

智匯方舟

Strategy Plan 20XX

COMPANY NAME

我們的專案



01 背景

AI時代中重要的能力
我們想要解決的問題
困難點
解決方式

02 目標用戶

小故事

03 專案介紹

總覽
產品介紹
程式架構
架構優勢

04 demo&未來的展望

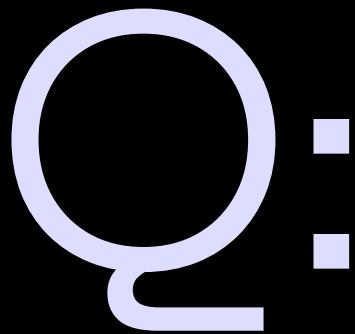
Demo
未來展望

05 結論&QA

小結
我們學到了什麼
QA

01

背景



在llm發展的時代, 重要的能力是什麼

A:

1. 提出問題

- 前知識很重要(決定問題的品質)

2. 存下答案

- 避免做白工
- 累積前知識

使用的工具

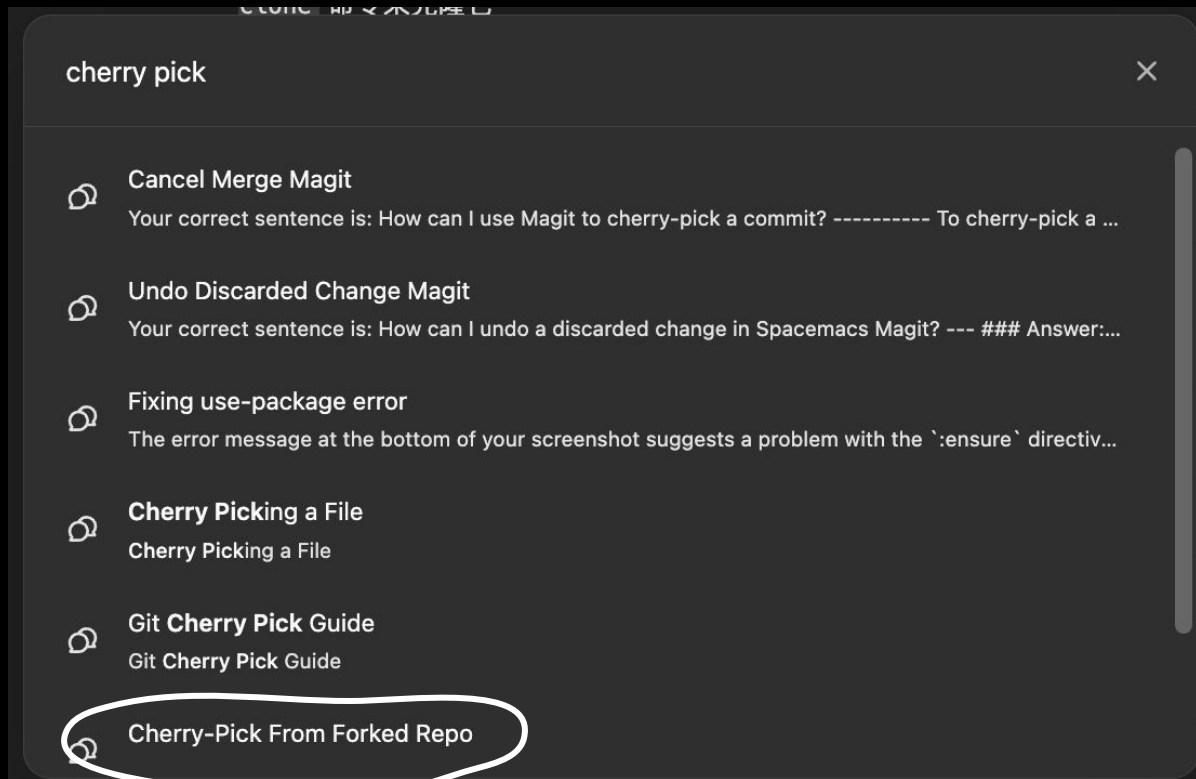
1. 查資料: perplexity
2. 問問題: Chatgpt
3. 學習: NotebookLm
4. 程式錯誤: Google stackoverflow

親身經驗

問題: how to cherry-pick from
other repo?

