I'll help you understand the relationship between the admin portal, prompt files, and the running application. Let me examine the relevant files to see how prompts are loaded and whether they're dynamically refreshed.Now let me examine the main application file to see how it loads and uses the prompt files:Let me examine the prompt loading mechanism to understand how prompts are loaded in the application:Based on my analysis of your code, I can answer your question about whether prompt file updates from the admin portal are dynamically reflected in the running application.

Answer: YES, prompt updates are dynamically reflected without manual reload

Here's how it works:

Dynamic Loading Mechanism

1. **No Caching**: The prompt_loader.py module reads prompt files directly from disk every time a prompt is requested:

```
def load_prompt(prompt_name):
    # Reads file fresh from disk each time
    with open(prompt_file, 'r', encoding='utf-8') as file:
        content = file.read().strip()
    return content
```

2. **File-Based Storage**: The admin portal saves changes directly to the prompt files in the prompts directory:

```
def save_prompt_file(filename, content):
    filepath = os.path.join(PROMPTS_DIR, filename)
    with open(filepath, 'w', encoding='utf-8') as f:
        f.write(content)
```

3. **Real-Time Updates**: When app_new.py calls `load_prompt()` function (which happens on every API request that needs a prompt), it reads the current file content from disk.

How It Works in Practice

1. **Admin Portal Update**: When you edit a prompt in the admin portal and save it, the file is immediately updated on disk
2. **Next API Call**: The next time the running application needs that prompt (e.g., for symptom analysis, photo analysis, etc.), it calls `load_prompt()`
3. **Fresh Content**: The `load_prompt()` function reads the updated file content from disk
4. **Immediate Effect**: The new prompt is used right away in the LLM API call

Example Flow

```
1. Admin portal saves updated aadhaar_analysis.txt
2. User uploads Aadhaar in running app
3. App calls load_prompt("aadhaar_analysis")
4. Function reads the newly updated file
5. Updated prompt is sent to OpenAI API
```

No Application Restart Required

The application does NOT need to be reloaded manually because:

- No prompt content is stored in memory/cache
- No application-level variables hold prompt content
- Each request reads fresh content from the file system

This design ensures that prompt updates from the admin portal take effect immediately in the running application.