# Software Test Report


**Made by:** Yaeer Gratsyani

# Table of Contents

# Project Introduction:

This STR project is a comprehensive exploration of the QA learning journey, aimed at building a structured and practical understanding of software testing. The project centers around a live e-commerce website called Music Gate, where real user flows were tested, bugs were reported, and automation was applied using Python and Selenium.

The goal of this project is to demonstrate the student's ability to design and execute a full QA process from end to end: building a test case tree, writing detailed manual tests, reporting bugs using a professional template, and creating automated scripts to validate functionality and identify issues.

By combining theory, hands-on practice, and personal insight, this project reflects not only technical proficiency but also a strong commitment to quality, structure, and professional standards.
It also demonstrates my ability to work independently, manage a full QA lifecycle, and deliver results with attention to detail and consistency.

# Website Introduction:

**Music Gate** is a well-established Israeli retailer specializing in musical instruments, accessories, and professional repair services. The company operates both online via musicgate.co.il and through physical branches located in Netanya and Petah Tikva.

The website offers a broad range of products, including guitars, keyboards, drums, audio gear, and studio equipment — all with secure checkout and free nationwide delivery on orders above ₪500. In addition, Music Gate provides in-house repair and customization services, catering to both beginners and professional musicians.

Known for its user-friendly interface, responsive customer support, and strong reputation within the local music community, Music Gate successfully combines modern e-Commerce functionality with the personal touch of a hands-on retail experience — making it a trusted platform for purchasing, upgrading, or servicing musical equipment.

As someone who owns four guitars (even though I don't play them much), I've personally purchased several items from Music Gate, such as a CORT classical guitar, a guitar strap, picks, and more — all directly from their physical store. This personal connection made the website a natural and meaningful choice for this QA project.

## You even can see a couple of pictures from their stores:


# Website Overview:

Music Gate's website is a professionally designed, Hebrew-language online platform that showcases the full scope of the brand’s offerings — from musical instruments and accessories to studio equipment, audio gear, and more. It features detailed product pages, advanced filtering tools, and secure payment options that support a smooth e-Commerce experience.

Beyond just online functionality, the website acts as a digital extension of Music Gate's physical presence in Israel, with access to in-store repair services, personalized expert advice, and dedicated customer support. The platform also includes blog articles, product demos, and exclusive deals aimed at helping musicians make informed decisions.

Whether for a beginner guitarist or a seasoned studio producer, the website is designed to deliver a seamless, reliable, and musician-focused shopping journey — combining the trust of a local store with the convenience of a nationwide online experience.

# Explanation of Each Scope:

## What is Fully Covered Scope?

A fully covered scope refers to a testing area that was thoroughly planned, executed, and documented during the course of the project. It includes the definition of clear testing objectives, the development of structured test cases, and the completion of repeated, consistent test executions. The scope is evaluated from multiple angles — including functionality, behavior, edge conditions, and consistency — and is supported by traceable evidence of the testing effort. This level of coverage provides high confidence in the quality and reliability of the component or feature, and indicates that it was tested to a professional QA standard.

## What is Partially Covered Scope?

A partially covered scope refers to a testing area that was only reviewed to a limited extent, often due to constraints such as time, resources, environment limitations, or prioritization decisions. While it may have been examined using manual, exploratory, or observational techniques, it did not receive the same level of structured attention as fully covered scopes. These areas are acknowledged in the documentation to ensure transparency and to indicate that further testing may be required before they can be considered fully validated. Partially covered scopes are treated with caution and are not assumed to be fully verified.

## What is Not Covered Scope?

A not covered scope refers to any testing area, feature, or functionality that was explicitly excluded from the testing effort during the course of the project. These areas were not reviewed, executed, or evaluated in any form — either manually or automatically. Exclusions may result from factors such as project scope definition, limited access, third-party ownership, incomplete development, or a conscious decision to defer testing to a future phase.

Documenting these areas serves to clarify the boundaries of the testing effort, prevent false assumptions about test coverage, and promote risk awareness. It also ensures transparency by clearly identifying which parts of the system remain unverified, and may require separate validation before release or future updates.

# Testing Scopes About the Website:

The STR project for Music Gate focused on testing a real-world e-Commerce platform designed for musicians, shoppers, and instrument enthusiasts. The testing effort was grounded in practical usage scenarios and included both manual and partially automated test cases, aimed at evaluating functional accuracy, user interaction, visual consistency, and system responsiveness.