第一步: Chatgpt

我記得自己這
個禮拜有問過
一樣的問題?



看完了長長的對話紀錄：不對啊，我記得我看過更簡單的做法

那麼怎麼 cherry-pick 不同 repo (同一個庫裏 fork 出來的)

- 在 GitHub 中，一個 fork 的 repository 的 cherry-pick 提交到另一個，你需要先將那個 fork 加入到你本地的 repository 作為一個遠程參考，然後才能 cherry-pick 提交，以下是一個基本的步驟指南：

- 添加遠程參考：首先，確保你有該 fork repository 的一個本地副本。如果沒有，你可以使用 `git clone` 來克隆它。

```
bash
git clone -> 添加 repo URL
cd <克隆下庫的目錄>
```

然後，添加 fork repository 作為一個新的遠程參考 (通常稱為 remote)：

```
bash
git remote add <fork-remote> <fork repo URL>
```

例如，如果 fork 名稱是 "forked_repo"，你可以這樣做：

```
bash
git remote add forked_repo <fork repo URL>
```

- 獲取遠程變更：使用 `git fetch` 命令從新增的遠程參考中獲取變更。

```
bash
git fetch <fork-remote>
```

這會下載 fork repository 的所有分支到本地，但不會合併到你的本地分支。

- Cherry-pick 變更：現在，你可以使用 `git cherry-pick` 命令來選擇你想要合併的提交。首先，確保你處於你要選擇提交的分支上。如果無權，可以使用 `git checkout` 切換分支。然後使用以下命令進行 cherry-pick。

```
bash
git cherry-pick <提交的 SHA-1 散列碼>
```

這裡的 <提交的 SHA-1 散列碼> 是你想要從 fork 的 cherry-pick 的提交的 ID。

- 處理衝突：如果 cherry-pick 過程中出現衝突，你需要手動解決這些衝突。在解決衝突後，使用 `git add` 標記解決衝突的文件，然後使用 `git cherry-pick --continue` 完成 cherry-pick 過程。

- 推送變更：完成這些操作後，你可以選擇將更改推送到你選擇的 repository。

```
bash
git push origin <分支名稱>
```

注意，cherry-picking 是一種選擇性複製，它只包含你選擇的提交。這意味著如果你需要多個提交，它不會自動複製其他提交的依賴。如果你需要所有相關提交的完整歷史記錄，可能更適合使用其他方法，如合併或 rebase。

那麼怎麼 diff 一個 commit

- 要查看一個 commit 的差異 (diff)，你可以使用 `git diff` 命令。這個命令允許你查看特定提交與其他提交之間的差異，以下有幾種基本的用法：

- 查看最新一次提交的差異：如果你只想查看最近一次提交所做的更改，可以使用：

```
bash
git diff HEAD^ HEAD
```

這裏，`HEAD` 指的是當前分支的最後提交，而 `HEAD^` 是指最新提交的父提交。

- 查看任意提交之間的差異：要查看特定提交的更改，你需要知道那個提交的 SHA-1 散列碼，然後使用以下命令：

```
bash
git diff <提交的 SHA-1 散列碼> <提交的 SHA-1 散列碼>
```

例如，如果提交的散列碼是 `abc123`，那么命令就是：

```
bash
git diff abc123^ abc123
```

- 查看兩個不同提交之間的差異：你也可以比較任意兩個提交之間的差異，例如，比較提交 `abc123` 和 `def456`：

```
bash
git diff abc123 def456
```

- 查看特定文件的差異：如果你只對特定文件的更改感興趣，可以在 `git diff` 命令後面加文件名：

```
bash
git diff <提交的 SHA-1 散列碼> <提交的 SHA-1 散列碼> -- <文件路徑>
```

例如，查看提交 `abc123` 中 `example.txt` 文件的差異：

```
bash
git diff abc123^ abc123 -- example.txt
```

注意，Git 的 SHA-1 散列碼是一個 40 字字符串，但通常只需要提供前 7 個字 (通常寫成 6 個)，只要它們在你的 repository 中是唯一的。通過這種方式，你可以快速查看不同提交之間的差異，幫助你理解代碼的演化和調試。

第二步：詢問GPT

跟一開始的一樣

我要怎麼從其他repo cherry pick

- ▶ 你的正確句子是 =>
我要如何從其他的 repo 進行 cherry-pick ?

