ChatGPT

Supabase Handler Canvas

This canvas summarizes the `SupabaseHandler` class, shows where its methods are called, and highlights relevant Supabase CLI commands.

`SupabaseHandler.__init__` & `_initialize_client`

Code snippet

```
class SupabaseHandler:
    def __init__(self):
        self.client: Optional[Client] = None
        self._initialize_client()

    def _initialize_client(self) -> None:
        if env_config.has_supabase_config:
            self.client = create_client(env_config.supabase_url,
env_config.supabase_key)
            print(f"✓ Supabase client initialized: {env_config.supabase_url[:
50]}...")
        else:
            print("⚠ Supabase credentials not found in environment variables")
```

This uses the official Supabase Python initialization pattern [1]. The handler is instantiated in `AdvancedWebCrawler.__init__` [2].

`store_crawl_results`

Code snippet

```
def store_crawl_results(self, results: List[Dict[str, Any]]) -> int:
    if not self.is_available:
        return 0
    stored_count = 0
    for result in results:
        url = result.get('url')
        raw_markdown = result.get('raw_markdown')
        analysis = result.get('analysis')
        if not url or not raw_markdown:
            continue
```

```
            data = {'url': url, 'content': raw_markdown[:10000]}
            if analysis:
                title = analysis.get('title', 'No title')
                summary = analysis.get('summary', 'No summary')
                analysis_header = f"TITLE: {title}\nSUMMARY: {summary}\n\n"
                data['content'] = analysis_header + data['content']
            response = self.client.table('pages').insert(data).execute()
            if response.data:
                stored_count += 1
        return stored_count
```

This method prepares each result and inserts it into the Supabase `pages` table [3]. It is invoked from `crawl_and_store_in_supabase` in `async_crawler.py` [4].

`check_connection`

Code snippet

```
def check_connection(self) -> bool:
    if not self.is_available:
        return False
    try:
        response = self.client.table('pages').select('id').limit(1).execute()
        print("✓ Supabase connection verified")
        return True
    except Exception:
        return False
```

This method performs a simple query to verify the database connection [5]. It is not used elsewhere but can be called manually to test connectivity.

Terminal commands from official docs

The Supabase CLI allows you to manage projects locally. According to the official docs:

- Use `supabase init` to initialize a new local project and create a `supabase` folder [6].
- Run `supabase start` to launch the local Supabase services (Postgres, API, Studio, etc.) [7].
- When finished, stop the stack without resetting your database using `supabase stop` [8].

These commands should be run in your terminal before using the Python client to ensure the local services are running.

---

1 3 5 supabase_handler.py
https://github.com/Gaya56/supa-crawl/blob/main/src/storage/supabase_handler.py

2 4 async_crawler.py
https://github.com/Gaya56/supa-crawl/blob/main/src/crawlers/async_crawler.py

6 7 8 Supabase CLI | Supabase Docs
https://supabase.com/docs/guides/local-development/cli/getting-started