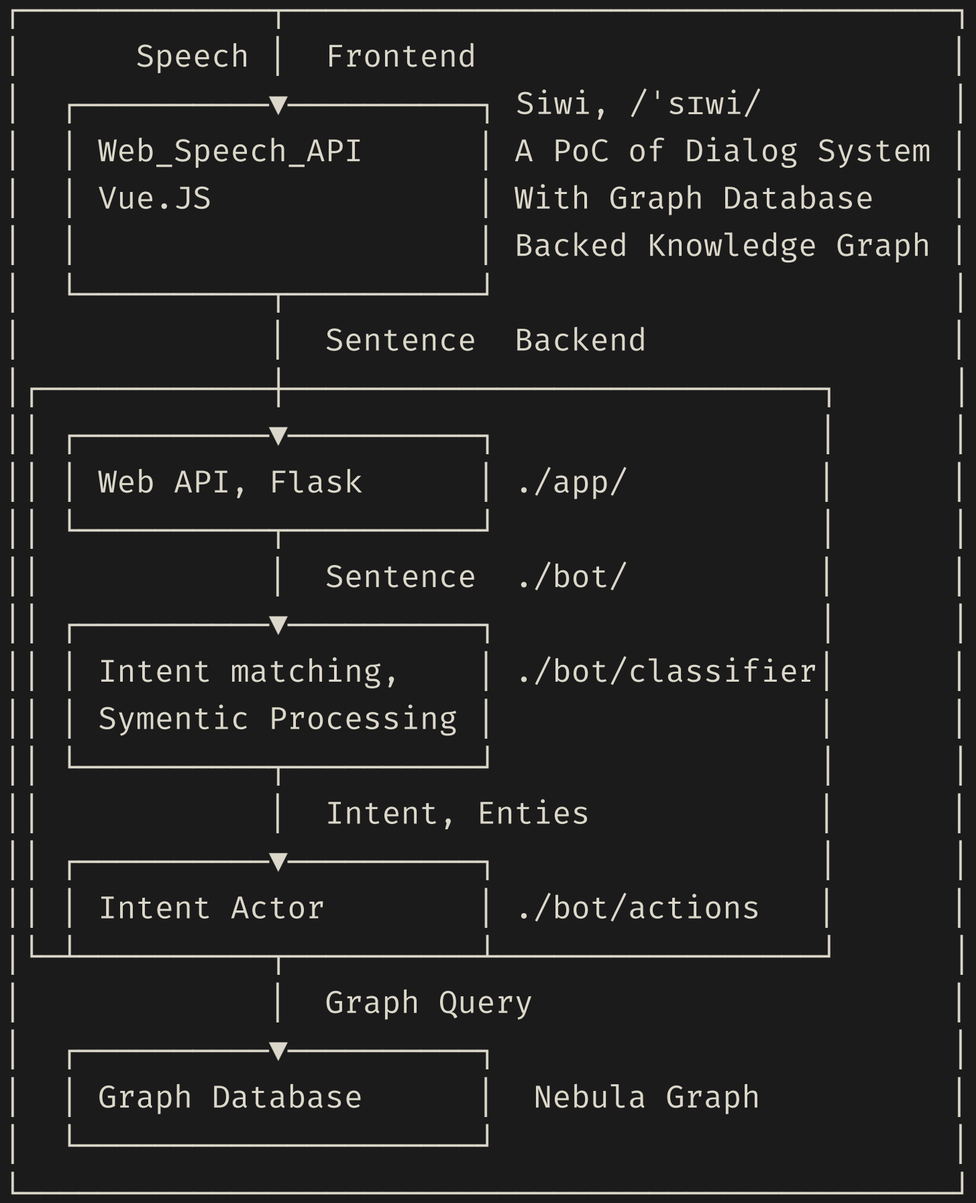


Code

```
.
├── README.md
├── src
│   ├── siwi                  # Siwi-API Backend
│   │   ├── app               # Web Server, take HTTP req > ca
│   │   └── bot               # Bot API
│   │       ├── actions       # Take Intent, Slots, Query KG h
│   │       ├── bot           # Entrypoint of the Bot API
│   │       ├── classifier    # Symentic Parse, Intent Match,
│   │       └── test          # Example Data as equivalent/moc
│   └── siwi_frontend         # Browser End
│       ├── README.md
│       ├── package.json
│       └── src
│           ├── App.vue       # Listen to user and pass Qs to
│           └── main.js
└── wsgi.py
```

wey-gu/**nebula-siwi**

Arch



Code

```
.
├── README.md
├── src
│   ├── siwi                  # Siwi-API Backend
│   │   ├── app               # Web Server, take HTTP req > ca
│   │   └── bot               # Bot API
│   │       ├── actions       # Take Intent, Slots, Query KG h
│   │       ├── bot           # Entrypoint of the Bot API
│   │       ├── classifier    # Symentic Parse, Intent Match,
│   │       └── test          # Example Data as equivalent/moc
│   └── siwi_frontend         # Browser End
│       ├── README.md
│       ├── package.json
│       └── src
│           ├── App.vue       # Listen to user and pass Qs to
│           └── main.js
└── wsgi.py
```