從其他 repo 進行 cherry-pick 的方法

如果你想從另一個 repository (repo) 中挑選特定的 commit，通常可以透過以下步驟來完成：

方法 1：使用 `git fetch + cherry-pick`

1. 切換到目標分支（確保你在正確的分支上進行 cherry-pick）：

```
bash
```

複製 編輯

給了很多做法，但不知道用那一個

有沒有更好的做法

- ▶ 你的正確句子是 =>
有沒有更好的做法？

更好的方法來從其他 repo 拿 commit

`git cherry-pick` 雖然有效，但如果你經常從另一個 repo 引入變更，可能會有更好的方式來處理，例如使用 `git merge` 或 `git rebase`，這樣可以避免手動挑選 commit，並且保持更完整的 commit 歷史。

以下是一些更好的做法，視你的需求選擇合適的方法：

方法 1：使用 `git format-patch + git am`（適合從別 repo 搬運特定 commit）

第四步: perplexity

跟GPT的回答差不多qq

第五步: Google

找看了三四個網頁終於找到了

TL;DR

```
git format-patch -k --stdout ${commit_hash_1}..${commit_hash_2} | git am -3 -k
```

這個月第三次做這件事了

我們想解決的問題

1. 紀錄下自己的解決方案
 - 不要讓努力白費
2. 解決方案可以在不同的**AI**被重複使用
 - 不然換一個產品就又要重新累積了

困難點

1. 精確性: 給**AI**知識質量
 - 重要的知識/取得的知識
2. 傳輸性: 要可以方便傳輸
 - 不需要問一個問題就傳整個知識庫
3. 拓展性
 - 對於新展品可以快速的開發

02

其他人是如何解決的

Save to notion+API

Save to Notion

<

form

網路知識庫

⚙️

Aa Name

PATH 雲端課程

≡ tag

數位學習資源

▼

≡ 喜歡的原因

可以跟OHA雲端教室結合，也可以單獨使用

▼ 3 more fields

Clip Page

▼

雲端知識庫

Table List + Filter Sort 🔍 ... New ▼

Aa Name	Tags	🔗 URL	+ ...
 Webduino模擬器「瘋」神榜：用Web:Bit陣列學Youtube影片控制與玩賽果遊戲	IoT	https://harmonica80.blogspot.com/2022/12/webduinowebbityoutube.html	
 Sweet Home 3D 最 Easy 開講：Monster Mash 輕鬆將2D照片轉成3D模型，歡喜入住室內設計空間！	3D	https://harmonica80.blogspot.com/2023/01/sweet-home-3d-easy-monster-mash-2d3d.html	

Save to notion+API

1. 精確性 🤔:

- 可以幫資料庫加上**Tag**
- 但資料之間沒辦法有相關性

2. 傳輸性 👍:

- 有提供**API**可以分配

3. 拓展性 👎:

- 無法處理資料之間關聯

Supermemory



Supermemory

1. 精確性 👍:

- 是直接透過LLM+Rag回答

2. 傳輸性 👎:

- 只提供詢問LLM的回答

3. 拓展性 👎:

- 無法處理資料之間關聯
- 分類要自己手動搬

我們的解法

1. api server

- 寫插件調用api即可

2. 良好的架構設計, Project + Node

- 易傳輸
- 高拓展性

問完之後直接 呼叫API

```
user: guosw  
urltype: important  
/auto_sum
```

▶ 已與 api.driseam.com 交談 ▾

已與 api.driseam.com 交談 ▾

已與 api.driseam.com 交談 ▾

已與 api.driseam.com 交談 ▾

已與 api.driseam.com 交談 ▾

已與 api.driseam.com 交談 ▾

已與 api.driseam.com 交談 ▾

The information about cherry-picking from another repository has been successfully stored in your project "Git" under the node "From Another Repository."

