

Мотивація

Я з дівчиною з 2021 року займаюся розвитком проекту PollyArt Store (<https://pollyart.store/ua>). Це мій пет-проект у сфері електронної комерції, який базується на CMS OpenCart з безліччю покращень, підмодулей, модулів, різноманітних системних розширень, архітектурних поліпшень та баг фіксів. Оскільки я один займаюся розробкою як бекенду так і фронтенду, мені б не завадила професійна допомога або інструменти, які б стали мені у нагоді стосовно розвитку цього проекту. І я вирішив що буду робити асистента з програмування під свій проект закриваючи свої потреби. Теоретично це прискорить роботу над проектом, спростить розробку, дозволить автоматизувати рутинні завдання та взагалі допоможе мені в розвитку використання інструментів ШІ та збільшить мою експертизу в різних аспектах інформаційних технологій. Частиною системи агента з програмування так курсовий проект по Генеративному ШІ є [NES AI Log Analyzer](#)

Вступ

Проблема: Розробка та підтримка складних E-commerce платформ, таких як сильно модифікований OpenCart, силами одного розробника є надзвичайно трудомістким процесом. Ключовою частиною цього процесу є аналіз лог-файлів для виявлення та виправлення помилок, що традиційно вимагає значних витрат часу та глибокої експертизи.

Існуючі підходи та їхні недоліки:

1. **Ручний аналіз:** Використання системних утиліт, як-от `grep`, `awk`, або ручний перегляд логів. Цей підхід є неефективним, повільним і не масштабується.
2. **Системи агрегації логів (ELK Stack, Graylog):** Потужні інструменти для збору, пошуку та візуалізації логів. Однак вони не надають семантичного розуміння помилок і не пропонують шляхів їх вирішення, а лише спрощують пошук.
3. **Класичне машинне навчання:** Підходи, що базуються на класифікації тексту (наприклад, Naive Bayes, SVM), можуть групувати схожі помилки, але не здатні аналізувати першопричину та генерувати конкретні рекомендації щодо виправлення коду.

Релевантне поле (Сучасний підхід): З появою великих мовних моделей (LLM), таких як GPT-4, Llama, та спеціалізованих кодових моделей (наприклад, [Qwen3-Coder](#)), з'явилася можливість створити інструменти нового покоління. Такі асистенти, як GitHub Copilot, продемонстрували ефективність LLM у допомозі розробникам. Дана робота застосовує цей підхід до вузькоспеціалізованої задачі — створення персоналізованого AI-асистента для аналізу логів конкретного E-commerce проєкту.

Опис

Архітектура системи: Проєкт є CLI-застосунком на Python, який використовує фреймворк **LangChain** для оркестрації взаємодії з великими мовними моделями. Архітектура підтримує два режими роботи:

1. **Локальний:** Використання відкритих LLM через **Ollama** (за замовчуванням [Qwen3-Coder:30b](#)), що забезпечує приватність даних та контроль над середовищем.
2. **Хмарний:** Використання API **OpenAI** (наприклад, [GPT-4o](#)) для отримання доступу до найпотужніших моделей та зменшення часу обробки запитів без оновлення існуючого заліза.
3. **Хмарних провайдерів** легко масштабувати за рахунок існуючих інтеграцій **LangChain** розширюючи метод `init_llm(...args)`

Алгоритм роботи:

1. **Ініціалізація та конфігурація:** Скрипт зчитує параметри командного рядка ([argparse](#)) та конфігурацію з [.env](#) файлу, визначаючи модель, шлях до лог-файлу та режим роботи.
2. **Парсинг логів:**
 - **Загальний режим:** Лог-файл обробляється як єдиний текстовий документ.
 - **Спеціалізований режим (`--oc True`):** Активується кастомний парсер для логів OpenCart/NES, який виділяє з тексту окремі помилки, їхній час, стек викликів та контекст.
3. **Взаємодія з LLM (Промпт-інжиніринг):** Формується детальний промпт, який містить:
 - Системну інструкцію (наприклад, "Ти — досвідчений PHP-розробник, що спеціалізується на...").
 - Уривок логу, що аналізується.
 - Вимоги до формату відповіді

[illegible]

[illegible]