wey-gu/**nebula-siwi**

一个接地气的小 Demo

Siwi 智能语音助手

Corp-Rel-Search

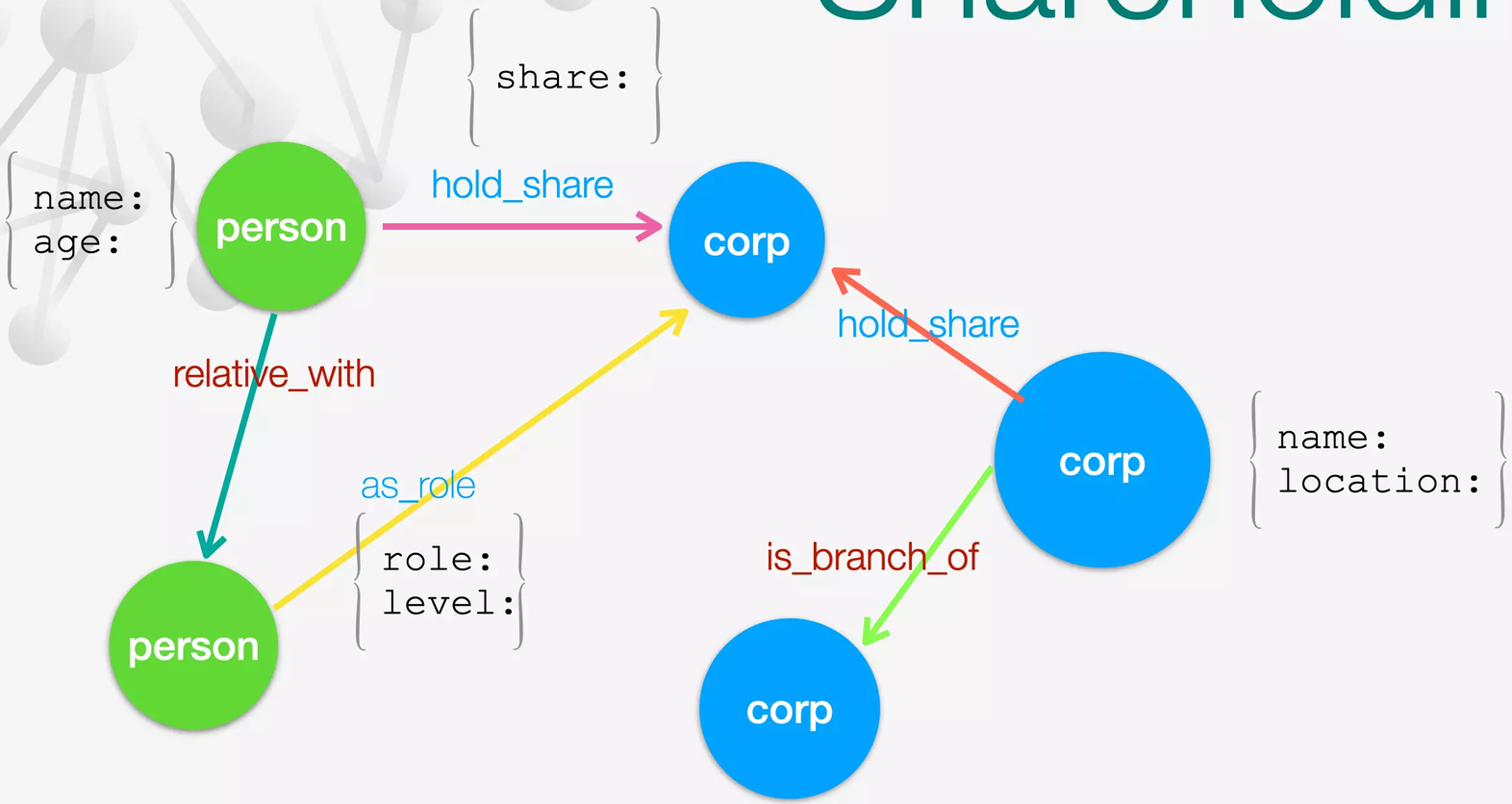
一个企业股权关系分析系统

墨天轮

https://nebula-graph.io

Shareholding

NebulaGraph

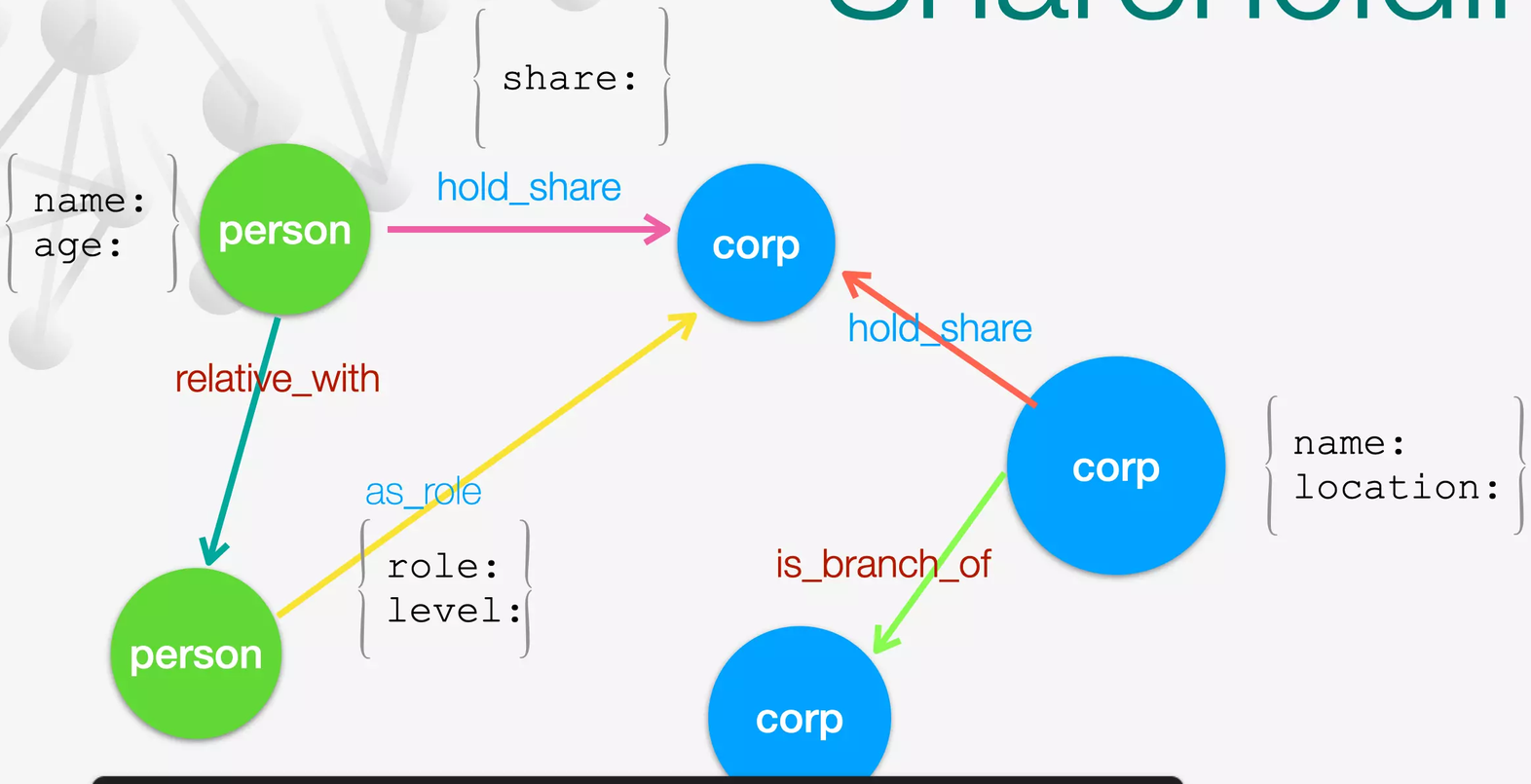


wey-gu/**nebula-shareholding-example**

https://nebula-graph.io

Shareholding

NebulaGraph



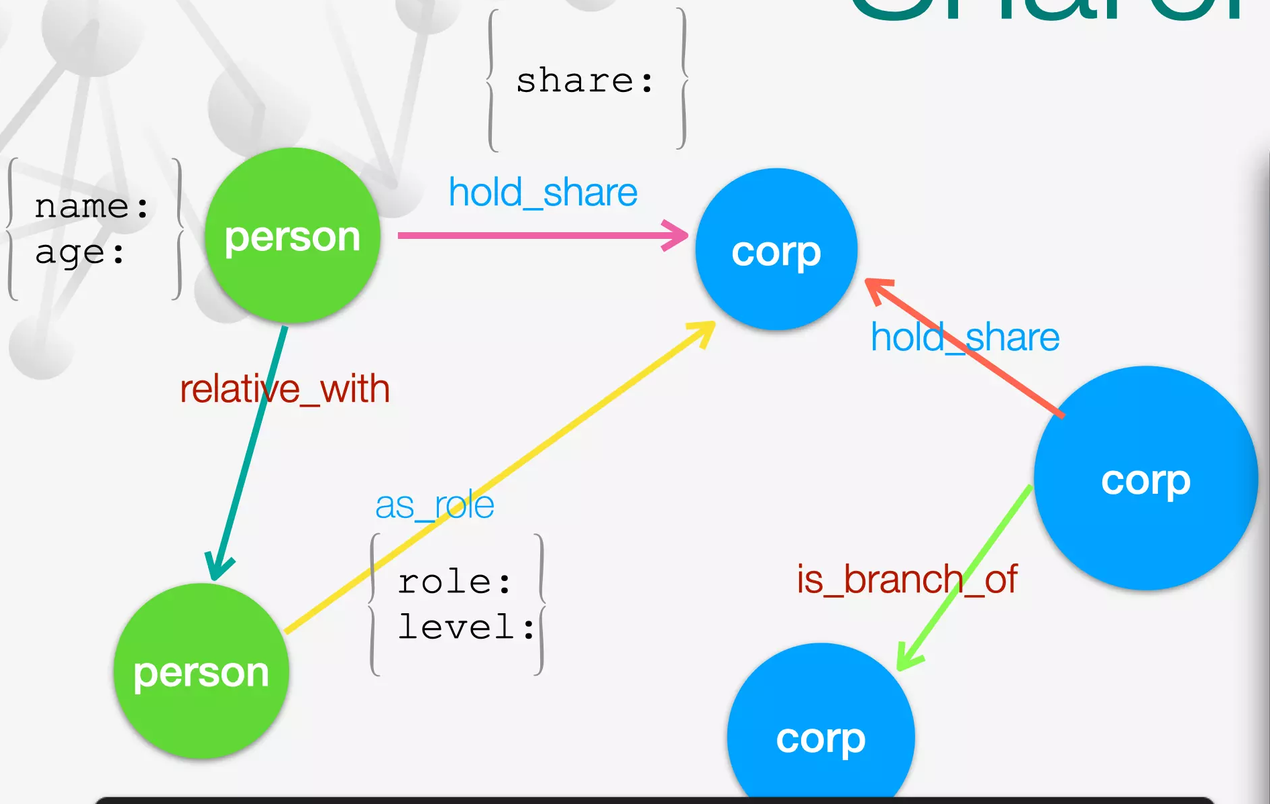

```
GO 1 TO 3 STEPS FROM "c_132" over * BIDIRECT;
```