Let me know if you need any modifications or additional details! 🚀

user: guosw
/answer
how can I cherry pick from other repository

已與 api.driseam.com 交談 ▾

已與 api.driseam.com 交談 ▾

已與 api.driseam.com 交談 ▾

已與 api.driseam.com 交談 ▾

To cherry-pick a commit from another repository, you have a few options:

1. Using `cherry-pick`

```
bash
git remote add other_repo <repo_url>
git fetch other_repo
git cherry-pick <commit_hash>
```

複製 編輯

2. Using `format-patch` and `am`

```
bash
git format-patch -1 <commit_hash> --stdout > commit.patch
git am < commit.patch
```

複製 編輯

3. Using `--git-dir`

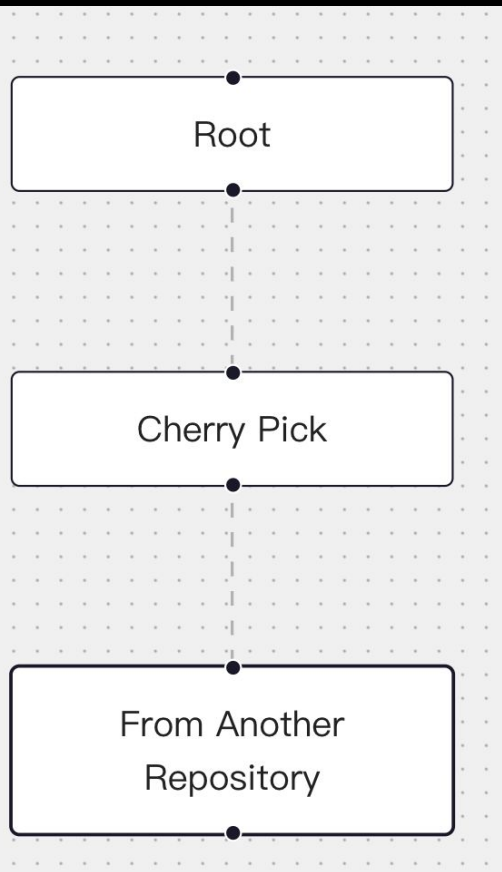
```
bash
git --git-dir=/path/to/source_repo/.git format-patch -1 <commit_hash> --stdout | git am
```

複製 編輯

Let me know if you need further clarification! 🚀

下次再問的時候就可以快速找到檔案

知識架構 - 即使資料很多也一目了然



From Another Repository

1. Using cherry-pick



```
1 git remote add other_repo <repo_url>
2 git fetch other_repo
3 git cherry-pick <commit_hash>
```

2. Using format-patch and am



```
1 git format-patch -1 <commit_hash> --stdout > commit.patch
2 git am < commit.patch
```

