February 20, 2025

Prepared by: testify team

REquirements gathering

for www.Shopify.com

# Overview

This document for Shopify.com outlines the essential features and functionalities required to ensure the platform operates effectively and meets the needs of its users, including merchants, customers, and administrators. This document serves as a comprehensive guide for developers, testers, and stakeholders, detailing the core and additional functionalities that Shopify.com must support. It covers areas such as user registration and authentication, store setup, product and order management, payment processing, shipping and fulfillment, theme customization, app integrations, reporting, and admin dashboard operations. By defining these requirements, the FRD ensures that Shopify.com delivers seamless, secure, and scalable e-commerce experience while maintaining high performance, usability, and reliability across all devices and browsers.

# Shopify Frontend

This document is intended to be used as an introduction to the Shopify default homepage and your first store made with Shopify. The website homepage is an overview of the store and its products or services. This structure is designed to engage visitors, showcase products, and encourage conversions, while also providing essential information about the brand and its offerings.

Shopify homepage frontend.


Stakeholder Analysis–– Identifying key stakeholders and their needs.

**Purpose:** Identify key stakeholders and understand their needs, expectations, and influence on the project.

## Steps for Stakeholder Analysis:

### Identify Stakeholders:

* + List all individuals or groups who have an interest in the project. Common stakeholders include:
    - **End Users:** Individuals who will use the software.
    - **Project Managers:** Responsible for project oversight and delivery.
    - **Business Analysts:** Gather and analyze requirements.
    - **Developers:** Build the software.
    - **Quality Assurance (QA) Team:** Responsible for testing the software.
    - **Clients/Customers:** External parties who have commissioned the project.

### Assess Stakeholder Needs:

* + For each stakeholder, identify their specific needs and expectations. This can include:
    - **End Users:** Usability, accessibility, and functionality.
    - **Project Managers:** Timelines, budget constraints, and project scope.
    - **Developers:** Clear requirements and technical specifications.
    - **QA Team:** Testable requirements and acceptance criteria.
    - **Clients/Customers:** Business value, return on investment, and quality assurance.

### Determine Influence and Interest:

* + Assess the level of influence and interest each stakeholder has in the project. This can help prioritize communication and engagement strategies.

## Stakeholder Analysis for Shopify Platform

* **shopkeepers**:
  + **Needs**: User-friendly interface for managing products, sales analytics, and customer engagement tools.
  + **Goals**: Increase sales, improve customer satisfaction, and streamline operations.
* **Customers**:
  + **Needs**: Easy navigation, secure checkout process, and access to product information.
  + **Goals**: Quick and efficient shopping experience, trust in the platform, and satisfaction with purchases.
* **Developers**:
  + **Needs**: Comprehensive API documentation, support for app integration, and tools for testing.
  + **Goals**: Build scalable and efficient applications that enhance the Shopify ecosystem.
* **Shopify Support Team**:
  + **Needs**: Access to customer queries, troubleshooting tools, and knowledge base.
  + **Goals**: Provide timely and effective support to enhance user experience.
* **Investors**:
  + **Needs**: Insights into platform performance, user growth metrics, and market trends.
  + **Goals**: Ensure profitability and long-term growth of the Shopify platform.

# User Stories & Use Cases

**Purpose:** Illustrate how users interact with the system to capture functional requirements.

## User Stories:

User stories are short, simple descriptions of a feature from the perspective of the end user. They typically follow the format:

*As a [type of user], I want [some goal] so that [some reason].*

**User Stories for Shopify Platform**

* As a trader, I want to easily manage my inventory so that I can keep track of stock levels and avoid overselling.
* As a customer, I want to filter products by category and price so that I can find what I need quickly.
* As a developer, I want to access the Shopify API documentation so that I can integrate my app seamlessly with the platform.
* As a customer, I want to receive notifications about my order status so that I can stay informed about my purchase.
* As a merchant, I want to analyze sales data through dashboards so that I can make informed business decisions.

## Use Cases:

Use cases provide a more detailed description of how users will interact with the system, including the steps involved.