[illegible]

```
\u0432 \u043a\u043e\u0434\u0456
\u0432\u0456\u0434\u0431\u0443\u0432\u0430\u0454\u0442\u044c\u0441\u0444
\u0432\u0438\u043a\u043b\u043a `fopen()` \u0414
\u0434\u0435\u0440\u043a\u043e\u0434\u0432\u0430\u0439\u0441\u0444,
\u0449\u043e \u0432\u0456\u0434 \u0443\u0441\u0434\u0432\u0434\u0432\u0434\u043e\u0432\u0434\u043e
\u0432\u0456\u0434\u043a\u043a\u0440\u0438\u0432\u0430\u0454
\u0444\u0430\u0439\u043b:\n\n``php\n$file_resource = fopen($filename,
'r+');\nif ($file_resource === false) {\n    //
\u0415\u0431\u0440\u043e\u0431\u0431\u043a\u0430
\u0434\u043e\u043c\u043b\u043a\u0438:
\u0434\u0430\u0434\u0438\u043a\u043b\u0430,
\u043b\u043e\u0433\u0443\u0432\u0432\u0434\u0434\u0444 \u0430\u0431\u0431\u043e
\u0432\u0438\u043a\u0438\u0434\u0434\u0434\u0444
\u0432\u0438\u0438\u0444\u0442\u043a\u0443\n    throw new Exception(\n"Cannot
open cache file: $filename\n");\n}\n``\n\n#### \u0410\u0440\u043e\u043a 3:
\u0414\u0434\u0440\u0432\u0432\u0432\u0456\u0440\u043a\u0430
\u0434\u0440\u0430\u0432\u0432 \u0443\u0442\u0442\u0443\u0443\u0434\u0434\u0434\u0434
\u0434\u043e\u0438 \u0438\u0440\u0440\u0435\u0442\u0456\u0457
\u043a\u0435\u0443\u0443\u0434\u0434\u043a\u043e\u0434\u0432\u0434\u0430\u0430\u0430
39\u0441\u0444, \u0449\u043e
\u0434\u0438\u0440\u0435\u043a\u0442\u043e\u0440\u0456\u0444
`/system/library/Cache/` \u0456\u0441\u0434\u0443\u0454 \u0442\u0430
\u043c\u0430\u0454 \u0434\u0440\u0432\u0432\u0432 \u0434\u0430
\u0437\u0430\u0434\u0431 \u043b\u0444
\u0432\u0435\u0431-\u0441\u0440\u0440\u0432\u0435\u0440\u0430
(\u0434\u0430\u0434\u0440\u0438\u043a\u043b\u0430\u0430\u0434, `chmod 755`
\u0430\u0431\u043e ``775`).\n\n#### \u0410\u0440\u043e\u043a 4:
\u0411\u043e\u0433\u0443\u0432\u0432\u0434\u0434\u0434\u0434\u0434\u0444
\u0434\u043e\u043c\u043b\u043a\u043e\u0431\u0431\u0431\u0434\u0430\u0430\u0430\u0439
\u043b\u043e\u0433\u0443\u0432\u0432\u0434\u0434\u0444
\u0434\u0434\u043e\u043c\u043b\u043e\u043a \u0434\u0440\u0438
\u0432\u0456\u0434\u043a\u0440\u0442\u0442\u0456
\u0444\u0430\u0439\u043b\u0456\u0432, \u0449\u0449\u043e\u0431
\u043b\u0435\u0443\u0448 \u0443\u0431\u0443 \u0443\u043b\u043e
\u0432\u0456\u0434\u043a\u0441\u0443\u0442\u0432\u0432\u0432\u0432\u0438
\u0434\u0434\u0432\u0432\u0432\u0432\u0432\u0432\u0432\u0432\u0432
\u0434\u0434\u0432\u0432\u0432\u0432\u0432\u0432\u0432 \u0430\u0438
\u0434\u043c\u0430\u0439\u0431\u0443\u0443\u0442\u0434\u044c\u043c\u043e\u043c\u0443:\n\n``php\n
nif (($file_resource = fopen($filename, 'r+')) === false) {\n
error_log(\n"Failed to open cache file: $filename\n");\n    return
false;\n}\n``\n\n---\n\n####
\u0414\u0456\u043a\u0443\u0443\u043c\u043e\u043a:\n\u0414\u0434\u043e\u043c\u043b\u0430\u0438\u0438\u0438\u0438\u0438
```

```
"additional_kwargs": {},
"response_metadata": {
  "model": "qwen3-coder:30b",
  "created_at": "2025-08-28T14:22:58.767234779Z",
  "done": true,
  "done_reason": "stop",
  "total_duration": 110022688435,
  "load_duration": 27834774,
  "prompt_eval_count": 1099,
  "prompt_eval_duration": 5143408601,
  "eval_count": 1088,
  "eval_duration": 104850912553,
  "model_name": "qwen3-coder:30b"
},
"id": "run--747b2fa0-e403-4893-ae88-24df0c20f5e6-0",
"usage_metadata": {
  "input_tokens": 1099,
  "output_tokens": 1088,
  "total_tokens": 2187
}
```

