

AI PROMPTS

1. OVERVIEW & PURPOSE

AI prompts are instructions or input written in natural language that users give to an artificial intelligence system to perform a specific task. These tasks can range from generating text, writing code, creating images, translating languages, analyzing data.

2. AI Used

Chatgpt

3. Prompt Used

a. Proposing Document Outline

To kick off our documentation, we leveraged ChatGPT to suggest a clear outline tailored to our objectives. This gave us a strong foundation to build a well-structured and focused report.

Nhóm tôi được yêu cầu viết use-case specification để đặc tả các chức năng của website.
Hay cho tôi 1 outline mẫu để đặc tả 1 use-case

Dưới đây là một outline mẫu chuẩn để đặc tả một Use Case Specification trong tài liệu thiết kế hệ thống, đặc biệt phù hợp với dự án website (ví dụ: thương mại điện tử, đặt vé, quản lý học sinh,...):

■ Use Case Specification: [Tên Use Case]

1. Use Case Name

Tên chức năng chính.
Ví dụ: Đăng nhập, Đặt hàng, Quản lý sản phẩm,...

2. Brief Description

Mô tả ngắn gọn mục tiêu và vai trò của use case.
Ví dụ: Chức năng cho phép người dùng nhập tài khoản và mật khẩu để truy cập vào hệ thống.

3. Actors

Danh sách các actor (vai trò tương tác).
Ví dụ: Người dùng, Quản trị viên, Khách truy cập,...

8. Exceptions (Ngoại lệ)

Các lỗi bất thường hệ thống có thể gặp.

Ví dụ:

- Kết nối cơ sở dữ liệu bị lỗi.
- Mất kết nối internet.

9. Special Requirements (Yêu cầu đặc biệt)

Nếu có các ràng buộc về bảo mật, hiệu suất, hoặc UI/UX.

Ví dụ:

- Phải có mã hóa mật khẩu.
- Hiển thị thông báo lỗi rõ ràng.

10. Related Use Cases

Danh sách các use case có liên quan (nếu có).

Ví dụ: Đăng ký, Quên mật khẩu.

✓ Gợi ý ví dụ:

Nếu use-case là "Đặt hàng" cho web bán hàng:

- **Actors:** Người dùng
- **Pre-condition:** Đã đăng nhập và có sản phẩm trong giỏ
- **Main Flow:**
 1. Người dùng nhấn nút "Thanh toán"
 2. Nhập địa chỉ giao hàng
 3. Chọn phương thức thanh toán
 4. Xác nhận đơn
 5. Hệ thống lưu đơn và hiển thị mã đơn

Bạn muốn tôi giúp viết đặc tả chi tiết cho một use case cụ thể trong hệ thống của bạn không? (ví dụ: "Quản lý sản phẩm" hay "Xem đơn hàng")?

📄 🗑️ 🔍 ✎️ 🔄 📌

4. Pre-conditions

Điều kiện cần có trước khi Use Case có thể được thực hiện.

Ví dụ: Người dùng đã đăng ký tài khoản.

5. Post-conditions

Kết quả sau khi Use Case kết thúc (thành công hoặc thất bại).

Ví dụ: Người dùng đăng nhập thành công và được chuyển đến trang chính.

6. Main Flow (Basic Flow)

Các bước chính xảy ra khi Use Case diễn ra suôn sẻ.

Ví dụ:

1. Người dùng truy cập trang đăng nhập.
2. Nhập email và mật khẩu.
3. Hệ thống xác thực thông tin.
4. Nếu hợp lệ, chuyển đến trang chính.

7. Alternative Flows (Các luồng thay thế)

Các tình huống phát sinh ngoài luồng chính.