**Use Cases for Shopify Platform**

1. **Checkout Process**:
   * **Actors**: Customer, Payment Gateway, Shopify System
   * **Preconditions**: Customer has items in the cart.
   * **Steps**:
     1. Customer clicks on the checkout button.
     2. Customer enters shipping information.
     3. Customer selects payment method.
     4. Payment gateway processes the payment.
     5. Shopify confirms the order and sends a confirmation email.
   * **Postconditions**: Order is successfully placed, and inventory is updated.
2. **Product Management**:
   * **Actors**: Merchant, Shopify System
   * **Preconditions**: Merchant is logged into their account.
   * **Steps**:
     1. Merchant navigates to the product management section.
     2. Merchant adds a new product with details (name, price, description).
     3. Merchant uploads product images.
     4. Merchants save the product.
   * **Postconditions**: New product is listed in the store and available for purchase.
3. **Customer Account Creation**:
   * **Actors**: Customer, Shopify System
   * **Preconditions**: Customer is on the registration page.
   * **Steps**:
     1. Customer fills in personal information (name, email, password).
     2. Customer submits the registration form.
     3. Shopify sends a verification email.
     4. Customer verifies their email.
   * **Postconditions**: Customer account is created and ready for use.

**Main Flow:**

1. The customer enters a product name in the search bar.
2. The customer clicks the "Search" button.
3. The system retrieves and displays a list of products matching the search criteria.
4. The customer selects a product to view its details.

**Alternate Flow:**

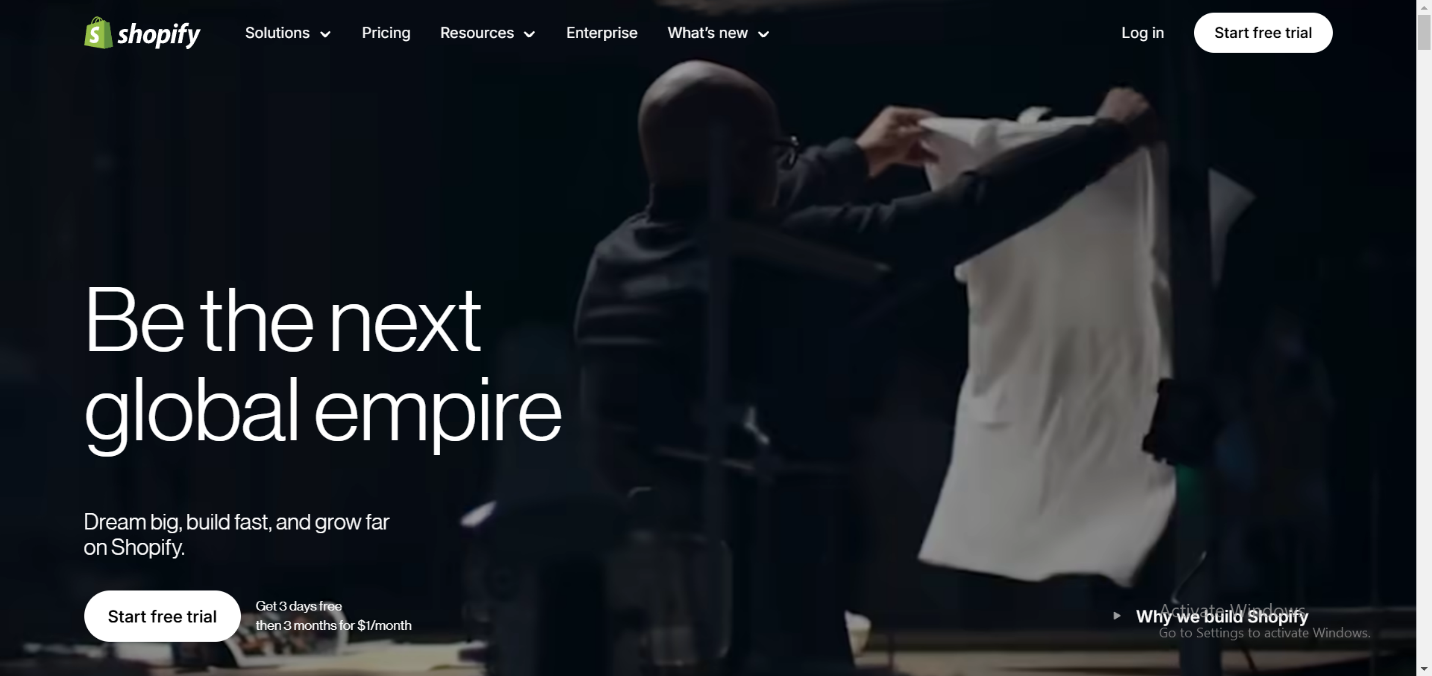
* If no products match the search criteria, the system displays a message indicating that no results were found.