Відповідь у текстовому файлі (тільки вивід мовної моделі без метаданих):

1. Аналіз помилки:

Ця помилка — **PHP Fatal error** типу ``Uncaught TypeError``. Вона виникає, коли функція ``flock()`` отримує неправильний тип аргументу. Конкретно:

```
> `flock(): Argument #1 ($stream) must be of type resource, bool given`
```

Це означає, що в коді на рядку 35 файлу ``/system/library/Cache/File.php`` викликається функція ``flock()``, але перший аргумент (файл-ресурс) — це **bool** (наприклад, ``false``), а не **resource**, який очікується.

Тобто, код намагається заблокувати файл для читання/запису, але він не вдалося відкрити файл і отримав ``false`` замість ресурсу.

2. Ймовірна причина:

Найбільш імовірною причиною є те, що в методі ``Cache\File->get()`` (або де-небудь в процесі роботи з кешем) відбувається спроба відкрити файл через ``fopen()``, але цей виклик повертає ``false`` — тобто файл не вдалося відкрити. Потім код намагається передати цей ``false`` у функцію ``flock()``, що призводить до фатальної помилки.

Можливі причини:

- Немає прав на запис у директорію кешу.
- Директорія кешу не існує або недоступна.
- Файл кешу пошкоджений або заблокований.
- Виникла помилка під час відкриття файлу через ``fopen()``.

3. Рекомендації до виправлення:

Крок 1: Перевірка відкриття файлу

У файлі ``/system/library/Cache/File.php`` на рядку 35 має бути код, який викликає ``flock($file_resource, ...)``.

Переконайся, що перед цим викликом перевіряється, чи ``$file_resource`` — це дійсно ресурс:

```
```php
if (is_resource($file_resource)) {
 flock($file_resource, LOCK_EX);
} else {
```



```

 // Логування або обробка помилки
 error_log("Failed to open cache file for locking.");
}
...