Ví dụ:

- **7.1 Email sai định dạng:**
Hệ thống hiển thị thông báo lỗi định dạng email.
- **7.2 Sai mật khẩu:**
Hệ thống báo lỗi và cho phép nhập lại.

b. Giving Examples Of Contents

To support the development of our documentation, we utilized ChatGPT to generate example content outlines. This approach provided us with a clear structure and ensured alignment with our project objectives. By reviewing these examples, we were able to identify relevant sections and prioritize information effectively. The result was a well-organized document built on a coherent foundation.

Hãy viết mẫu cho tôi về chức năng đăng nhập của người dùng với website bán hàng.

Dưới đây là mẫu Use Case Specification hoàn chỉnh cho chức năng "Đăng nhập người dùng" trong một website bán hàng:

■ Use Case Specification: Đăng nhập người dùng

1. Use Case Name

Đăng nhập người dùng

2. Brief Description

Use case này cho phép người dùng đã có tài khoản trên hệ thống website bán hàng đăng nhập bằng email và mật khẩu để truy cập các chức năng dành riêng cho người dùng, như mua hàng, xem đơn hàng và chỉnh sửa thông tin cá nhân.

3. Actors

- Người dùng (User)
- Hệ thống xác thực (Authentication System)

4. Pre-conditions

- Người dùng đã đăng ký tài khoản hợp lệ.
- Người dùng đang ở trên trang đăng nhập của website.

5. Post-conditions

- Nếu đăng nhập thành công: người dùng được chuyển đến trang chính hoặc trang họ yêu cầu truy cập ban đầu.
- Nếu thất bại: người dùng nhận được thông báo lỗi tương ứng.

6. Main Flow (Basic Flow)

1. Người dùng truy cập trang đăng nhập.
2. Nhập email và mật khẩu.
3. Nhấn nút "Đăng nhập".
4. Hệ thống kiểm tra thông tin xác thực.
5. Nếu thông tin hợp lệ, hệ thống đăng nhập người dùng và chuyển đến trang chính.

7. Alternative Flows

7.1. Email không đúng định dạng

- Hệ thống hiển thị thông báo: "Email không đúng định dạng."

7.2. Tài khoản không tồn tại

- Hệ thống thông báo: "Tài khoản không tồn tại. Vui lòng đăng ký."

7.3. Mật khẩu không đúng

- Hệ thống hiển thị: "Mật khẩu không chính xác. Vui lòng thử lại."

- **Lỗi kết nối hệ thống:**
Hệ thống thông báo: "Không thể kết nối tới máy chủ. Vui lòng thử lại sau."
- **Hệ thống bị lỗi khi xác thực:**
Hệ thống hiển thị thông báo lỗi chung và ghi log nội bộ.

- Mật khẩu được mã hóa (hash) khi lưu và kiểm tra.
- Sau 5 lần đăng nhập sai liên tiếp, tài khoản bị khóa tạm thời trong 15 phút.
- Giao diện phải hiển thị rõ thông báo lỗi, hỗ trợ nhập lại dễ dàng.
- Hỗ trợ chức năng "Hiện/ẩn mật khẩu" để tăng tính lợi cho người dùng.

- Đăng ký tài khoản
- Quên mật khẩu
- Đăng xuất
- Chỉnh sửa thông tin cá nhân

As part of the documentation process, we used AI to translate content from Vietnamese to English and vice versa. This ensured consistent and accurate language conversion while preserving the original meaning and tone. Overall, the approach contributed to greater accessibility and clarity in our documentation.



♦ c. Use-case model (10 points)

📌 Mục tiêu:

Xây dựng sơ đồ mô hình Use-case diagram thể hiện toàn bộ chức năng chính của hệ thống.

✅ Cần giải quyết:

1. Use-cases:
 - Các chức năng chính (đăng nhập, tìm kiếm, đặt hàng, thanh toán,...)
2. Actors:
 - Ai tương tác với hệ thống? (User, Admin, Guest,...)
3. Relationships:
 - Sử dụng các mối quan hệ include, extend, hoặc generalization.
4. Thiết kế sơ đồ:
 - Dùng StarUML, Visio, hoặc công cụ khác.
 - Chèn sơ đồ vào file Use-case Specification.