# Functional Tests

## Navigating the homepage

The Shopify homepage is designed to introduce the platform to potential users, explain its features, and encourage visitors to sign up for a trial or learn more about starting an online store. Here's what you'll typically see:

- Header section: The header will be displayed at the top of the page, on every page of the store; not just the home page.



The header has the following navigation options:

* Store logo: redirects the customer to the homepage of the website.
* Navigation menu: consists of

1) Solutions: lists eCommerce solutions for different business types.

2) Pricing: redirects to Shopify’s pricing plans.

3) Resources: basically guides, blogs, and help center.

4) Enterprise: redirects to Shopify for enterprises.

5) What’s new: has links to all previous versions and the changelogs.

* Log In button: provides existing users with quick access to dashboard.
* Start free trial button: The primary call-to-action (CTA) for new users to sign up for Shopify. Clicking it takes users to the signup page.

- Hero section: This is the large, attention-grabbing area below the header. It typically features a bold changing headline (e.g., “Be the next global empire”), a subheadline highlighting benefits (like ease of use or scalability), and a primary CTA button (e.g., “Start free trial”). Often, it’s paired with a dynamic visual like a video, or high-quality image of a store interface to showcase Shopify’s platform in action.

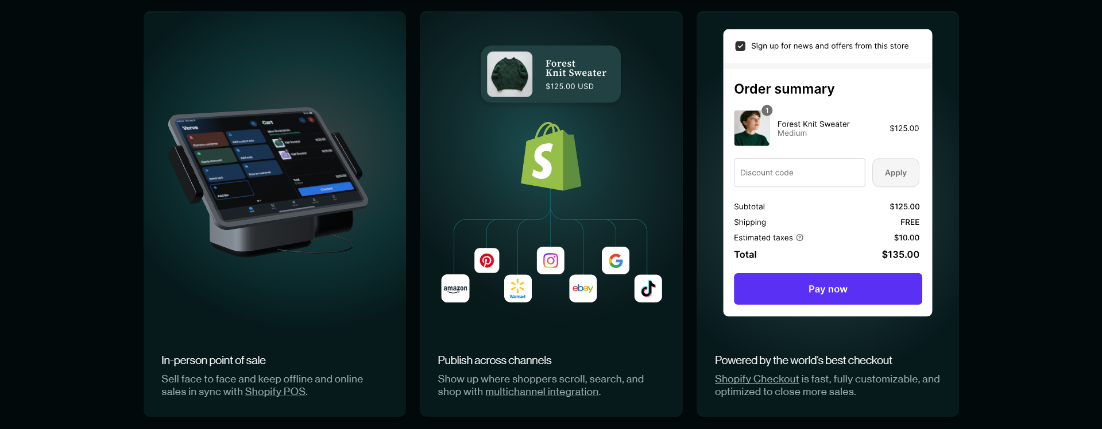
- Key benefits section: Highlights three core pillars (e.g., “Sell online and in person”, “Sell locally and globally”, “Sell on desktop and mobile”). Each is a block with an icon (e.g., clothing, skincare products), a short title, and a one-sentence description. Often links to deeper pages like “Online Store” or “Point of Sale”.

A screenshot of a mobile device

AI-generated content may be incorrect.

- Social proof section: Features a row of brands using Shopify (e.g., Gymshark, Summer Solace Tallow) and a stat like “...with $500M+ sales annually.” Followed by a button, pick a plan that fits, redirects you to the pricing page.