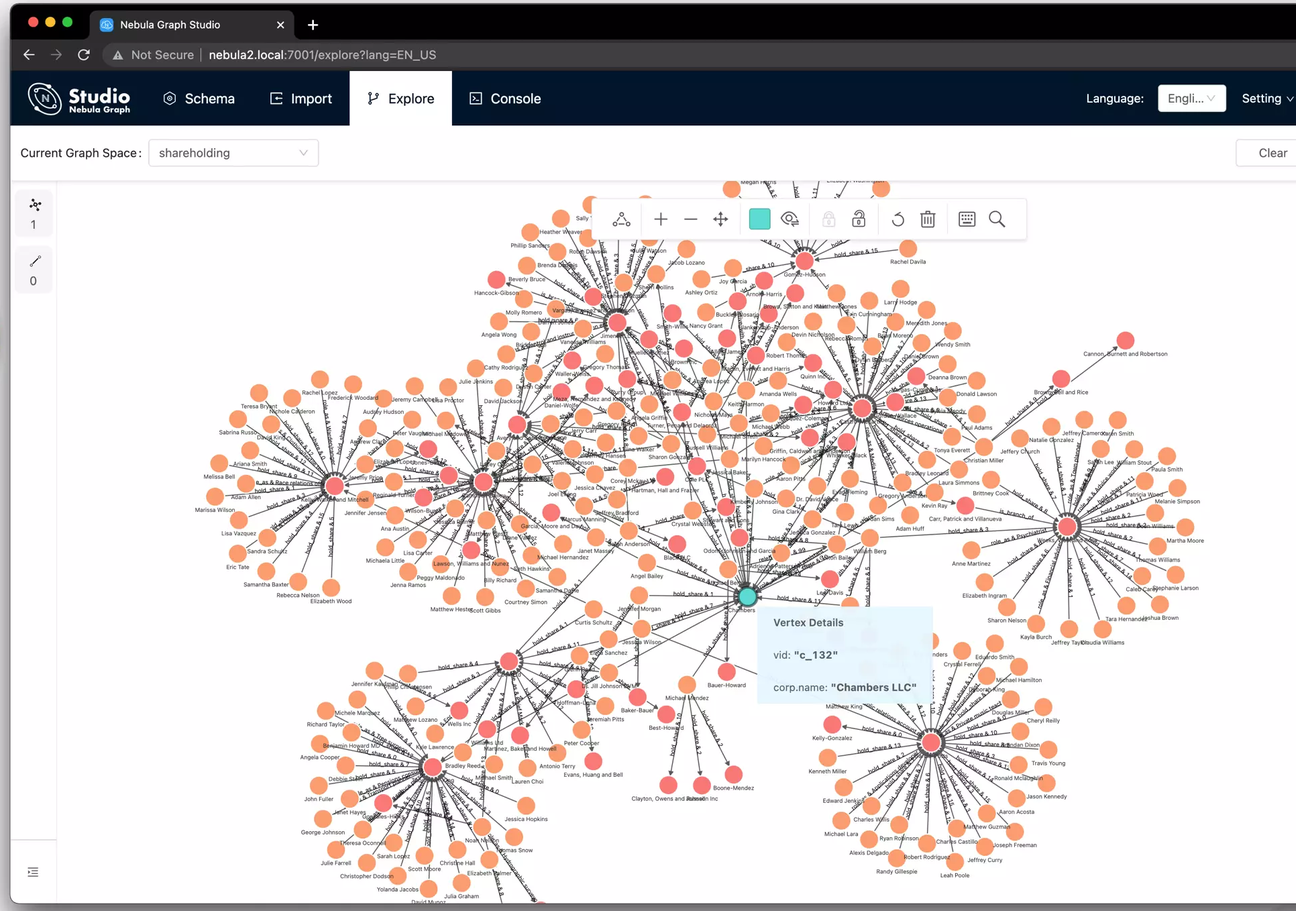
wey-gu/**nebula-shareholding-example**

https://nebula-graph.io

Shareholding

NebulaGraph

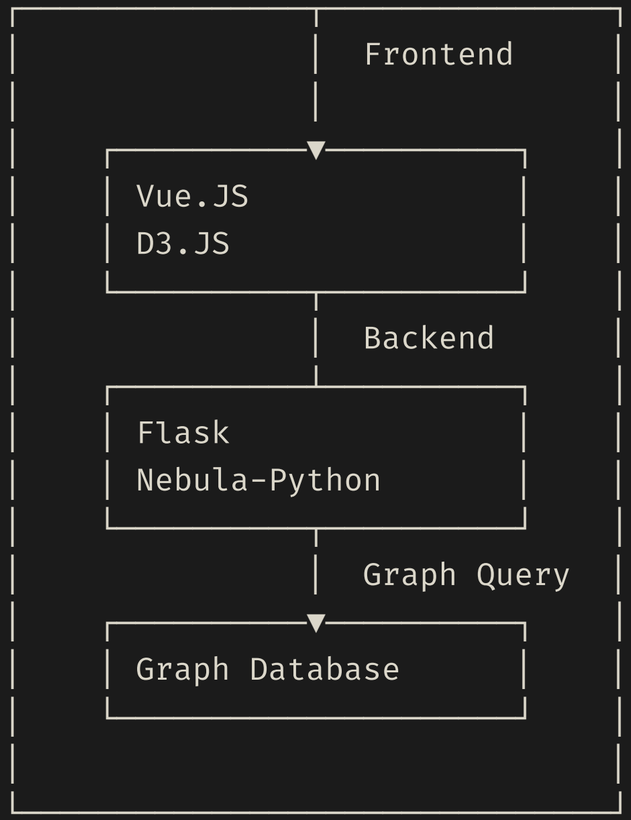





```
GO 1 TO 3 STEPS FROM "c_132" over * BIDIRECT;
```

wey-gu/**nebula-shareholding-example**

架构 Corp-Rel-Search



wey-gu/**nebula-corp-rel-search**

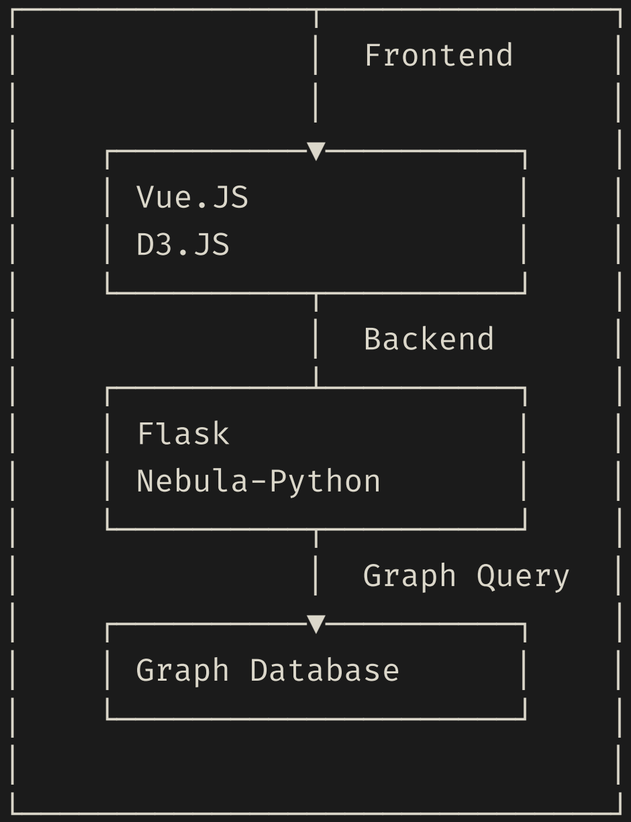
The Code

```
├── README.md          # You could find Design Logs here
├── corp-rel-backend
│   └── app.py         # Flask App to handle Requst and call
├── corp-rel-frontend
│   └── src
│       ├── App.vue
│       └── main.js    # Vue App to call Flask App and Rende
└── requirements.txt
```

```python
@app.route("/api", methods=["POST"])
def api():
    entity = request.get_json().get("entity", "")
    if entity:
        resp = query_shareholding(entity)
        data = make_graph_response(resp)
    # ...
    return jsonify(data)

def query_shareholding(entity):
    query_string = (
        f"MATCH p=(v)-[e:hold_share|:is_branch_of|:reletive
        f"WHERE id(v) IN ['{ entity }'] RETURN p LIMIT 100"
    )
    session = connection_pool.get_session('root', 'nebula')
    resp = session.execute(query_string)
    return resp
```

架构 Corp-Rel-Search



wey-gu/**nebula-corp-rel-search**

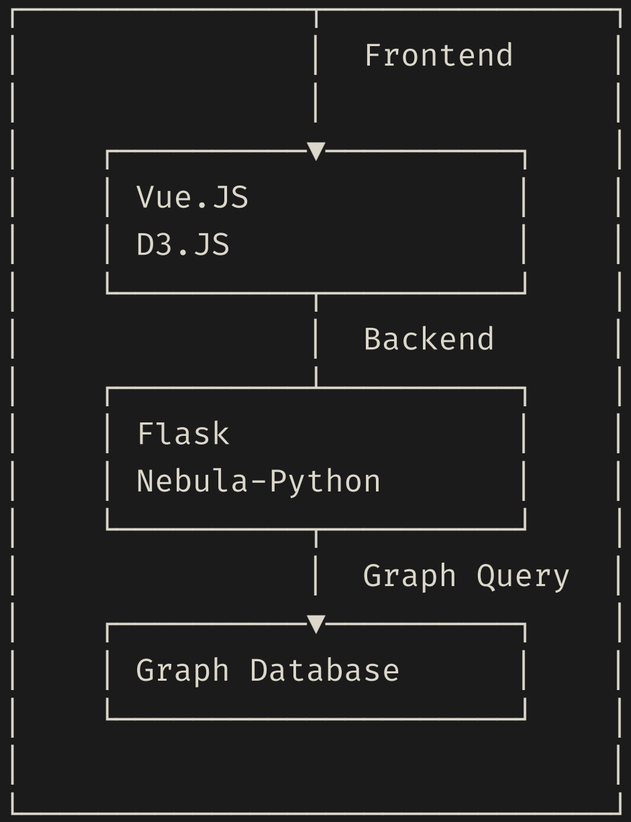
The Code

```
├── README.md          # You could find Design Logs here
├── corp-rel-backend
│   └── app.py         # Flask App to handle Requst and call
├── corp-rel-frontend
│   └── src
│       ├── App.vue
│       └── main.js    # Vue App to call Flask App and Rende
└── requirements.txt
```

```python
@app.route("/api", methods=["POST"])
def api():
    entity = request.get_json().get("entity", "")
    if entity:
        resp = query_shareholding(entity)
        data = make_graph_response(resp)
    # ...
    return jsonify(data)

def query_shareholding(entity):
    query_string = (
        f"MATCH p=(v)-[e:hold_share|:is_branch_of|:reletive
        f"WHERE id(v) IN ['{ entity }'] RETURN p LIMIT 100"
    )
    session = connection_pool.get_session('root', 'nebula')
    resp = session.execute(query_string)
    return resp
```

架构 Corp-Rel-Search



wey-gu/**nebula-corp-rel-search**

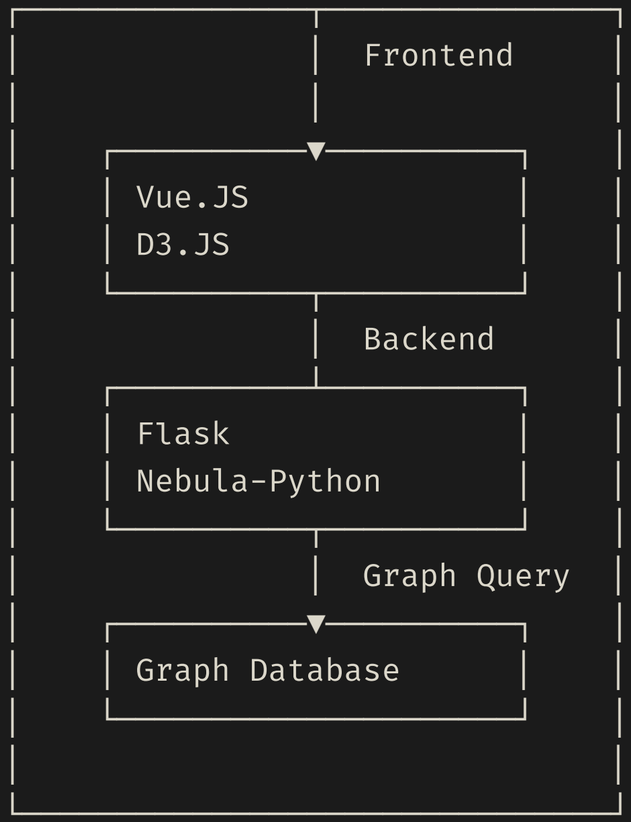
The Code

```
├── README.md          # You could find Design Logs here
├── corp-rel-backend
│   └── app.py         # Flask App to handle Requst and call
├── corp-rel-frontend
│   └── src
│       ├── App.vue
│       └── main.js    # Vue App to call Flask App and Rende
└── requirements.txt
```

```python
@app.route("/api", methods=["POST"])
def api():
    entity = request.get_json().get("entity", "")
    if entity:
        resp = query_shareholding(entity)
        data = make_graph_response(resp)
    # ...
    return jsonify(data)

def query_shareholding(entity):
    query_string = (
        f"MATCH p=(v)-[e:hold_share|:is_branch_of|:reletive
        f"WHERE id(v) IN ['{ entity }'] RETURN p LIMIT 100"
    )
    session = connection_pool.get_session('root', 'nebula')
    resp = session.execute(query_string)
    return resp
```