Test scopes were defined and executed with attention to realistic user behavior — such as searching for products, browsing categories, adding items to the cart, initiating checkout, and navigating across devices. Each scope was selected based on its importance to the customer experience and its relevance to Music Gate's core business model.

Some scopes were fully tested through structured QA methods — including test case design, step-by-step validation, and repeated hands-on execution across multiple flows and edge cases. These included areas like sign-in and registration, cart functionality, product display, and search behavior.

Other scopes, such as payment integration, mobile performance, or advanced filtering, were only partially covered due to limited access to production systems or project time constraints. In those cases, testing was still conducted using manual exploration, interface inspection, and usability analysis — guided by QA logic and best practices.

This approach allowed the STR project to maintain a strong balance between coverage and feasibility, ensuring that Music Gate's most critical e-Commerce flows were tested in depth, while also acknowledging the boundaries of available resources — just as in any real-world QA process.

# Testing Scopes Based on The Music Gate Project

## Fully Covered Scopes:

### Functional Testing:

Validated that the system's core features behaved according to defined requirements, covering essential workflows and ensuring reliable input-output behavior across user-facing components.

### UI (User Interface) Testing:

Reviewed the visual structure and consistency of the interface, checking that elements such as layout, alignment, typography, and spacing adhered to design expectations and did not break across views.

### Usability Testing:

Evaluated how intuitive and user-friendly the interface and workflows were, focusing on ease of navigation, clarity of actions, and user satisfaction during typical tasks.

### Regression Testing:

Repeated test cases for features previously verified, to ensure they remained stable and unaffected by code changes, updates, or newly introduced functionality.

### Exploratory Testing:

Applied unscripted, intuition-driven testing to explore the system's behavior in flexible ways, with the goal of discovering unexpected issues or inconsistencies.

# Partially Covered Scopes:

## Responsiveness & Compatibility Testing

Confirmed that the website displayed and functioned properly across a range of screen sizes, devices, and modern browsers, ensuring a consistent experience regardless of platform.

## Performance Testing:

Assessed the system's responsiveness and basic load times to identify potential delays or slow interactions that could negatively affect user experience under typical usage.

## Accessibility Testing:

Conducted surface-level checks to verify that the site was navigable via keyboard and met basic accessibility principles, supporting inclusive use by users with varying needs.

## Security Awareness Testing:

Reviewed publicly visible components for basic security hygiene, including secure access points, input handling, and the presence of encryption mechanisms where applicable.

## Automation Testing:

Implemented initial automation scripts for selected flows to support faster validation, reduce manual repetition, and establish a base for future test scalability.

# Testing Tree

**The testing tree represents a structured breakdown of all test cases executed during the Music Gate STR project. It is organized by functional areas, user flows, and critical UI components to ensure full traceability and modular coverage. Each section targets a specific aspect of the website, from authentication and cart functionality to responsiveness and accessibility. This structure supports efficient test planning, execution, and reporting. It reflects a comprehensive and organized QA approach aligned with real-world e-Commerce behavior.**

**This testing tree outlines 36 structured sections covering key areas of the MusicGate web platform, including authentication, browsing, search, cart, checkout, and footer utilities.**
**Each section is organized into focused subcategories, enabling thorough validation of core features, edge cases, and UI/UX behaviors.**
**Special attention is also given to performance, accessibility, and browser compatibility to ensure a complete end-to-end QA process.**