Крок 2: Перевірка `fopen()`
Де в коді відбувається виклик `fopen()` – переконайся, що він успішно відкриває
файл:

```php
$file_resource = fopen($filename, 'r+');
if ($file_resource === false) {
    // Обробка помилки: наприклад, логування або викидання винятку
    throw new Exception("Cannot open cache file: $filename");
}
...

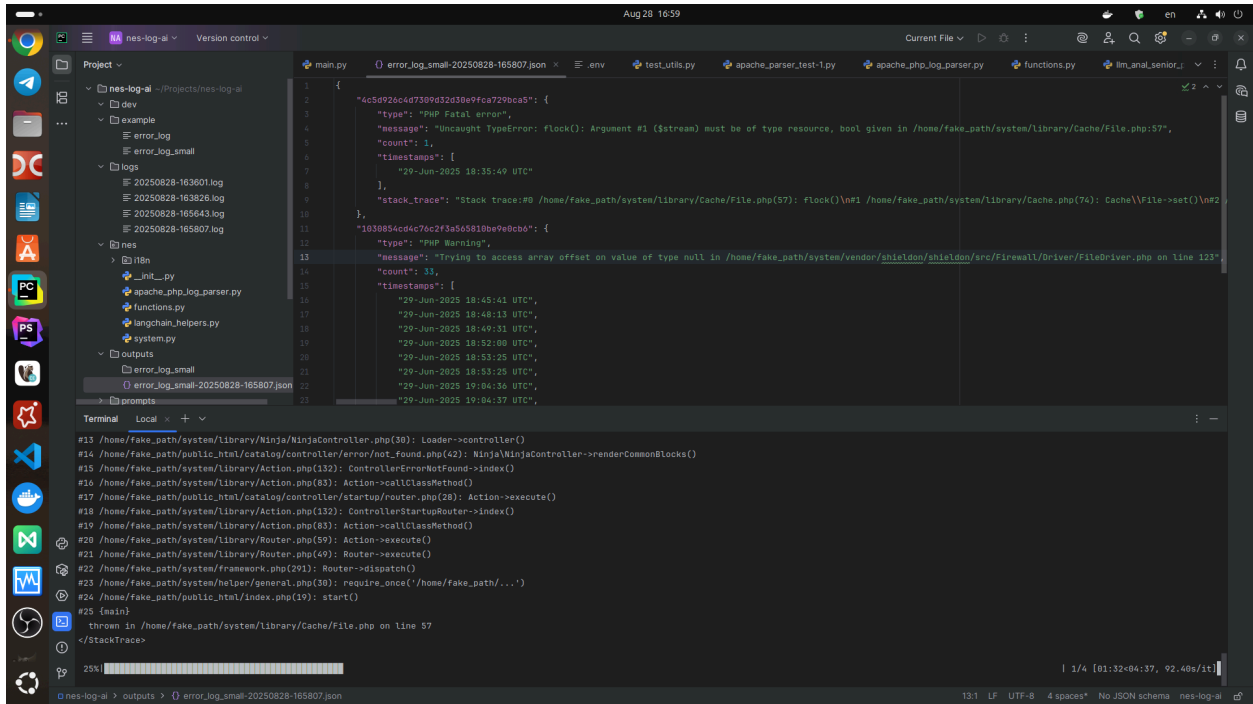
#### Крок 3: Перевірка прав доступу до директорії кешу
Переконайся, що директорія `/system/library/Cache/` існує та має права на запис
для веб-сервера (наприклад, `chmod 755` або `775`).

#### Крок 4: Логування помилок
Додай логування помилок при відкритті файлів, щоб легше було відстежувати
проблеми в майбутньому:

```php
if (($file_resource = fopen($filename, 'r+')) === false) {
 error_log("Failed to open cache file: $filename");
 return false;
}
...