架构 Corp-Rel-Search



wey-gu/**nebula-corp-rel-search**

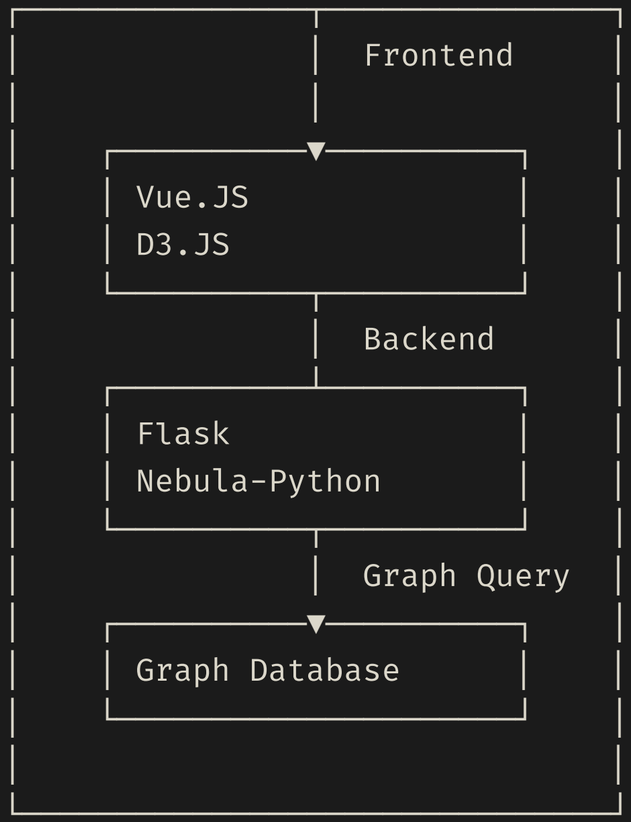
The Code

```
├── README.md          # You could find Design Logs here
├── corp-rel-backend
│   └── app.py         # Flask App to handle Requst and call
├── corp-rel-frontend
│   └── src
│       ├── App.vue
│       └── main.js    # Vue App to call Flask App and Rende
└── requirements.txt
```

```python
@app.route("/api", methods=["POST"])
def api():
    entity = request.get_json().get("entity", "")
    if entity:
        resp = query_shareholding(entity)
        data = make_graph_response(resp)
    # ...
    return jsonify(data)

def query_shareholding(entity):
    query_string = (
        f"MATCH p=(v)-[e:hold_share|:is_branch_of|:reletive
        f"WHERE id(v) IN ['{ entity }'] RETURN p LIMIT 100"
    )
    session = connection_pool.get_session('root', 'nebula')
    resp = session.execute(query_string)
    return resp
```

架构 Corp-Rel-Search



wey-gu/**nebula-corp-rel-search**

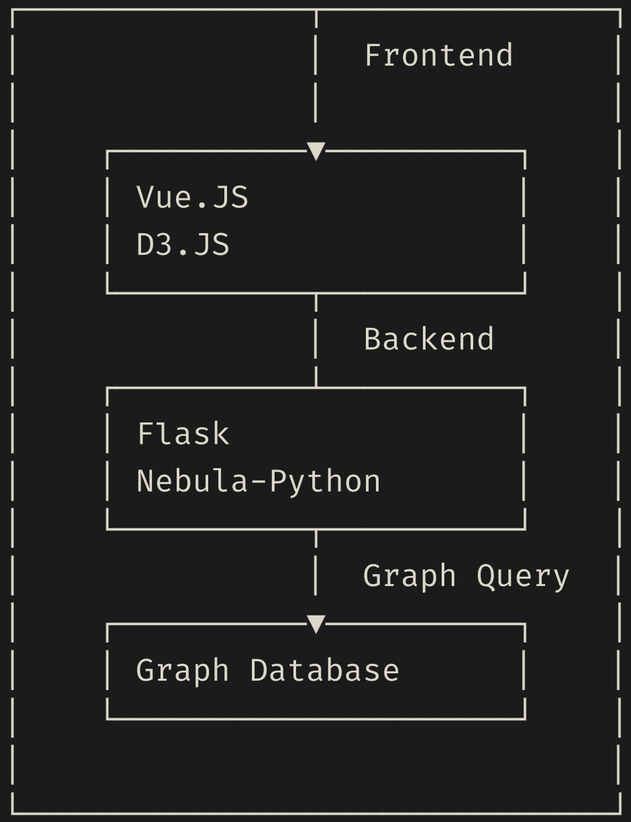
The Code

```
├── README.md          # You could find Design Logs here
├── corp-rel-backend
│   └── app.py         # Flask App to handle Requst and call
├── corp-rel-frontend
│   └── src
│       ├── App.vue
│       └── main.js    # Vue App to call Flask App and Rende
└── requirements.txt
```

```python
@app.route("/api", methods=["POST"])
def api():
    entity = request.get_json().get("entity", "")
    if entity:
        resp = query_shareholding(entity)
        data = make_graph_response(resp)
    # ...
    return jsonify(data)

def query_shareholding(entity):
    query_string = (
        f"MATCH p=(v)-[e:hold_share|:is_branch_of|:reletive_
        f"WHERE id(v) IN ['{ entity }'] RETURN p LIMIT 100"
    )
    session = connection_pool.get_session('root', 'nebula')
    resp = session.execute(query_string)
    return resp
```

架构 Corp-Rel-Search



wey-gu/**nebula-corp-rel-search**

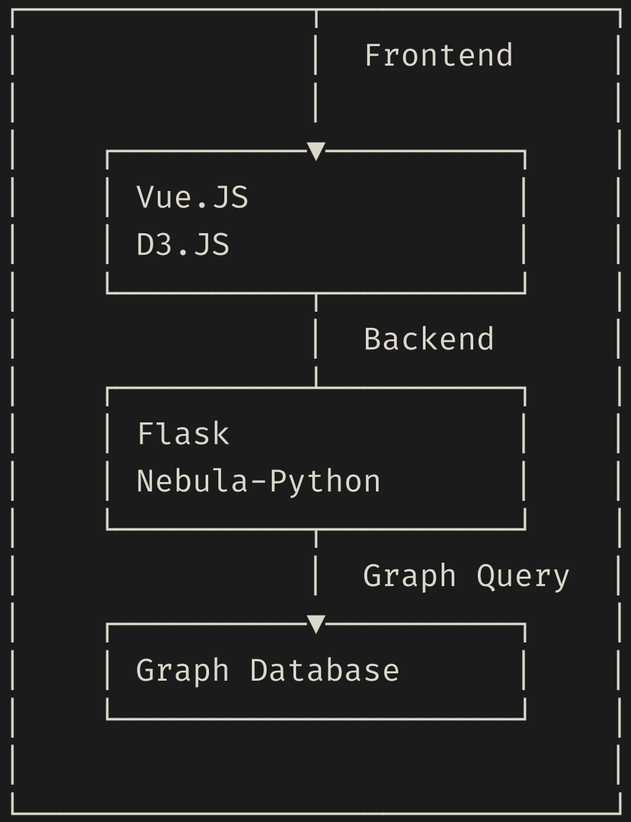
The Code

```
├── README.md          # You could find Design Logs here
├── corp-rel-backend
│   └── app.py         # Flask App to handle Requst and call
├── corp-rel-frontend
│   └── src
│       ├── App.vue
│       └── main.js    # Vue App to call Flask App and Rende
└── requirements.txt
```

```python
@app.route("/api", methods=["POST"])
def api():
    entity = request.get_json().get("entity", "")
    if entity:
        resp = query_shareholding(entity)
        data = make_graph_response(resp)
    # ...
    return jsonify(data)

def query_shareholding(entity):
    query_string = (
        f"MATCH p=(v)-[e:hold_share|:is_branch_of|:reletive
        f"WHERE id(v) IN ['{ entity }'] RETURN p LIMIT 100"
    )
    session = connection_pool.get_session('root', 'nebula')
    resp = session.execute(query_string)
    return resp
```

