# Generate Test Plan using Al (Gemini)

Below are my requirements. I have 3 testers. I have 45 days. Testing should be performed in Mac and windows. Browsers are chrome and safari:

Project high level Requirements: Project Name: "Style Time" – A Fashion E-commerce Platform

 Project Description: Style Time is an online fashion marketplace designed to connect independent fashion designers with customers who crave unique and trendy clothing. The platform will provide a user-friendly shopping experience with personalized recommendations, secure transactions, and seamless customer support.

Target Audience: Fashion-conscious individuals aged 18-35 who value individuality and seek out unique clothing pieces.

#### Key Features and Requirements:

- User Accounts:
- Registration/Login: Secure user registration and login with email or social media accounts.
- Profile Management: Allow users to update personal information, shipping addresses, and payment details.
- Order History: View past orders and track current order status.
- Product Catalog:
- Browse/Search: Intuitive browsing by category, brand, style, or search by keywords.
- Product Pages: Detailed product descriptions, high-resolution images, size charts, and customer reviews.
- ..... (paste all the high level requirement from <u>README.MD</u> file )

#### Gemini

2.5 Flash ▼

#### Hello, Anum

Below are my requirements. I have 3 testers. I have 45 days. Testing should be performed in Mac and windows. Browsers are chrome and safari: Project high level Requirements: Project Name: "Style Time" – A Fashion E-commerce Platform Project Description: Style Time is an online fashion marketplace designed to connect independent fashion designers with customers who crave unique and trendy clothing. The platform will provide a user-friendly shopping experience with personalized recommendations, secure transactions, and seamless customer support. Target Audience: Fashion-conscious individuals aged 18-35 who value

Q Deep Research



### Outcome - User can read the full test plan created by Gemini

#### Phase 4: UAT Support, Final Regression & Sign-off (Days 41-45)

- User Acceptance Testing (UAT) Support: Assist UAT users, clarify doubts, and log any issues they find.
- Final Regression: Perform a final, targeted regression on critical paths to ensure no new defects have been introduced.
- Test Closure: Document test results, prepare a summary report of tested features, open defects, and overall quality.
- Knowledge Transfer: Ensure all findings are well-documented for future reference.

#### 5. Roles & Responsibilities (3 Testers)

While roles can be flexible, here's a suggested distribution:

- Tester 1 (Lead/Senior):
  - · Overall test strategy and planning.
  - Mentoring junior testers.
  - · Handling complex test scenarios.
  - Primary responsibility for User Accounts & Product Catalog.
  - Mac testing focus.
  - · Defect triaging and communication with development.
- Tester 2
  - Primary responsibility for Shopping Cart & Checkout.
  - Windows testing focus.
  - Assisting with cross-browser testing.
  - · Detailed test case execution and defect reporting.
- Tester 3:
- · Primary responsibility for Seller Dashboard & Customer Support.
- Cross-OS (Mac/Windows) and cross-browser testing as needed.
- · Assisting with test data preparation and exploratory testing.
- Detailed test case execution and defect reporting.

Project Name: Style Time - A Fashion E-commerce Platform

**Duration: 45 Days** 

Testers: 3

Testing Environments: Mac & Windows

Browsers: Chrome & Safari

#### 1. Test Strategy & Approach

Given the time and resources, a **risk-based**, **agile testing approach** will be most effective. We will prioritize testing of core functionalities and high-risk areas first.

- Iterative Testing: Break down features into smaller, testable chunks.
- Parallel Testing: Utilize the 3 testers to test different modules or features concurrently.
- Cross-Browser/OS Testing: Emphasize testing critical paths on both Mac (Safari/Chrome) and Windows (Chrome).
- Exploratory Testing: Allow testers time for unscripted exploration to uncover unexpected issues.
- Defect Management: Implement a clear process for logging, tracking, and retesting defects.