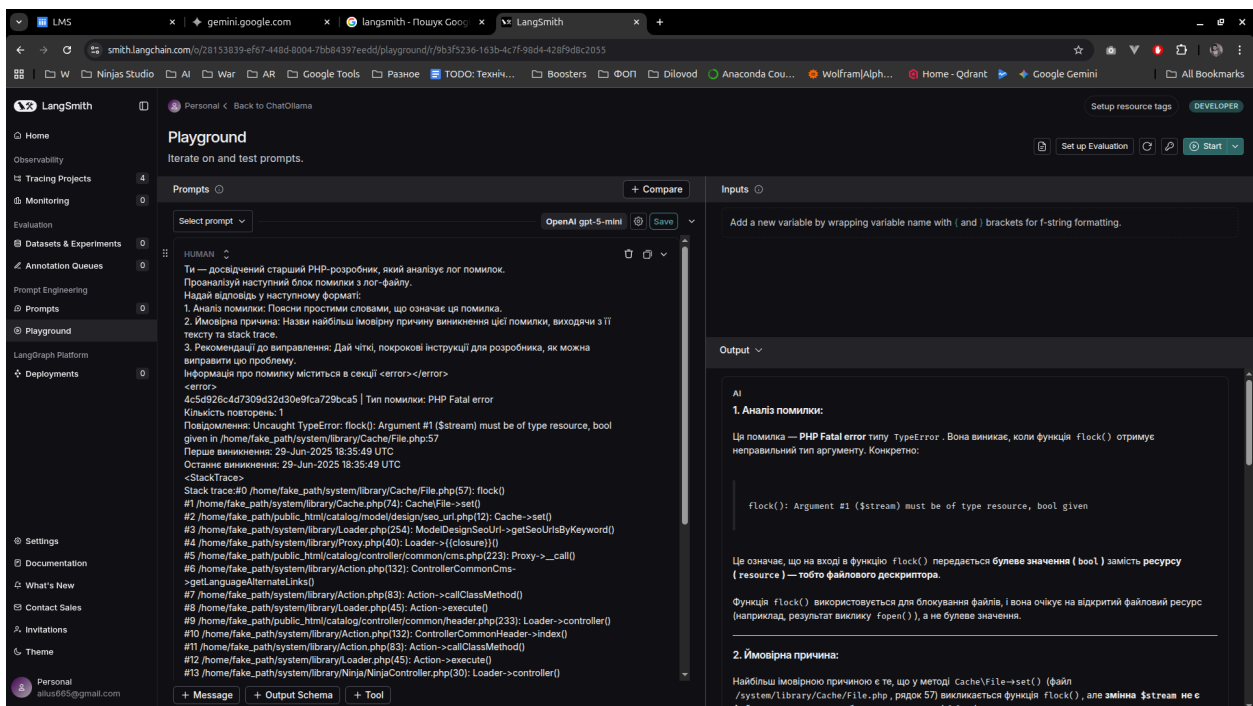
Підсумок:
Помилка виникає через спробу передати `false` у функцію `flock()`, яка очікує
ресурс. Це означає, що файл не був успішно відкритий. Виправити потрібно
перевіркою результату `fopen()` перед використанням `flock()`.

```



Малюнок 1: Приклад роботи скрипта в терміналі

<https://github.com/zeratulus/nes-log-ai/blob/master/docs/000-work-example.png>



Малюнок 2: LangSmith Playground з результатами роботи AI Log Analyzer

<https://github.com/zeratulus/nes-log-ai/blob/master/docs/001-langsmith-playground-uk.png>

Name	Input	Output	Error	Start Time	Latency	Dataset	Annotation Queue	Tokens	Cost
ChatOllama	human: Ти — досвідч...	ai: ## 1. Аналіз...		8/29/2025, 11:39:32...	109.73s			2,127	
ChatOllama	human: Ти — досвідч...	ai: ## 1. Аналіз...		8/29/2025, 11:37:44...	108.61s			2,187	
ChatOllama	human: Ти — досвідч...	ai: 1. **Аналіз по...		8/29/2025, 11:36:23...	80.84s			1,293	
ChatOllama	human: Ти — досвідч...			8/29/2025, 11:34:21...	122.15s			2,066	
ChatOllama	human: Ти — досвідч...	LLMResult	KeyboardInterrup...	8/29/2025, 11:21:35...	22.32s			0	
ChatOllama	human: Ти — досвідч...			8/29/2025, 11:18:50...	165.14s			2,066	

**Малюнок 3:** LangSmith лістинг запитів. Приклад логування запитів та загальна інформація по роботі скрипта.

<https://github.com/zaratulus/nes-log-ai/blob/master/docs/002-langsmith-uk.png>

Input	Output
<p>human: Ти — досвідчений старший PHP-розробник, який аналізує лог помилок. Проаналізуй наступний блок помилки з лог-файлу. Наддай відповідь у наступному форматі:</p> <ol style="list-style-type: none"> <li>Аналіз помилки: Поясни простими словами, що означає ця помилка.</li> <li>Імовірна причина: Назви найбільш імовірну причину виникнення цієї помилки, виходячи з її тексту та stack trace.</li> <li>Рекомендації до виправлення: Дай чіткі, покрокові інструкції для розробника, як можна виправити цю проблему.</li> </ol> <p>Інформація про помилку міститься в секції &lt;error&gt;&lt;/error&gt;</p> <pre>&lt;error&gt; 36dda802e4b7c6a4fb3887ed8547aebd   Тип помилки: PHP Fatal error Кількість повторень: 1 Повідомлення: Uncaught TypeError: flock(): Argument #1 (\$stream) must be of type resource, bool given in /home/fake_path/system/library/Cache/File.php:57 Перше виникнення: 29-Jun-2025 19:22:43 UTC Останнє виникнення: 29-Jun-2025 19:22:43 UTC &lt;/error&gt;</pre> <p>Stack trace: #0 /home/fake_path/system/library/Cache/File.php(57): flock()  #1 /home/fake_path/system/library/Cache.php(74): Cache/File-&gt;set()  #2 /home/fake_path/public_html/catalog/model/design/seo_url.php(12): Cache-&gt;set()  #3 /home/fake_path/system/library/Loader.php(254): ModeDesign/SeoUrl-&gt;getSeoUrlByKeyword()  #4 /home/fake_path/system/library/Proxy.php(40): Loader-&gt;(closure)()</p>	<p>## 1. Аналіз помилки:</p> <p>Ця помилка — **PHP Fatal error** типу 'Uncaught TypeError'. Вона означає, що код намагався передати неправильний тип даних у функцію 'flock()'. Конкретно: &gt; flock(): Argument #1 (\$stream) must be of type resource, bool given</p>