架构 Corp-Rel-Search



wey-gu/**nebula-corp-rel-search**

The Code

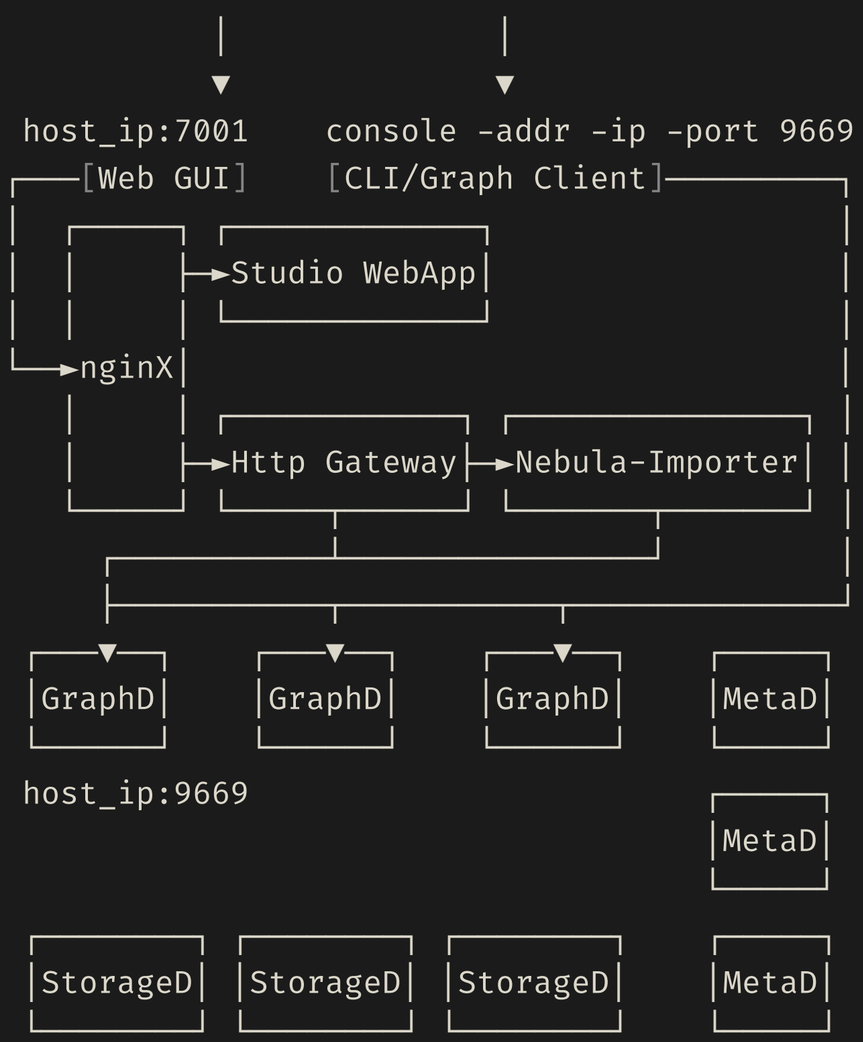
```
├── README.md          # You could find Design Logs here
├── corp-rel-backend
│   └── app.py         # Flask App to handle Requst and call
├── corp-rel-frontend
│   └── src
│       ├── App.vue
│       └── main.js    # Vue App to call Flask App and Rende
└── requirements.txt
```

```python
@app.route("/api", methods=["POST"])
def api():
    entity = request.get_json().get("entity", "")
    if entity:
        resp = query_shareholding(entity)
        data = make_graph_response(resp)
    # ...
    return jsonify(data)

def query_shareholding(entity):
    query_string = (
        f"MATCH p=(v)-[e:hold_share|:is_branch_of|:reletive
        f"WHERE id(v) IN ['{ entity }'] RETURN p LIMIT 100"
    )
    session = connection_pool.get_session('root', 'nebula')
    resp = session.execute(query_string)
    return resp
```

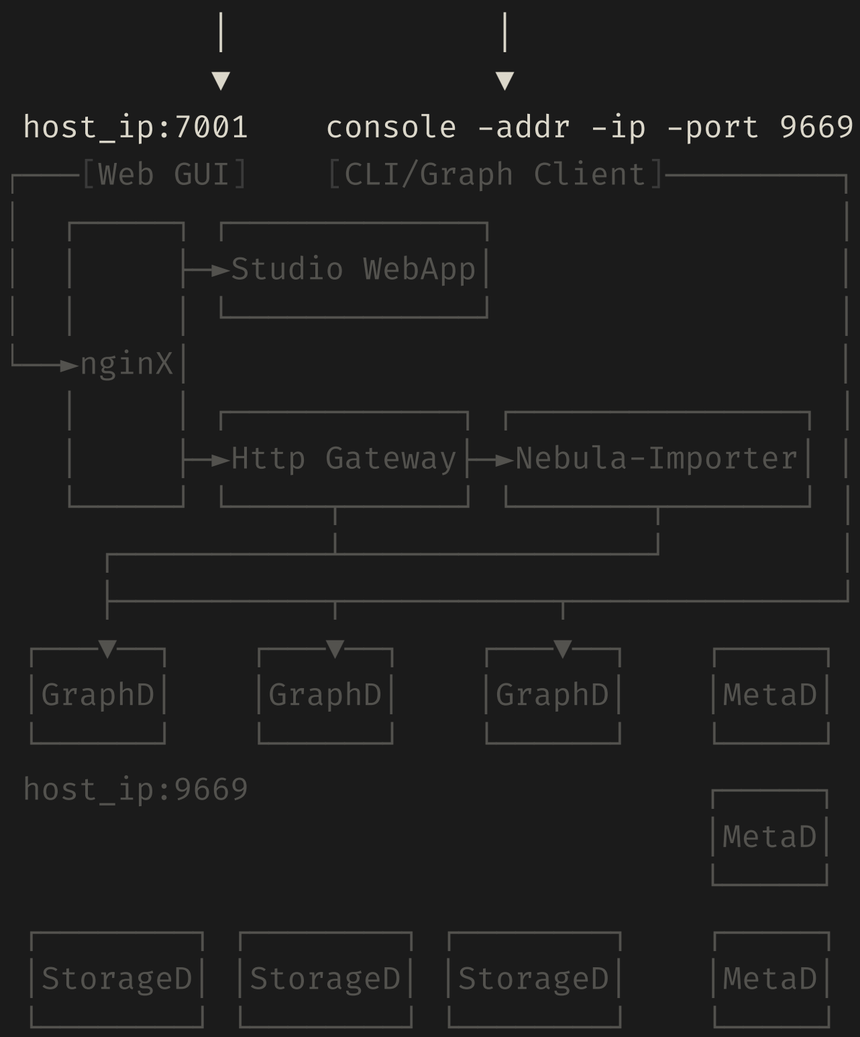
Nebula Graph in Docker and Dataset Import

```
curl -fsSL nebula-up.siwei.io/install.sh | bash
```



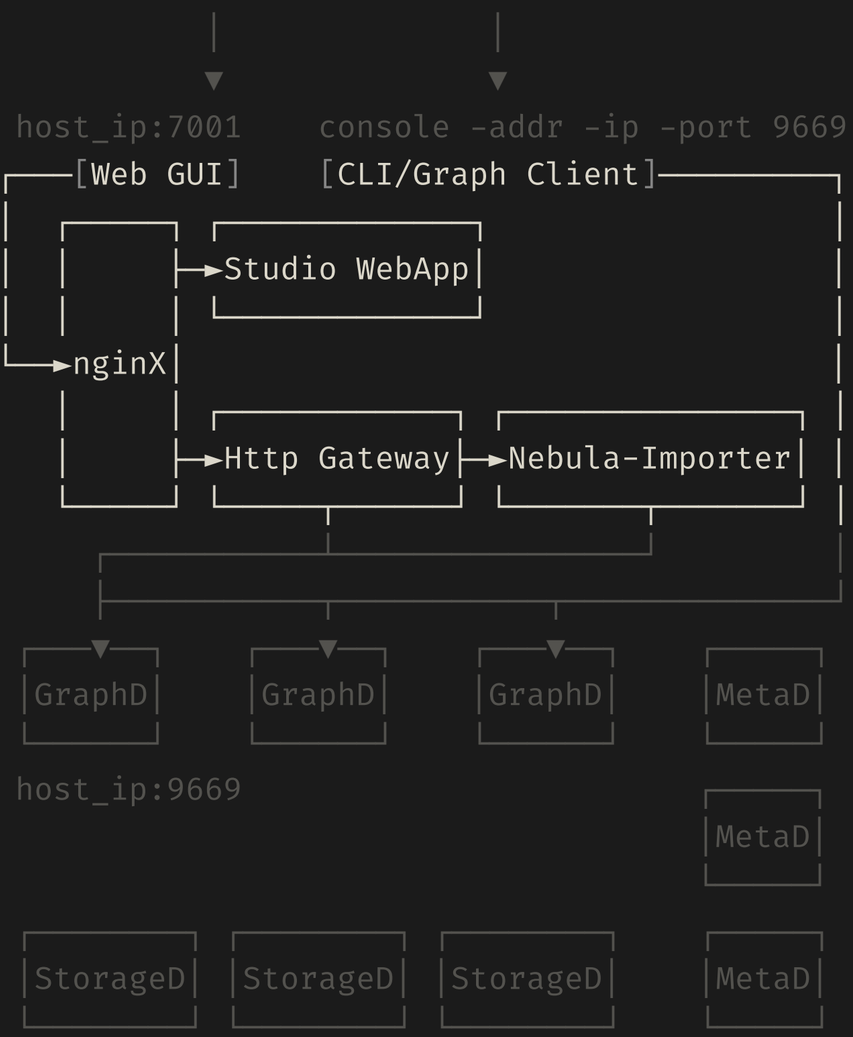
Nebula Graph in Docker and Dataset Import

```
curl -fsSL nebula-up.siwei.io/install.sh | bash
```