#### 2. Scope of Testing

#### In-Scope (Mandatory for 45 Days):

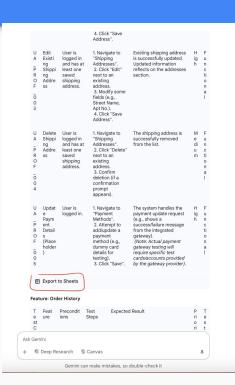
- User Accounts:
  - Registration/Login (Email, Social Media if integrated in this phase)
  - Profile Management (Update personal info, shipping, payment)
  - · Order History (Viewing past orders, tracking current status)
- Product Catalog:
  - Browse/Search (Category, Brand, Style, Keywords)
  - Product Pages (Descriptions, Images, Size Charts, Customer Reviews)
  - Recommendations (Basic functionality)
- Shopping Cart:
  - Add/Remove Items
  - Quantity Adjustments

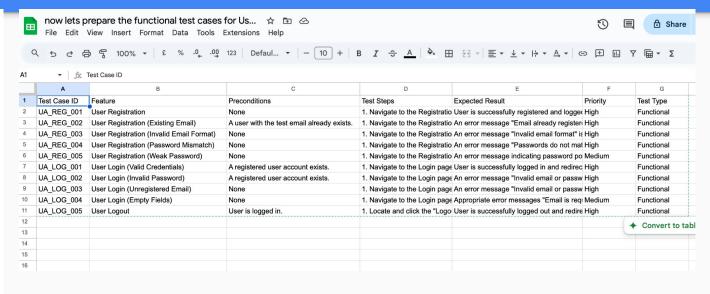
#### Ask Gemini

+ @ Deep Research @ Canvas

# Generate Test Cases for User Account And Product Catalog Session of the requirement

• now let's prepare the functional test cases for User Accounts and product Catalog





## Generate Test Data

can we have test data as well as for each case combination. add this test data to spreadsheet as new column for each test case

can we have test data as well as for each case combination, add this test data to spread sheet as new column for each test case Yes, absolutely! Incorporating test data directly into your test cases is a best practice. It makes your tests much clearer, repeatable, and easier for anyone to execute. Instead of a separate spreadsheet column for each specific data point (which can get unwieldy), I'll add a single "Test Data" column to the test case table. This column will contain the specific values or conditions needed for that test case. Here's how we'll integrate the test data into the functional test cases for User Accounts and Product Catalog: Functional Test Cases: User Accounts (with Test Data) Module: User Accounts Feature: Registration/Login T Fea Pre Test Steps e ture con st ditio C ns Expected Result Email: U Use Non 1. User is successfully registered and logged ig u testuser@ex in. A confirmation R stra Registrati <br >br > Password: message or redirect SecureP@ss1 on page. 2. Enter 23 <br> dashboard/homepage valid Confirm Password: Password, SecureP@ss1 Confirm Password. 3. Click Ask Gemini

Gemini can make mistakes, so double-check it

+ @ Deep Research @ Canvas

can we have test data as well as for each case co 🌣 🗈 🗠  File Edit View Insert Format Data Tools Extensions Help								
C	\ 5	号 100%	¥ £ % .	000 123	Defaul ▼   - 10 +   B I ÷	A   >. H	53 -   ≣ -	<u>↓</u> •  ÷ • A
E1 ▼   $\hat{f}_{\mathbb{K}}$ Test Data								
	А	В	С	D	E	F	G	Н
1	Test Case ID	Feature	Preconditions	Test Steps	Test Data	Expected Result	Priority	Test Type
2	UA_REG_001	User Registration	None	1. Navigate to th	Email: testuser@example.com Password: S	User is successf	High	Functional
3	UA_REG_002	User Registration	A user with the to	1. Navigate to th	Email: existing@example.com Password: Ne	An error messag	High	Functional
4	UA_REG_003	User Registration	None	1. Navigate to th	Email: invalid@.com Password: SecureP@s	An error messag	High	Functional
5	UA_REG_004	User Registration	None	1. Navigate to th	Email: nomatch@example.com Password: F	An error messag	High	Functional
6	UA_REG_005	User Registration	None	1. Navigate to th	Email: weakpass@example.com Password:	An error messag	Medium	Functional
7	UA_LOG_001	User Login (Valid	A registered use	1. Navigate to th	Email: testuser@example.com Password: Se	User is successf	High	Functional
8	UA_LOG_002	User Login (Inva	A registered use	1. Navigate to th	Email: testuser@example.com Password: W	An error messag	High	Functional
9	UA_LOG_003	User Login (Unre	None	1. Navigate to th	Email: unregistered@example.com Passwor	An error messag	High	Functional
10	UA_LOG_004	User Login (Emp	None	1. Navigate to th	Email: (leave empty) Password: (leave empt	Appropriate erro	Medium	Functional
11	UA_LOG_005	User Logout	User is logged in	1. Locate and cli	(Logged-in state of testuser@example.com)	User is successf	High	Functional
12								
13								

• instead of multiple test cases, can you prepare one solid functional test flow as end for user account and product catalog together