- Feature showcase section: A grid or tabbed section highlighting tools like “in person point of sale”, “Publish across channels”, “Powered by the world’s best checkout”. Each has a thumbnail (e.g., order summary), a title, and a brief pitch (e.g., “Shopify Checkout is fast, fully customizable, and optimized to close more sales.”). Links lead to dedicated feature pages. The page goes further into more features like desktop and mobile compatibility, shopify app store, and even frontend code customization if you are an experienced developer.



- Footer section: dark background with white text, split into columns: “Shopify” (About, Careers), “Products” (Online Store, POS), “Support” (Help, Contact), and “Global impacts” (Terms, Privacy). Includes social icons (Twitter, Instagram, etc.) and a language selector (e.g., English, Français).

|  |  |
| --- | --- |
| Functional test no. | Description |
| FR-1 | The system should provide a structured and user-friendly navigation menu.   * Users should access sections such as pricing, features, and resources from the homepage. * The system should ensure consistent navigation across mobile and desktop. |
| FR-2 | The system should provide a responsive design for desktops, tablets, and mobile devices.   * Users should have neat and user-friendly experience on different platforms. |
| FR-3 | The system should provide a dashboard for store management.   * Users should access store settings, product management, and analytics from a centralized dashboard. * The dashboard should display key performance metrics in a user-friendly format. |

## User registration

Users should be able to create accounts and access the Shopify platform with ease. A secure and seamless registration and authentication process ensures a smooth user experience while maintaining data security and preventing unauthorized access. The registration process is essential before the user gets access to Shopify features and services.

A screenshot of a login screen

AI-generated content may be incorrect.

|  |  |
| --- | --- |
| Functional test no. | Description |
| FR-4 | The system should allow users to sign up using an email address, Google, or Apple ID.   * Users must provide a valid email address. * The system should check for duplicate email addresses to prevent multiple accounts with the same credentials. |
| FR-5 | The system should send a verification email upon registration.   * The system should resend the verification email if the user requests it. * The verification link should be valid for 24 hours. |
| FR-6 | The system should allow users to log in securely using email/password, Google, or Apple ID.   * Users should have the option to enable two-factor authentication (2FA) for enhanced security. * If login attempts exceed three failures, the system should temporarily lock the account for security purposes. * Users should be able to view and manage their active login sessions. |
| FR-7 | The system should provide a password reset option via email.   * Users should be required to answer a security question or complete a CAPTCHA verification before resetting their password. * The reset link should expire within 30 minutes of being sent. * The system should prevent the reuse of previous passwords for security purposes. |