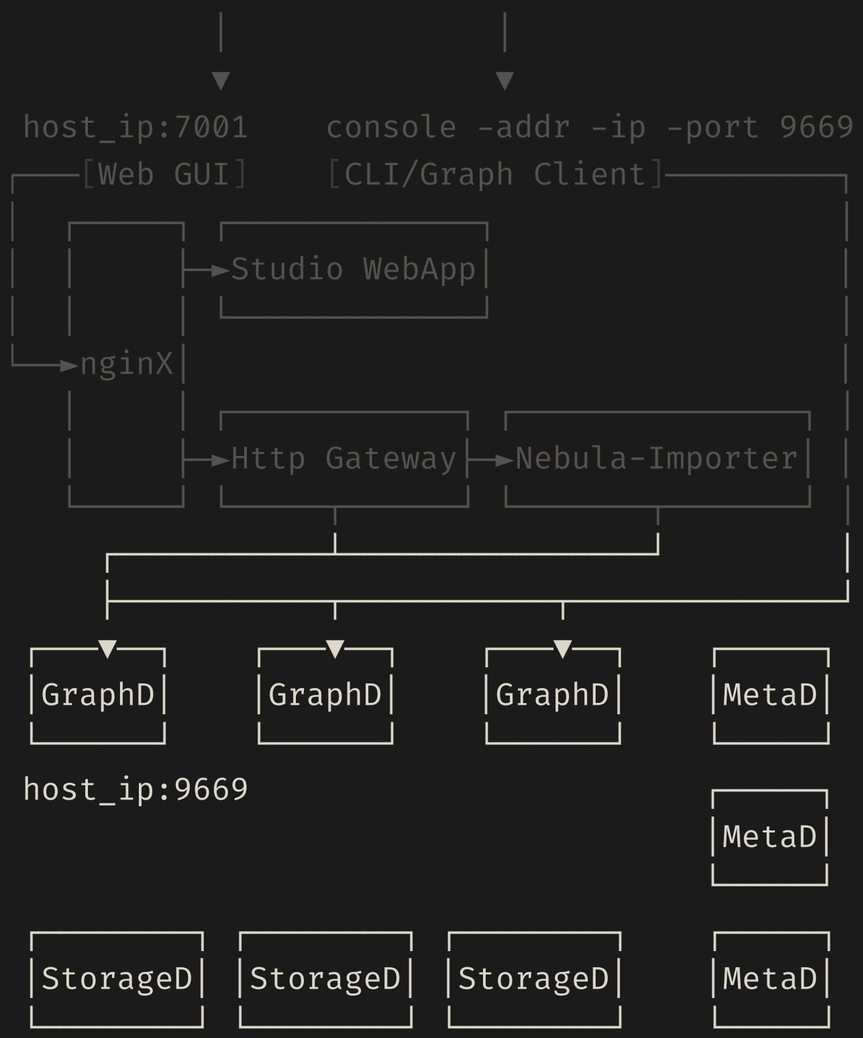
Nebula Graph in Docker and Dataset Import

```
curl -fsSL nebula-up.siwei.io/install.sh | bash
```



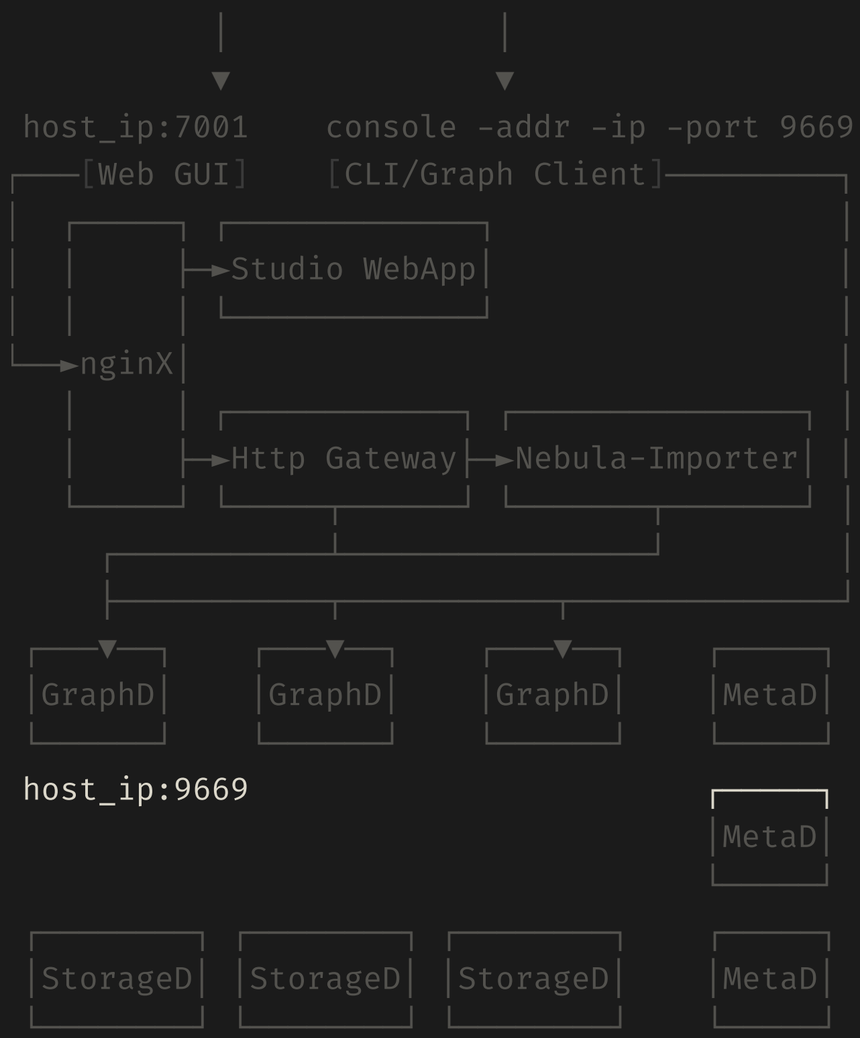
Nebula Graph in Docker and Dataset Import

```
curl -fsSL nebula-up.siwei.io/install.sh | bash
```



Nebula Graph in Docker and Dataset Import

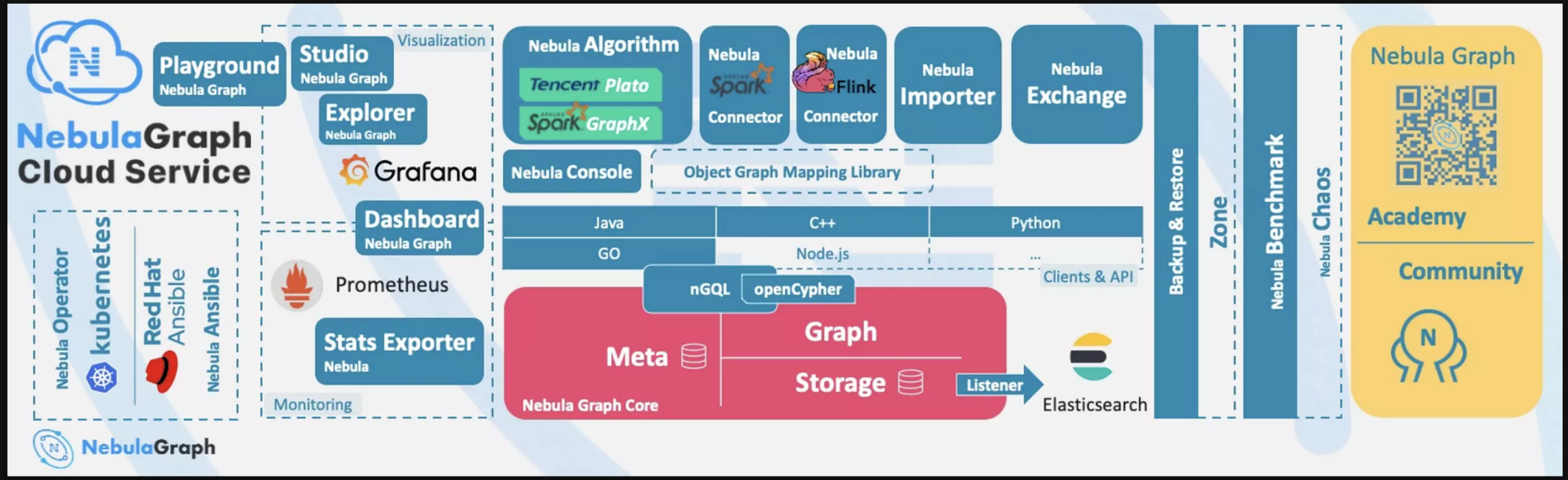
```
curl -fsSL nebula-up.siwei.io/install.sh | bash
```



Nebula Landscape

Nebula 社区生态非常丰富，并且还在日益拓展，欢迎同学们了解、参与贡献。

- Deployment, Monitoring
- Data Visualization
- Algorithm, Analytic
- Clients, Connectors, ETL



vesoft-inc/**nebula**

墨天轮

为什么选择 Nebula?

为什么选择 Nebula?

- ❤️ Open Source

ⓘ x-lab/2020 开源报告

为什么选择 Nebula?

- ❤️ Open Source
- 🏂 生态、性能、只有它能 hold 得住、拥抱标准、领先

ⓘ x-lab/**2020 开源报告**

为什么选择 Nebula?