Style Time E-commerce Platform Testing ~

 Can you prepare one solid functional test flow for user account and product catalog together; below the high level requirement; Project Name; "Style Time" -A Fashion E-commerce Platform Project Description: Style Time is an online fashion marketplace designed to connect independent fashion designers with customers who crave unique and trendy clothing. The platform will provide a userfriendly shopping experience with personalized recommendations, secure transactions, and seamless customer support. Target Audience: Fashionconscious individuals aged 18-35 who value individuality and seek out unique clothing pieces. Key Features and Requirements: User Accounts: Registration/Login: Secure user registration and login with email or social media accounts, Profile Management: Allow users to update personal information. shipping addresses, and payment details. Order History: View past orders and track current order status. Product Catalog: Browse/Search: Intuitive browsing by category, brand, style, or search by keywords. Product Pages: Detailed product descriptions, high-resolution images, size charts, and customer reviews. Recommendations: Personalized product suggestions based on browsing history and preferences.

I'll create a comprehensive functional test flow that integrates user account management with product catalog functionality for the Style Time platform.

Style Time - Integrated User Account & Product Catalog Test Flow



I've created a comprehensive functional test flow that integrates user account management with product catalog functionality for your Style Time platform. This test flow covers the complete user journey from registration through product discovery and purchase, ensuring both features work seamlessly together.

The test flow is structured in 5 phases:

- 1. User Registration and Profile Setup Validates account creation and personal preference configuration
- 2. Product Catalog Exploration Tests browsing, searching, and product detail functionality
- 3. Personalized Experience Validation Ensures recommendations work based on user preferences and behavior
- 4. Account Integration with Shopping Verifies how account information enhances the shopping experience
- 5. Cross-Feature Validation Tests how changes in one area affect the other

Key highlights of this test flow:

. End-to-End Integration: Tests how user account data influences product recommendations and shopping experience

Reply to Claude..



#### Style Time - Integrated User Account & Product Catalog Test Flow

#### Test Flow: Complete User Journey - Registration to Purchase

#### Test Scenario Overview

Test Name: End-to-End User Account Creation and Product Discovery Flow Objective: Validate the complete user journey from account registration through product browsing, personalized recommendations, and order placement Test Duration: 45-60 minutes

Prerequisites: Clean test environment, test payment gateway configured, sample product catalog loaded

#### Phase 1: User Registration and Profile Setup

#### Step 1: User Registration

Action: Navigate to Style Time homepage and initiate registration

- Click "Sign Up" button
- · Choose registration method (Email vs Social Media)
- · Test Data:
- · Email: testuser\_fashion@example.com
- Password: StyleTime2024!
- Name-Sarah Johnson
- Age: 28

#### Expected Results:

- · Registration form displays correctly
- · Email verification sent successfully
- · User redirected to profile completion page

#### Validation Points:

- · Email format validation works
- · Password strength requirements enforced
- · Age verification for target audience (18-35)

#### Step 2: Profile Management Setup

Action: Complete user profile with personal preferences

- · Add shipping address: 123 Fashion Ave, New York, NY 10001
- · Add payment method: Test credit card (41111111111111)
- · Set style preferences: Bohemian, Minimalist, Vintage · Set size preferences: Medium, Size 8, US sizing
- Expected Results:

# Generate Test Strategy

• with the test cases what you created above, give me which test go in which layer of pyramid?

#### 1. Unit Tests (Bottom Layer - Most numerous, fastest, developer-centric)

- These tests are not directly represented by the functional test cases I provided above, as those were higher-level.
- . What would go here: Individual functions or methods within your codebase.
- User Accounts:
  - UA\_REG\_001 (and others for registration): The validation logic for email format, password strength, and uniqueness before hitting the database.
     The password hashing function.
  - UA\_LOG\_001 (and others for login): The authentication logic that compares a provided password hash to a stored hash.
  - UA\_PROF\_001: The data validation logic for updating personal info (e.g., checking for valid name characters).
- Product Catalog:
- PC\_BROW\_003 (and others for search/filter/sort): The logic for filtering products by category, brand, style, or sorting by price. The function that parses search queries.
- PC\_PROD\_001 : The data serialization logic that takes raw product data
  and formate it for display.
- PC\_REC\_001: The recommendation algorithm logic itself, ensuring it correctly suggests products based on given inputs.