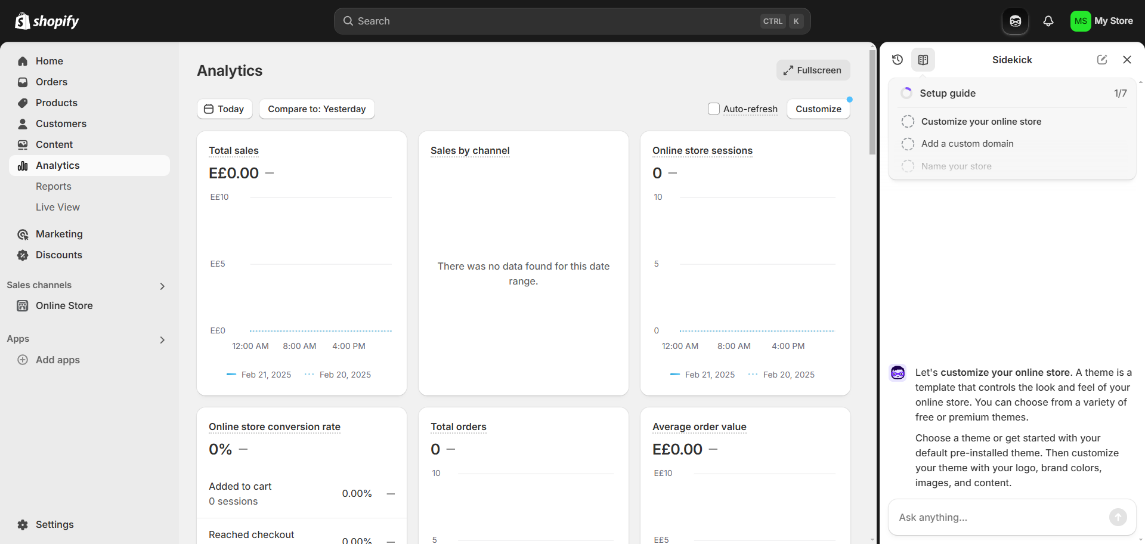
## Store Setup and Management

Main functionality of the website. It is the functionality required so that users be able to create and manage their online stores. It includes setting up store names, customizing design templates, and managing product listings to ensure an efficient and user-friendly experience.

A screenshot of a computer

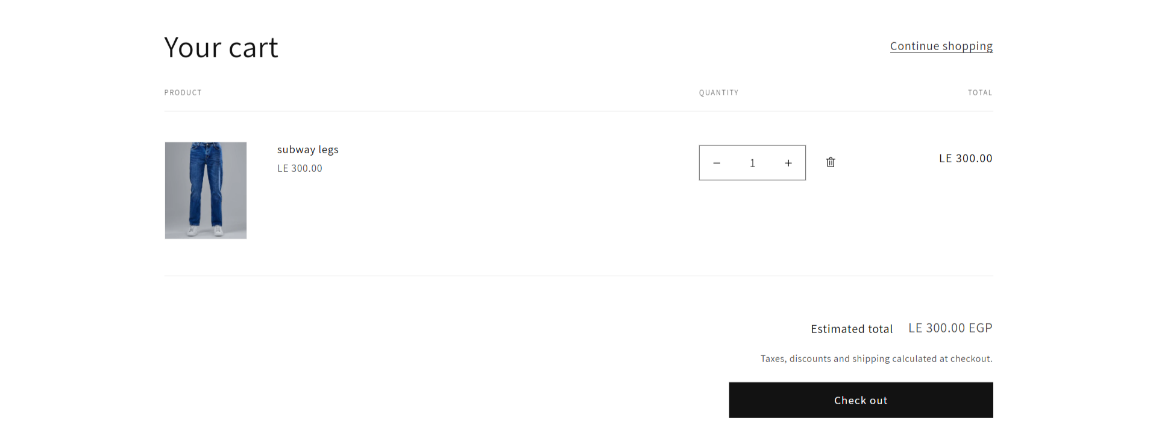
AI-generated content may be incorrect.

|  |  |
| --- | --- |
| Functional test no. | Description |
| FR-8 | The system must allow users to set up new store with unique name.   * Users should enter a store name, which the system should validate for uniqueness. * The system should provide real-time feedback on name availability. |
| FR-9 | The system should allow users to customize their store appearance.   * Users should be able to select and modify store themes. * Users should upload and configure store logos, banners, and colors. * Users should be able to preview changes before applying. |
| FR-10 | The system should allow users to add, update, delete products.   * Users should enter product details including title, description, images, and pricing. * Users should be able to set product category and stock availability. * Users should have the ability to bulk upload products via CSV files. * The system should support product variants such as size and color. |
| FR-11 | The system should allow users to configure payment options.   * Users should connect their store to payment gateways like PayPal, Stripe, or Shopify Payments. * The system should allow users to define tax settings based on location. * Users should be able to enable or disable specific payment methods. |
| FR-12 | The system should allow users to manage shipping options.   * Users should define shipping zones and rates based on customer location. * The system should integrate with major shipping carriers to calculate real-time shipping costs. * Users should offer free shipping, flat-rate shipping, or weight-based shipping. |
| FR-13 | The system should allow users to manage discount codes.   * Users should define discount codes with percentage-based or fixed-amount discounts. * Users should set expiration dates and usage limits for discount codes. * The system should allow automatic discounts for specific products or customer groups. |
| FR-14 | The system should provide inventory management features.   * Users should track stock levels and receive shortage alerts. * The system should automatically update inventory after order fulfillment. * Users should enable or disable backorders for out-of-stock products. |
| FR-15 | The system should allow users to manage orders efficiently.   * Users should view and filter orders based on status (pending, shipped, delivered, canceled, etc.). * The system should provide automated order status updates to customers. * Users should issue refunds, and handle returns directly from the dashboard. |
| FR-16 | The system should allow users to integrate third-party apps.   * Users should browse and install apps from Shopify App Store. * The system should allow API-based integrations for custom functionalities. * Store owners should manage installed apps from their dashboard. |