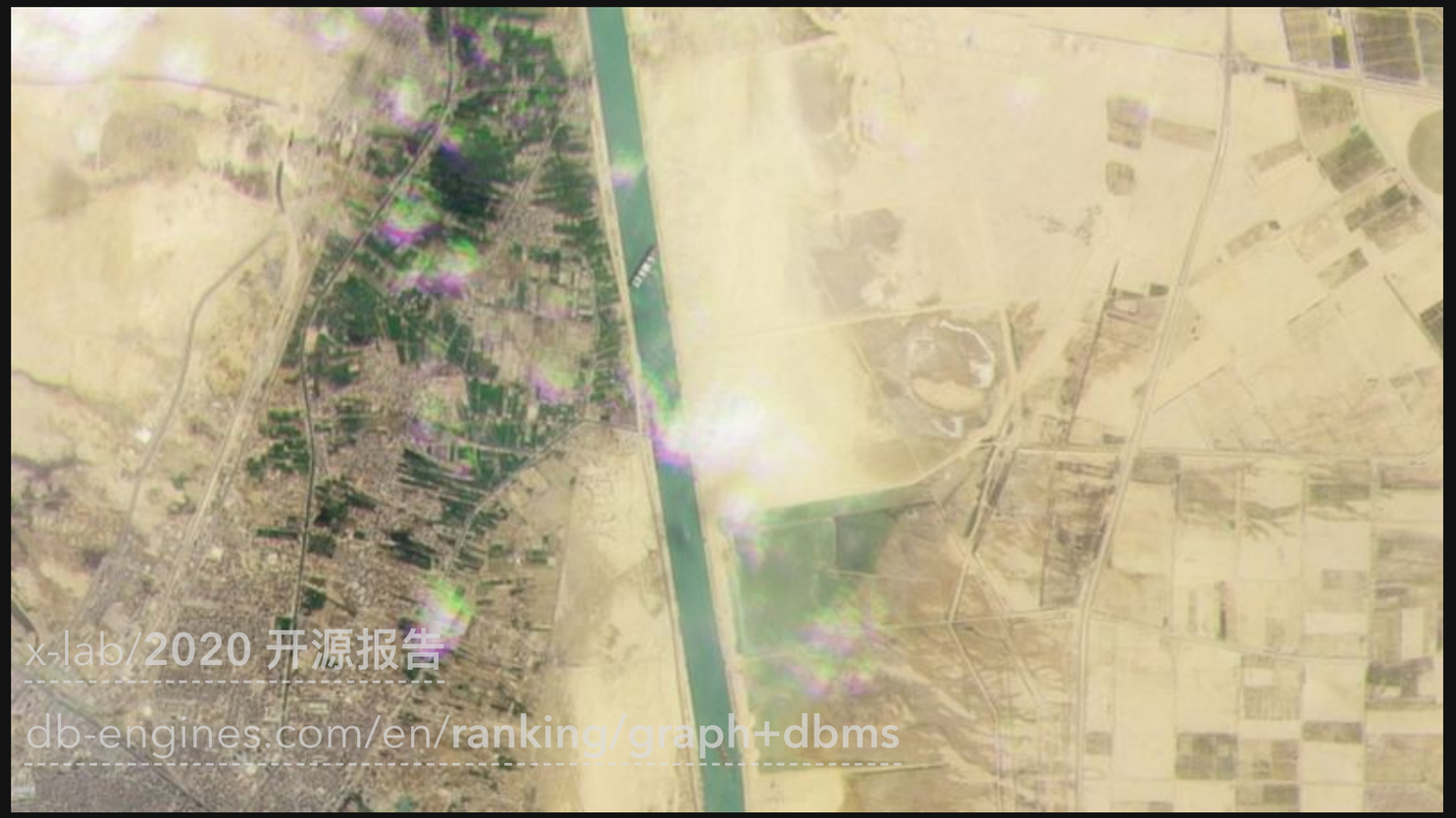
- ❤️ Open Source
- 🏄 生态、性能、只有它能 hold 得住、拥抱标准、领先
- 🐋 Scale matters, Nebula Nailed it! Nebula 擅长超大规模图数据处理

ⓘ x-lab/**2020 开源报告**

ⓘ db-engines.com/en/**ranking/graph+dbms**

为什么选择 Nebula?

- ❤️ Open Source
- 🏄 生态、性能、只有它能 hold 得住、拥抱标准、领先
- 🐋 Scale matters, Nebula Nailed it! Nebula 擅长超大规模图数据处理



ⓘ x-lab/2020 开源报告

ⓘ db-engines.com/en/ranking/graph+dbms

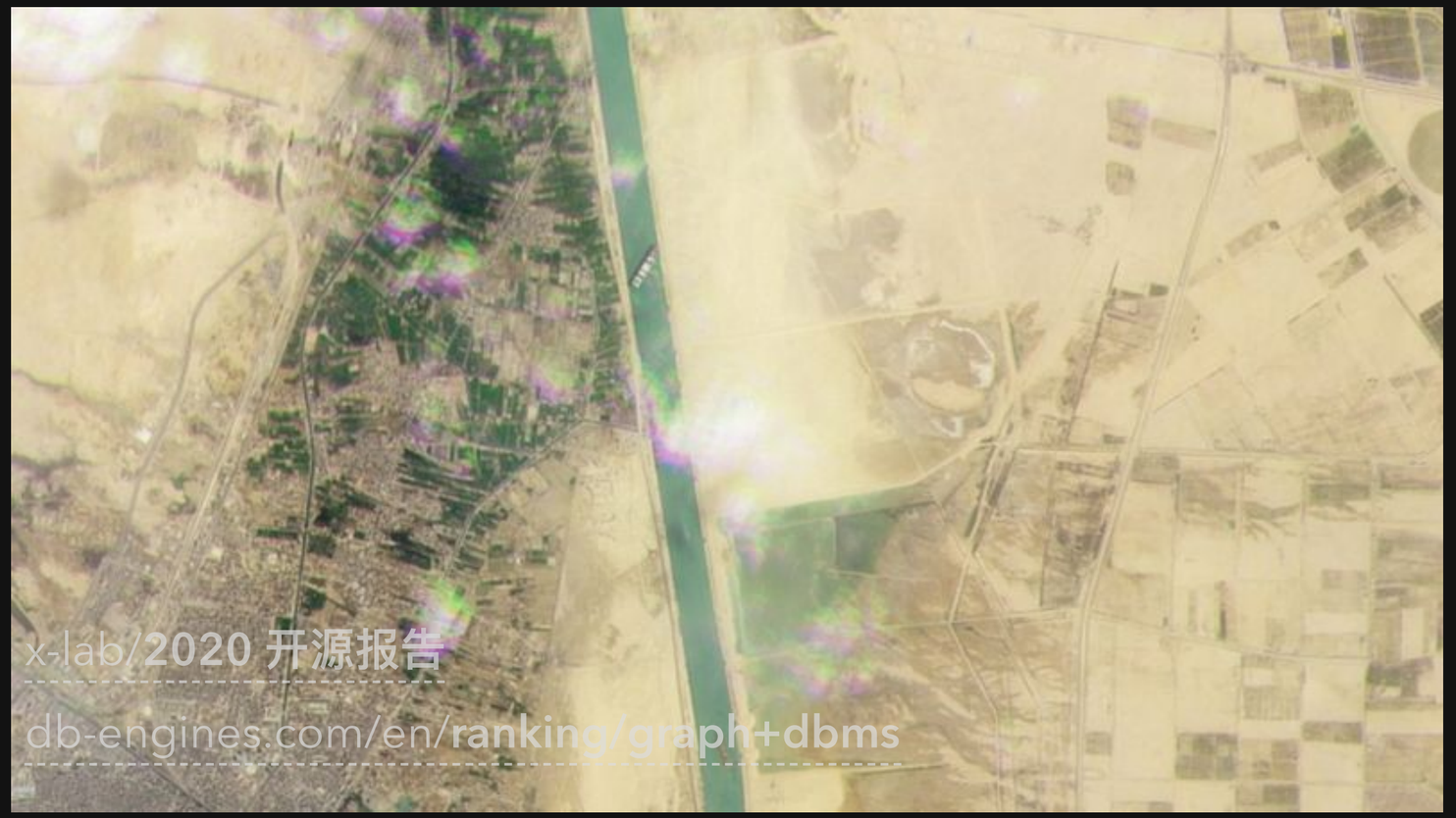
Wey Gu 古思为
@wey_gu

Scale makes differences
@NebulaGraph

> **Martin Beeby** @thebeebs
> Small and medium businesses may have similar problems to big tech companies. But should never assume they need to use the same solution. The answer is not always a shiny new library, a complicated frontend build process, or the latest orchestrator. It is all a matter of scale.

为什么选择 Nebula?