3. Using --git-dir

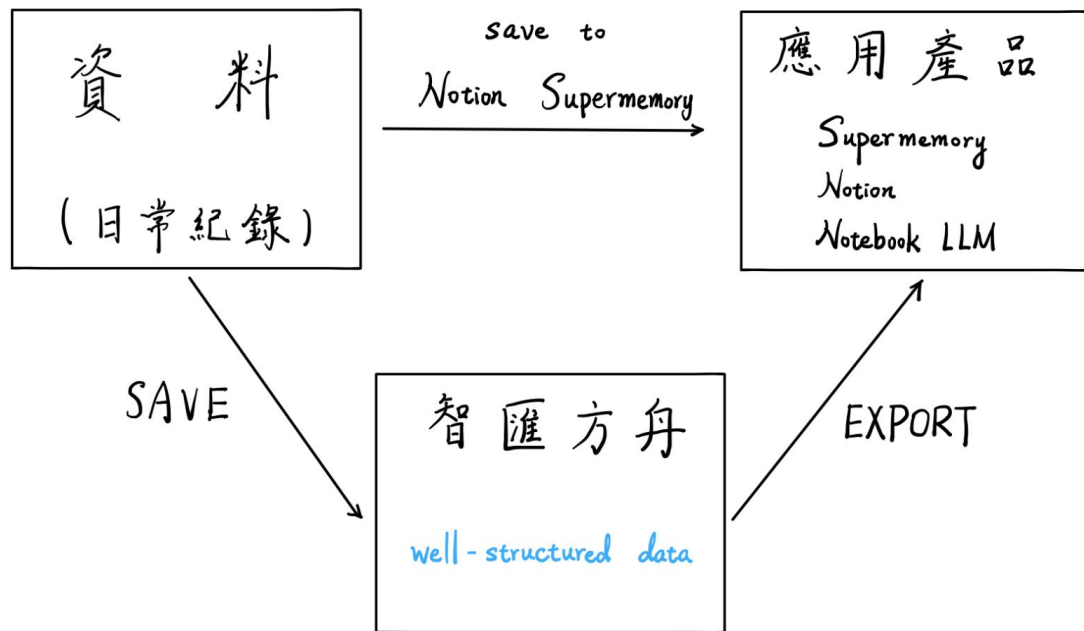


```
1 git --git-dir=/path/to/source_repo/.git format-patch -1 <commit_hash> --stdout | git am
```