## Checkout and payment processing

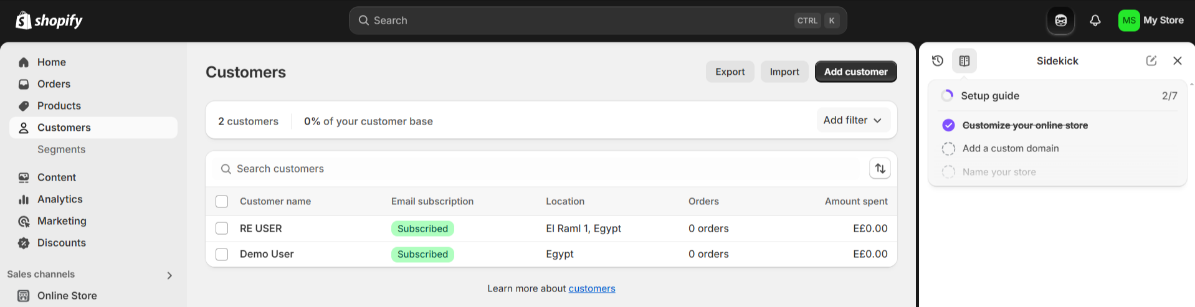
One of the most important functionalities of any e-Commerce aiding service is the payment processing, the process of making purchases on Shopify-powered stores. A seamless and secure checkout system ensures customer trust and reduces cart abandonment by providing multiple payment options and order confirmation notifications.



|  |  |
| --- | --- |
| Functional test no. | Description |
| FR-17 | The system should allow customers to complete purchases using a secure checkout process.   * Customers should enter shipping and billing details before proceeding with payment. * The system should validate the shipping address and calculate applicable taxes. |
| FR-18 | The system should support multiple payment options.   * Customers should be able to pay using credit/debit cards, PayPal, Apple Pay, or other supported methods. * The system should process transactions securely and provide confirmation |
| FR-19 | The system should generate order confirmation emails.   * Customers should receive a confirmation email with order details immediately after a successful purchase. * Store owners should be notified of new orders in their dashboard. |
| FR-20 | The system should allow customers to apply discount codes   * Customers should enter a valid discount code before completing their purchase. * The system should verify the discount eligibility and apply the discount automatically. * The system should display the updated total after applying the discount. |

## Customer Management

This section outlines the tools that should be available for store owners to manage their customers effectively. Features such as importing / exporting customers, customer personal details, order history tracking, customer segmentation, tax collection status, and promotional email capabilities enable businesses to enhance customer engagement and retention.



|  |  |
| --- | --- |
| Functional test no. | Description |
| FR-21 | The system should allow store owners to manage customer profiles.   * Store owners should view a list of registered customers and their order history. * The system should provide filtering options based on purchase behavior. |
| FR-22 | The system should allow store owners to send promotional emails.   * Store owners should draft, schedule, and send promotional emails to customers. * The system should allow segmentation of customer lists based on preferences and past purchases. |
| FR-23 | The system should enable customer segmentation based on demographics and purchase history.   * Store owners should create targeted marketing campaigns for different customer segments. * The system should generate analytics on customer behavior for improved decision-making. |
| FR-24 | The system should allow store owners to import and export customer data.   * Store owners should be able to bulk add or remove customers through CSV imports. * The system should support exporting customer data for external use. |

# Non-functional Tests

Non-functional testing plays a crucial role in ensuring Shopify.com meets industry standards, especially for performance, security, usability, and reliability. Unlike functional testing, which verifies specific features and workflows, non-functional testing assesses how well the platform operates under different conditions and constraints.