- ❤️ Open Source
- 🏄 生态、性能、只有它能 hold 得住、拥抱标准、领先
- 🐋 Scale matters, Nebula Nailed it! Nebula 擅长超大规模图数据处理



ⓘ x-lab/2020 开源报告

ⓘ db-engines.com/en/ranking/graph+dbms

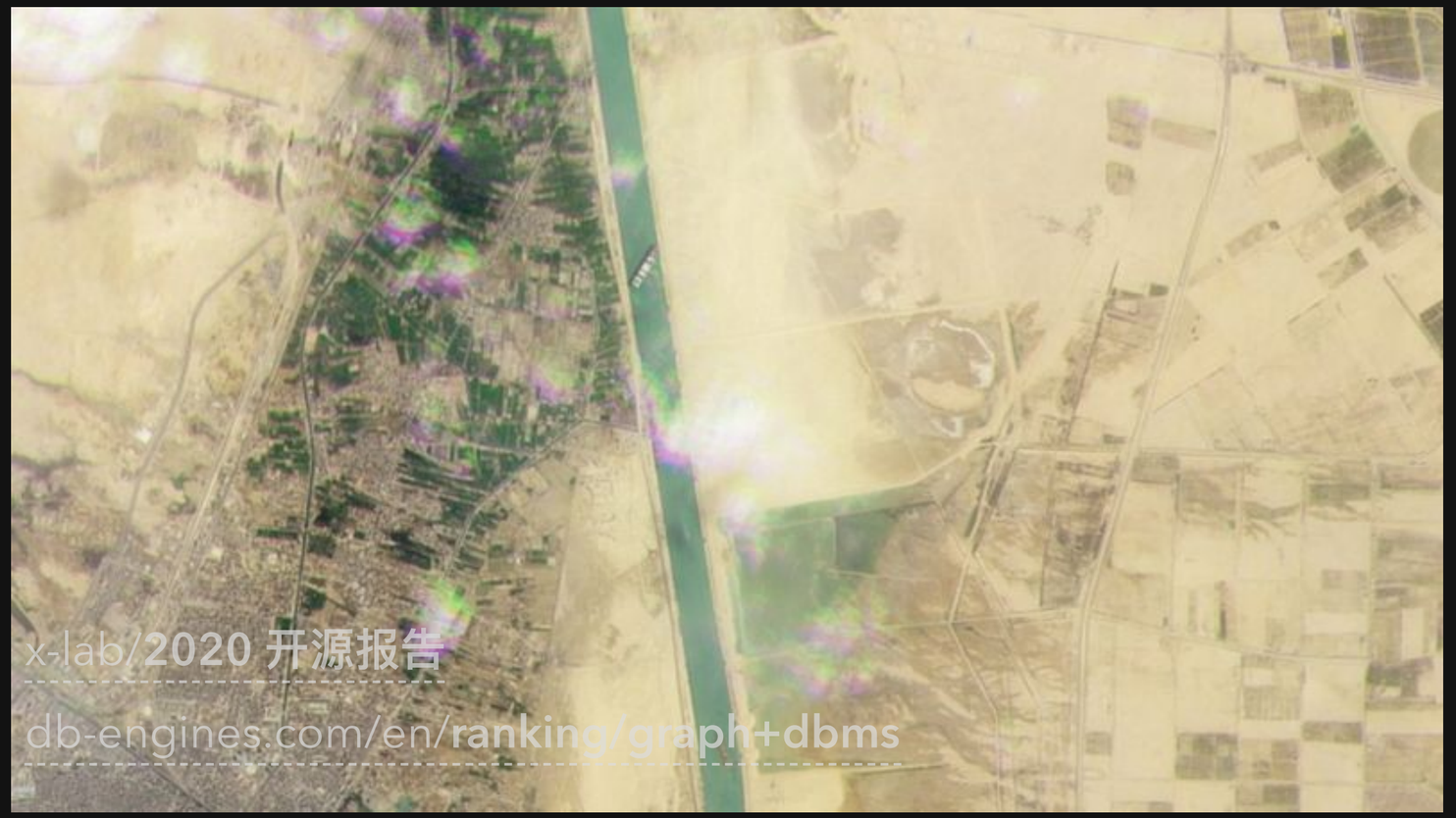
Wey Gu 古思为
@wey_gu

Scale makes differences
@NebulaGraph

> **Martin Beeby** @thebeebs
> Small and medium businesses may have similar problems to big tech companies. But should never assume they need to use the same solution. The answer is not always a shiny new library, a complicated frontend build process, or the latest orchestrator. It is all a matter of scale.

为什么选择 Nebula?

- ❤️ Open Source
- 🏄 生态、性能、只有它能 hold 得住、拥抱标准、领先
- 🐋 Scale matters, Nebula Nailed it! Nebula 擅长超大规模图数据处理



ⓘ x-lab/2020 开源报告

ⓘ db-engines.com/en/ranking/graph+dbms

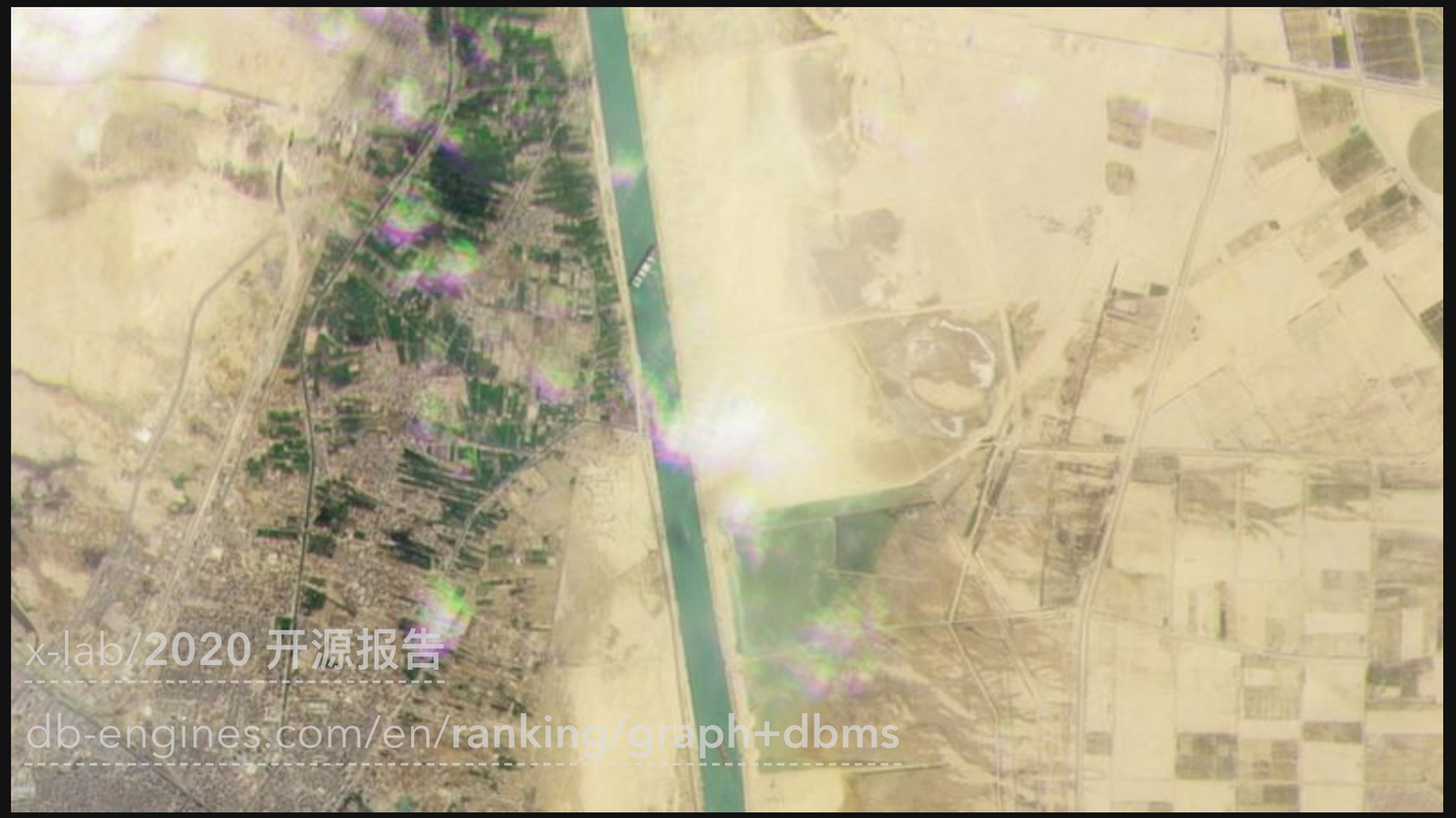
Wey Gu 古思为
@wey_gu

Scale makes differences
@NebulaGraph

> **Martin Beeby** @thebeebs
> Small and medium businesses may have similar problems to big tech companies. But should never assume they need to use the same solution. The answer is not always a shiny new library, a complicated frontend build process, or the latest orchestrator. It is all a matter of scale.

为什么选择 Nebula?

- ❤️ Open Source
- 🏄 生态、性能、只有它能 hold 得住、拥抱标准、领先
- 🐋 Scale matters, Nebula Nailed it! Nebula 擅长超大规模图数据处理



ⓘ x-lab/2020 开源报告

ⓘ db-engines.com/en/ranking/graph+dbms

Wey Gu 古思为
@wey_gu

Scale makes differences
@NebulaGraph

> **Martin Beeby** @thebeebs
> Small and medium businesses may have similar problems to big tech companies. But should never assume they need to use the same solution. The answer is not always a shiny new library, a complicated frontend build process, or the latest orchestrator. It is all a matter of scale.

回顾

siwei.io/talks/2021-modb

墨天轮 线上直播

Dec. 9th, 2021

回顾

- 🗺️ 基础： **什么是图？图数据库？**

siwei.io/talks/2021-modb

墨天轮 线上直播
Dec. 9th, 2021

回顾

- 🗺️ 基础：**什么是图？图数据库？**
- ✨ 疗效：**为什么需要图数据库？**

siwei.io/talks/2021-modb

墨天轮 线上直播
Dec. 9th, 2021

回顾

- 🗺️ 基础： **什么是图？图数据库？**
- ✨ 疗效： **为什么需要图数据库？**
- 👨‍💻 适应症： **到底图数据库可以做什么？**

siwei.io/talks/2021-modb

墨天轮 线上直播
Dec. 9th, 2021

回顾

- 🗺️ 基础：**什么是图？图数据库？**
- ✨ 疗效：**为什么需要图数据库？**
- 👨‍💻 适应症：**到底图数据库可以做什么？**
- 🌌 Why Nebula（星云）：**处理超大规模图数据、活跃的开源社区、丰富的生态、分布式架构、领先、快速进化、云原生...**

siwei.io/talks/2021-modb

墨天轮 线上直播
Dec. 9th, 2021