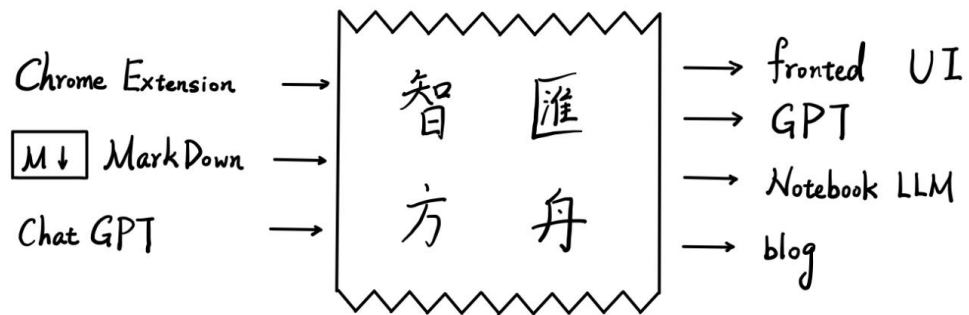
03

專案介紹

總覽



產品介紹



產品架構

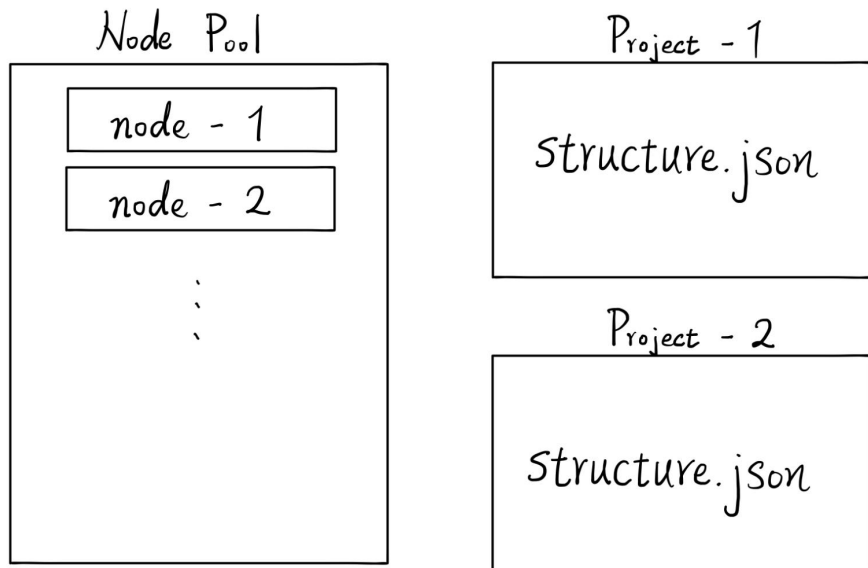
1. Node - 資料

- 一個知識單元
- 資料分Important, relate, other

2. Project - 結構

- 只負責node間的連接

產品架構



產品架構

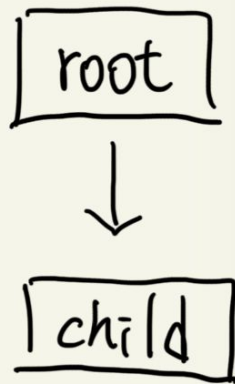
Node

```
{  
  nodeID: "  ",  
  Title: "  ",  
  important_data: [...],  
  relate_data: [...],  
  other_data: [...]  
}
```

Project

structure.json

```
{  
  nodeA: [node B]  
  node B: []  
  nodeTitle: [  
    nodeA: root  
    node B: child  
  ]  
}
```



架構優勢

1. Node 內分類

- 更有彈性

2. NodePool

- 不同的**project** 可以共用一個**node**

實際例子

Summarizer(chrome extension)

- 總結當前並加入智匯方舟內

創新: auto upload

- 自動將設定好的網站加入到
relate data

04

demo&未來展望

Demo

未來展望

API Server

1. 可以存文檔
2. 增加對node內文的搜索

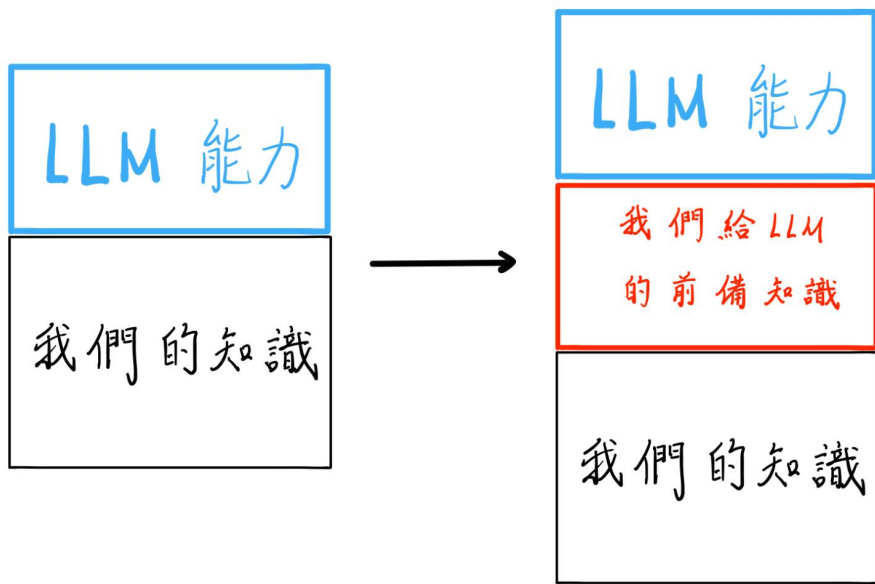
插件

- 自動生成個人的技術Blog

05

結論

總結



我們學到什麼

Summarizer

- javascript

前端顯示界面

- Vue, Vue flow

API Server

- fastapi

我們學到什麼

1. 如何快速迭代出可用的版本
2. 順著一開始的產品想法不斷擴大目標

“不要只專注於當前**AI**模型的不足之處，
因為這些短板可能隨時會被解決”

THANK YOU

COMPANY NAME

I want to use cherry pick from another repository
<repo_url>

1. please give me the one line code
2. please use git format-patch and git am
3. don't use git fetch

🔍 To apply a commit from another repository without using `git fetch`, you can create a patch from the desired commit in the source repository and then apply it to your current repository. Here's how you can do it:

1. One-line command:

```
bash 📋 複製 ✎ 編輯  
ce_repo>/.git format-patch -k -1 --stdout <commit_hash> | git am -3 -k
```