**Малюнок 4:** LangSmith запит. Приклад логування запиту та інформація по роботі скрипта.

<https://github.com/zaratulus/nes-log-ai/blob/master/docs/003-langsmith-uk.png>

# Результати

## Ключові метрики ефективності:

1. **Скорочення часу на зневадження (Time Saved):** Це найголовніший показник. Можна провести порівняльний аналіз: "Час на діагностику типової помилки вручну (в середньому 15-20 хв) проти часу на отримання відповіді від AI-асистента (в середньому 1-2 хв на моєму залізі)".
2. **Точність рекомендацій (Accuracy):** Асистент доволі непогано обробляє помилки використовуючи модель за замовчуванням qwen3-coder:30b. Протестував на синтетичному Java лозі та PHP OpenCart.
3. **Продуктивність (Performance):** на GPU NVIDIA RTX 3060 обробка великого файлу лога зайняла близько доби. 938 унікальних записів логу різної складності на "живих" помилках реальної системи.

## Сильні сторони:

- **Швидкість:** Радикальне прискорення процесу аналізу рутинних помилок.
- **Контекстне розуміння:** Завдяки спеціалізованому парсеру обробка групованих помилок займає менше часу прибираючи деякі дублювання, також корисно впливає деталізація мови програмування та специфіки проекту в режимі обробки.
- **Гнучкість:** Можливість вибору між локальними та хмарними моделями та можливість обробки різних форматів логу.

## Слабкі сторони:

- **Високі вимоги до ресурсів:** Для локальної роботи потрібне потужне апаратне забезпечення 50Gb RAM + 10VRAM.
- **"Галюцинації" LLM:** Існує ризик отримання неправильних або нерелевантних порад, особливо для складних або нетипових помилок.
- **Залежність від якості логів:** Якщо логи неінформативні, аналіз буде неточним.

# Висновки

**Підсумок:** У ході роботи було розроблено AI-асистента для аналізу логів, орієнтованого на потреби соло-розробника E-commerce проекту на базі OpenCart. Інструмент успішно

автоматизує процес діагностики помилок, використовуючи сучасні LLM. На даному етапі розробки, це значно скорочує час на розробку та підтримку.

#### **Можливі напрямки для покращення:**

1. **Інтерактивний режим:** Розвиток системи до формату чат-бота, де можна вести діалог з лог-файлом ("Покажи всі помилки цього типу", "Що передувало цій помилці?").
2. **Проактивний моніторинг:** Створення сервісу, який би в реальному часі відстежував логи та автоматично надсилав сповіщення з аналізом критичних помилок.
3. **Автоматичне виправлення коду:** Найбільш амбітний напрямок — навчити систему не тільки пропонувати рішення, а й генерувати готовий код (у форматі `.diff` чи pull request) для виправлення помилки.
4. **Розширення бази знань RAG:** Підключення до векторної бази даних як і логів так й усієї кодової бази проєкту та офіційної документації OpenCart для надання ще більш точних та контекстуалізованих відповідей.
5. **Розширення асортименту хмарних моделей:** Google Gemini 2.5 Pro, Claude Sonnet 4 та інших.
6. **Розробка специфічних парсерів під типи логів.** Дозволить скоротити час обробки та надасть можливість фільтрувати непотрібні лог рівні.