#### 2. Integration Tests (Middle Layer - Fewer than Unit, faster than UI, often API-

- These tests verify that different system components work together. For an ecommerce platform, these are often API tests (testing the backend endpoints directly without the UI).
- . Test Case IDs that would fit here (as API tests):
  - User Accounts:
  - UA\_REG\_001, UA\_REG\_002, UA\_REG\_003, UA\_REG\_004,
    UA\_REG\_005: Testing the Registration API endpoint directly with
    various valid/invalid inputs to ensure it interacts correctly with the user
    service and database.
  - UA\_LOG\_001, UA\_LOG\_002, UA\_LOG\_003, UA\_LOG\_004: Testing the Login API endpoint for correct authentication and session management.

PC\_PROD\_005, PC\_PROD\_006: Testing the Single Product Details

API endpoint (e.g., /opi/products/{id}) to ensure all relevant data
(description, images, size charts, reviews, options) is returned.

 PC\_REC\_001, PC\_REC\_002: Testing the Recommendation API endpoint to ensure it returns relevant product suggestions.

#### 3. UI / End-to-End Tests (Top Layer - Fewest, slowest, browser-driven)

- These tests interact with the application through its graphical user interface. Your detailed "functional test flow" fits perfectly here.
- · Test Case IDs that would fit here:
- ST\_EZE\_001\_(Your combined End-to-End Functional Test Flow): This single flow encompasses the entire user journey through the Ul for ore accounts and product catalog, making it a perfect candidate for the top of the pyramid. It werlifes that all integrated components work together from the user's perspective.
- Any specific UI-centric checks or cross-browser/OS verification of the visual layout and interactivity.
- . Specific UI interactions NOT covered by E2E flow:
- · Verifying that clicking a product image opens a larger view.
- . Testing interactive elements like "Add to Wishlist" (if implemented).
- Testing responsiveness of the UI on various screen sizes (though part of non-functional, it's UI-driven).

#### In essence, for your provided test cases:

- Most of the individual, granular UA\_XXX and PC\_XXX functional test cases
  would best be automated as Integration (API) Tests. This gives you fast
  feedback on the backend logic and data flow.
- The comprehensive ST\_E2E\_001 End-to-End Functional Test Flow is your primary UI Test, verifying the full user experience. You'd run this flow for each required browser/OS combination.

This approach aligns well with Shift-Left: developers are constantly running unit and integration tests, catching bugs early. Your QA team focuses on the critical, user-facing E2E flow for overall system validation and cross-browser checks.

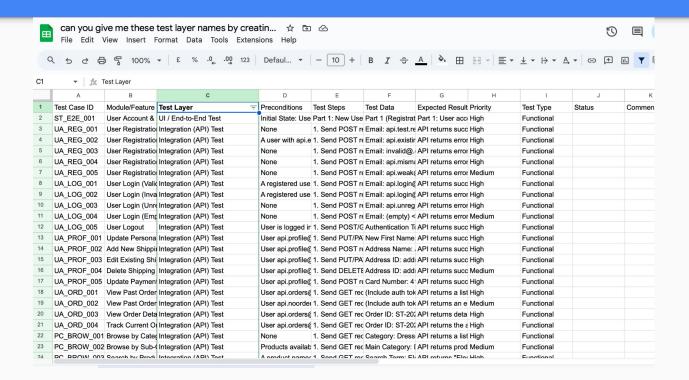
₽ ₽ C < € :</p>

Ask Gemini

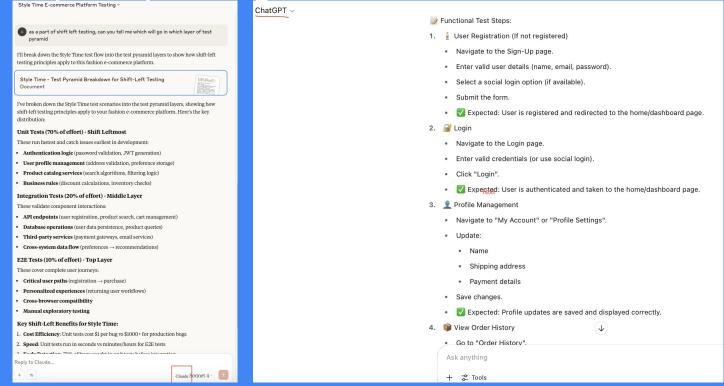
+ @ Deep Research @ Canvas

Gemini can make mistakes, so double-check it

 can you give me these test layer names by creating another column for each row so that I can see all the details in one spreadsheet



User can use ChatGPT, Claude too to use the same prompts to get the better results



## Thanks!