♦ d. Use-case specification (35 points)

📌 Mục tiêu:

Mô tả chi tiết từng Use-case trong sơ đồ trên.

✅ Cần giải quyết:

Với mỗi Use-case, cần điền rõ:

Mục cần viết	Nội dung
Use-case name	Tên chức năng
Actors	Các actor tham gia
Pre-conditions	Điều kiện ban đầu (VD: người dùng phải đăng nhập trước khi đặt hàng)
Basic Flow	Luồng xử lý chính của chức năng
Alternative Flows	Luồng phụ, rẽ nhánh (VD: sai mật khẩu, hết hàng...)
Post-conditions	Kết quả sau khi thực hiện Use-case
(Tùy chọn) Exceptions	Nếu có lỗi hệ thống, phải xử lý ra sao

📌 Lưu ý:

- Mỗi Use-case bắt buộc phải có Basic Flow.
- Số lượng Use-case nên từ 6–10 tùy độ phức tạp.

d. Generating Code

We used ChatGPT to assist in modifying, adding, and refactoring existing code structures based on evolving project needs. The model provided reliable code suggestions aligned with the current logic and architecture, enabling smoother integration and maintenance. This process minimized manual errors and ensured consistency across modules. Ultimately, it enhanced code quality and streamlined the development workflow.

```
const mongoose = require("mongoose");

const couponSchema = new mongoose.Schema({
  percent: { type: Number, required: true, min: 1, max: 100 },
  code: { type: String, required: true, unique: true },
}, { timestamps: true });

module.exports = mongoose.model("coupon", couponSchema);

dựa vào mô hình dữ liệu của tôi hãy viết hàm để thêm mới dữ liệu
```



Dưới đây là hàm thêm mới dữ liệu (coupon) vào MongoDB bằng Mongoose, dựa trên `couponSchema` mà bạn đã cung cấp:

✓ Hàm thêm mới `coupon` bằng `async/await`

js Copy Edit

```
const Coupon = require("../models/coupon"); // đường dẫn cần chỉnh đúng

/**
 * Thêm một coupon mới vào database.
 * @param {String} code - Mã giảm giá (unique).
 * @param {Number} percent - Phần trăm giảm (1-100).
 * @returns {Object} Kết quả thành công hoặc thông báo lỗi.
 */
async function createCoupon(code, percent) {
  try {
    // Kiểm tra đầu vào
    if (!code || typeof percent !== 'number') {
      throw new Error("Thiếu mã code hoặc percent không hợp lệ.");
    }

    // Tạo coupon mới
    const newCoupon = new Coupon({ code, percent });

    // Lưu vào MongoDB
    const savedCoupon = await newCoupon.save();
```

```
    console.log("Coupon đã được tạo:", savedCoupon);
    return savedCoupon;
  } catch (err) {
    if (err.code === 11000) {
      console.error("Coupon code đã tồn tại.");
    } else {
      console.error("Lỗi khi tạo coupon:", err.message);
    }
    throw err;
  }
}
```

✓ Ví dụ gọi hàm

js

Copy Edit

```
createCoupon("SALE30", 30)
  .then(result => {
    console.log("Thành công:", result);
  })
  .catch(err => {
    console.log("Thất bại:", err.message);
  });
```

🔧 Lưu ý:

- Hàm này không phụ thuộc vào Express – bạn có thể dùng trực tiếp trong backend, seeder, hoặc test script.
- Nếu bạn đang dùng trong route Express, nên đặt phần logic này vào controller và gọi từ router.

Bạn muốn mình viết thêm hàm kiểm tra trùng mã trước khi tạo hoặc chuyển code về viết hoa tự động không? Mình có thể thêm logic vào.

📄 🍌 🗨️ 🔊 ✎️ ⬆️ ↺ ✓