Given that Shopify serves millions of businesses worldwide, the system must deliver consistent performance across various traffic levels, maintain robust security against cyber threats, and provide a seamless user experience. High-traffic sales events, such as Black Friday, require the system to scale effectively while maintaining fast response times and uninterrupted service. Additionally, compliance with global data protection regulations is necessary to protect user privacy and maintain trust.

This section focuses on the non-functional requirements that define Shopify’s operational standards, ensuring the platform remains efficient, secure, and reliable for all users.

## 1. Performance testing

- Consists of: load testing, stress testing, endurance testing, spike testing, volume testing.

* Load testing: evaluating the system’s performance under the expected workload
* Stress testing: evaluating the system's performance under a load much higher than expected.
* Endurance (Soak) testing: to determine if the system can sustain the continuous expected load for a long duration. Issues like memory leakage can be found with endurance testing.
* Spike testing: checking the system’s behavior on sudden increase of current users, above and below expected workload.
* Volume testing: performed by feeding the application with a high volume of data. Identifies the bottleneck in the application with a high volume of data.

|  |  |
| --- | --- |
| Non-Func. test no. | Description |
| NFR-01 | Load Testing: How the system performs under expected workloads.   * Simulate concurrent users browsing, adding items to carts, and completing purchases. * Measure page load times and response times under normal conditions. |
| NFR-02 | Stress Testing: Evaluates system performance under extreme conditions to determine the breaking point.   * Gradually increase traffic beyond normal loads to identify when performance starts degrading. * Check system recovery post high-stress scenarios. |
| NFR-03 | Endurance (Soak) Testing: Assesses system stability and memory management over extended periods.   * Run continuous usage tests over days to detect memory leaks and performance degradation. * Monitor database efficiency under prolonged high loads. |
| NFR-04 | Spike Testing: Examines system response to sudden increases and decreases in user load.   * Simulate flash sales with a sudden surge in transactions. * Assess system performance during rapid traffic increases and decreases. |
| NFR-05 | Volume Testing: Tests system performance by inserting large amounts of data into databases and assessing response times.   * Populate the system with high volumes of products, orders, and users. * Identify performance bottlenecks when handling bulk operations. |

## 2. Security testing

- Security Testing checks if the software is secure and protects sensitive information. Aims to find and identify vulnerabilities that can be exploited by hackers.

- Mainly focuses on: User authentication (is this a valid user account?), user authorization (what authorities does this user have?), data encryption & decryption, network security, client/server-side application security.

|  |  |
| --- | --- |
| Non-Func. test no. | Description |
| NFR-06 | Authentication Testing: Ensures only authorized users can access restricted areas.   * Test login mechanisms, multi-factor authentication, and session management. * Validate that incorrect login attempts trigger appropriate security measures. |
| NFR-07 | Authorization Testing: Validates user permissions and access levels.   * Ensure admin users have access to restricted settings while regular users do not. * Verify that users cannot access other merchants’ data. |
| NFR-08 | Data Encryption Testing: Confirms that sensitive information is encrypted during storage and transmission.   * Check encryption standards for stored payment and customer data. * Validate secure data transmission using SSL/TLS protocols. |
| NFR-09 | Network Security Testing: Assesses protection mechanisms against external attacks like DDoS and SQL injection.   * Simulate cyber-attacks to identify vulnerabilities. * Validate firewall and security patch effectiveness. |
| NFR-10 | Application Security Testing: Examines vulnerabilities in both client-side and server-side applications.   * Perform penetration testing to uncover security loopholes. * Ensure input validation prevents malicious script injections. |

## 3. Recovery testing

- Assesses how well a system can recover after a failure or crash. It checks if the software can resume normal operation without losing data.

- The software should be tested once failure occurs whether it is a power supply failure, external server being unreachable, wireless signal loss, physical conditions, database servers go down, API response failure.

|  |  |
| --- | --- |
| Non-Func. test no. | Description |
| NFR-11 | Power Failure Recovery: Tests system stability after sudden power outages.   * Verify auto-restart procedures and data integrity after power loss. |
| NFR-12 | Downtime Recovery: Assesses how the system responds when external servers become unreachable.   * Test failover mechanisms to ensure minimal downtime. |
| NFR-13 | Network Disruptions: Evaluates Shopify’s ability to recover from Wi-Fi or mobile network failures.   * Check session persistence after brief network disruptions. |
| NFR-14 | Database Crash Recovery: Tests whether the system can restore lost or corrupted data without affecting ongoing operations.   * Perform backup and restore procedures. * Validate data consistency post-recovery. |
| NFR-15 | API Failure Recovery: Examines system behavior when third-party service APIs become unresponsive.   * Simulate third-party payment gateway failures. * Ensure the system retries or provides user-friendly fallback messages. |

## 4. Compatibility testing

- Ensures the software is compatible with different environments.

* Operating system compatibility: check the application on different OS (windows, linux, IOS, android, macOS).
* Browser compatibility: run the application on different browsers (Chrome, Safari, Brave, Opera, Firefox).
* Hardware compatibility: test the application on devices with varying ram sizes and disk space.
* Backward compatibility.
* Forward compatibility.

|  |  |
| --- | --- |
| Non-Func. test no. | Description |
| NFR-16 | Operating System Compatibility: Validates Shopify’s functionality on Windows, Linux, macOS, Android, and iOS.   * Perform UI and performance tests across platforms. |
| NFR-17 | Browser Compatibility: Tests performance on Chrome, Safari, Firefox, Edge, Opera, and Brave.   * Ensure feature consistency across browsers. |
| NFR-18 | Hardware Compatibility: Evaluates system behavior on devices with varying RAM and storage capacities.   * Test app performance on low-end and high-end devices. |
| NFR-19 | Backward Compatibility: Ensures updates do not break functionality on older versions.   * Check previous features remain functional post-update. |
| NFR-20 | Forward Compatibility: Verifies that newer versions will be supported on same previously used devices.   * Test upcoming OS/browser beta versions for compatibility. |

## 5. Configuration testing

checks if the software works correctly with different configurations or settings. It ensures that the application adapts well to various setups.

|  |  |
| --- | --- |
| Non-Func. test no. | Description |
| NFR-21 | Testing with Various Database Configurations to ensure smooth integration with different storage solutions.   * Validate MySQL, PostgreSQL, and NoSQL database performance. |
| NFR-22 | Validating Different Payment Gateway Settings to confirm proper transactions.   * Test PayPal, Stripe, and bank payment integrations. |
| NFR-23 | Ensuring Correct Functionality with Different Regional Settings, including language and currency adjustments.   * Switch store settings to different languages/currencies to verify conversions. |

## 6. Installation testing

Assesses the process of installing and uninstalling the software. It checks if the installation is smooth and if the software can be removed without causing issues.

|  |  |
| --- | --- |
| Non-Func. test no. | Description |
| NFR-24 | Smooth Installation Testing: Ensuring setup completes without errors on different environments.   * Verify Shopify app installation on various hosting platforms. |
| NFR-25 | Update Testing: Verifying software updates install successfully without disrupting existing data.   * Perform regression testing after updates. |
| NFR-26 | Uninstallation Testing: Checking that the system can be removed cleanly without leaving residual files or corrupting settings.   * Validate data cleanup post-uninstallation. |

7. Sanitation (Garbage) testing

involves checking for unnecessary or leftover data in the system. It ensures that the software cleans up after itself and doesn't leave unused or "garbage" data behind.

|  |  |
| --- | --- |
| Non-Func. test no. | Description |
| NFR-27 | Clearing Temporary Data: Ensuring session and cache data do not accumulate unnecessarily.   * Verify session timeouts and cache clearance mechanisms |
| NFR-28 | Removing Orphan Records: Checking databases for unused records after account deletions.   * Validate automatic cleanup of abandoned carts and inactive accounts. |
| NFR-29 | Validating Automatic Data Cleanup Routines to prevent bloated storage and performance issues.   * Test scheduled cleanup tasks for efficiency. |