**Note**: The Testing Tree will be shown in page 11.

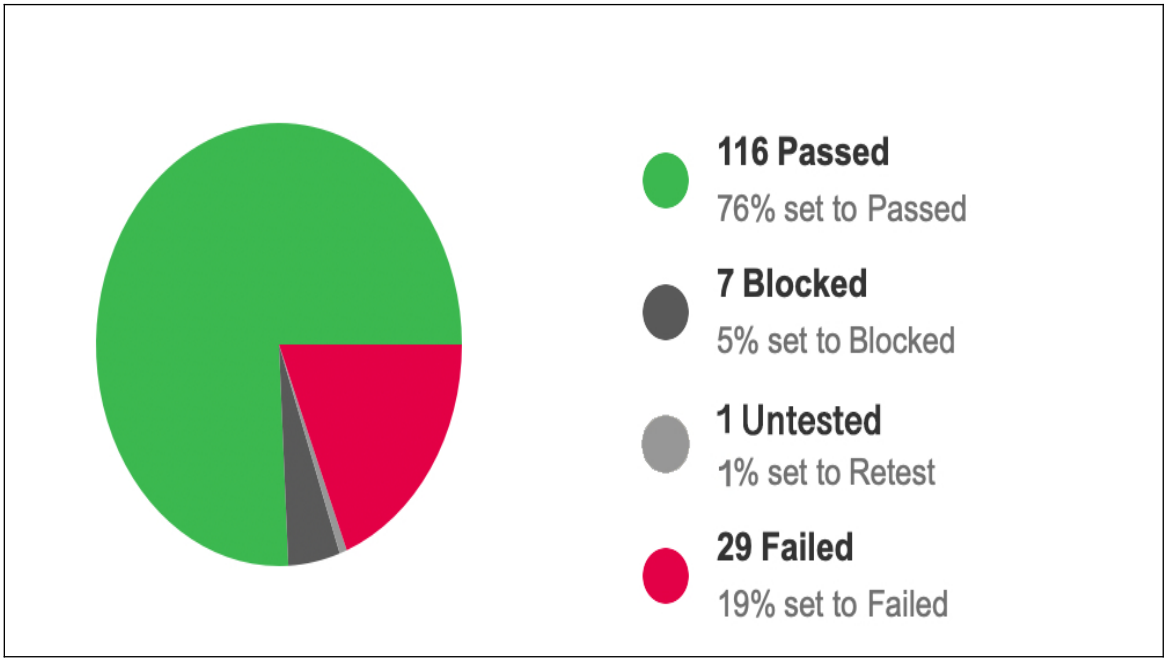
- Browsers
  - Login Browsers
    - Negative Tests
      - Logout
  - Sign-In & Registration
    - Login - Validations
    - Login - Security
    - Registration - Valid & Invalid
    - Login – Functional
    - Registration – Validations
    - Registration – Valid Cases
    - Session Management
  - Cart & Checkout
    - Cart Functionality
    - Checkout Flow
    - Cart & Checkout – Edge Cases
  - Homepage
    - Banner & Hero Components
    - Product Previews & Cards
    - Add to Cart Functionality
    - Responsive Layout & Scroll Behavior
    - Search & Autocomplete
      - Search Functionality
      - Product Filters
      - Sorting
      - Empty Results
    - Footer & Utility Links
      - Footer Links
      - Social Media Integration
      - Utility Navigation
      - Legal & Policies
      - Responsiveness
    - Categories
    - UI Product View
  - Performance & Load
  - Accessibility

# TestRail Status

This chart presents a summary of the test execution status for the QA project using TestRail. TestRail is a professional test management tool used to create, organize, and track software testing activities. It provides clear visibility into test coverage and progress throughout the project lifecycle.

Out of 153 test cases, **116 tests (76%)** were marked as **Passed**, indicating that most functionalities were successfully validated. **29 tests (19%)** were marked as **Failed**, pointing to areas needing fixes or further review. **7 tests (5%)** were marked **as Blocked**, meaning they couldn't be executed due to unresolved issues. **1 test (1%)** is remained as **Untested**, likely awaiting clarification or retest.
This distribution, as recorded in TestRail, reflects the current state of testing and helps prioritize areas for improvement moving forward.

You can see results of the status visually in the Music Gate Project:



# Test Cases by Section

Test Cases by Sections provides a structured breakdown of all test cases according to the application's key functional areas. Within TestRail, this organization enhances visibility into test coverage per module, supports efficient tracking of execution progress, and ensures systematic validation across the entire system. It allows QA teams to manage and maintain their test suites in a clear, scalable, and traceable manner.

The project has been structured into thirty-six distinct sections, each representing a specific functional area, ensuring thorough and organized test coverage across the entire application.

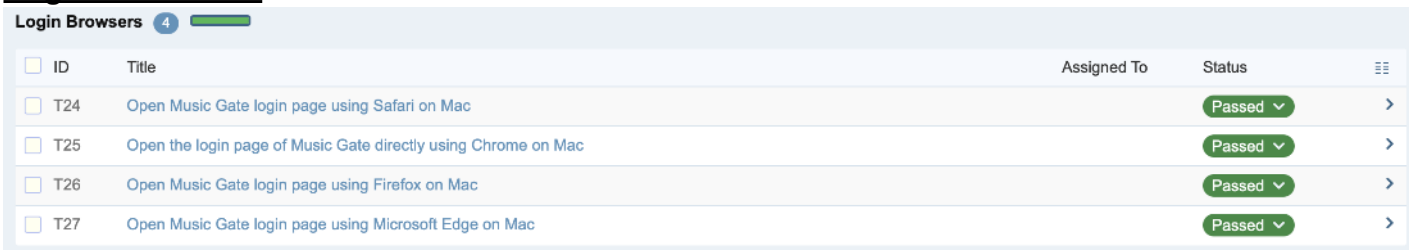
## Sections:

### Browsers:

Browsers 8

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T16 | Type "Music Gate" in English in the Safari browser | | Passed |
| T17 | Search for "Music Gate" using Chrome on Mac | | Passed |
| T18 | Search for "Music Gate" using Firefox on Mac | | Passed |
| T19 | Search for "Music Gate" using Microsoft Edge on Mac | | Passed |
| T20 | Search for "Music Gate" using Chrome Incognito on Mac | | Passed |
| T21 | Search for "Music Gate" using Microsoft Edge InPrivate on Mac | | Passed |
| T22 | Search for "Music Gate" using Firefox Private Browsing on Mac | | Passed |
| T23 | Access Music Gate directly via URL in Safari Private Browsing on Mac | | Passed |

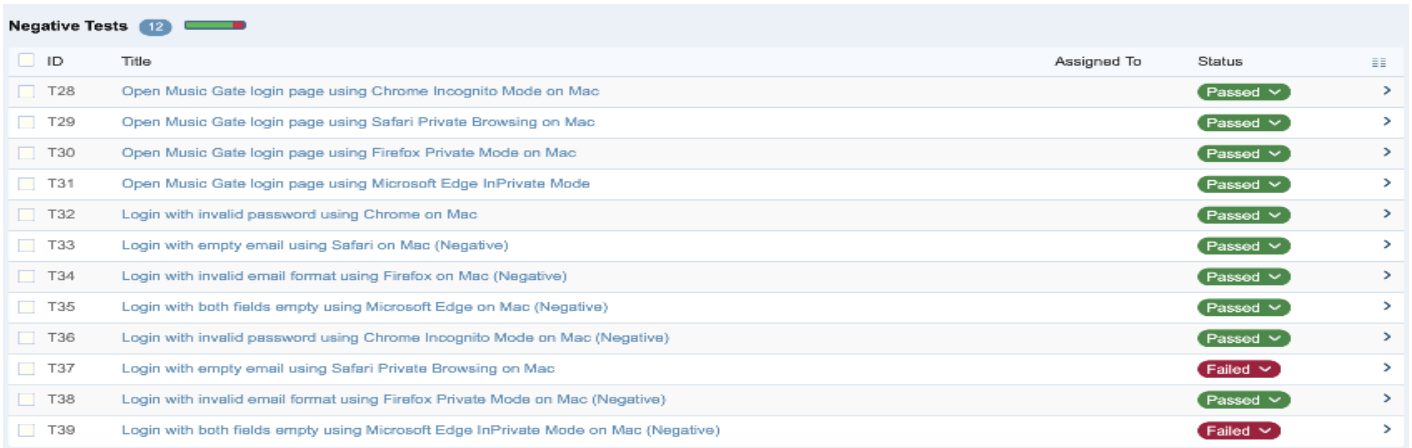
### Login Browsers:



Login Browsers 4

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T24 | Open Music Gate login page using Safari on Mac | | Passed |
| T25 | Open the login page of Music Gate directly using Chrome on Mac | | Passed |
| T26 | Open Music Gate login page using Firefox on Mac | | Passed |
| T27 | Open Music Gate login page using Microsoft Edge on Mac | | Passed |

### Negative Tests:



Negative Tests 12

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T28 | Open Music Gate login page using Chrome Incognito Mode on Mac | | Passed |
| T29 | Open Music Gate login page using Safari Private Browsing on Mac | | Passed |
| T30 | Open Music Gate login page using Firefox Private Mode on Mac | | Passed |
| T31 | Open Music Gate login page using Microsoft Edge InPrivate Mode | | Passed |
| T32 | Login with invalid password using Chrome on Mac | | Passed |
| T33 | Login with empty email using Safari on Mac (Negative) | | Passed |
| T34 | Login with invalid email format using Firefox on Mac (Negative) | | Passed |
| T35 | Login with both fields empty using Microsoft Edge on Mac (Negative) | | Passed |
| T36 | Login with invalid password using Chrome Incognito Mode on Mac (Negative) | | Passed |
| T37 | Login with empty email using Safari Private Browsing on Mac | | Failed |
| T38 | Login with invalid email format using Firefox Private Mode on Mac (Negative) | | Passed |
| T39 | Login with both fields empty using Microsoft Edge InPrivate Mode on Mac (Negative) | | Failed |

## Logout:

**Logout** 8

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T40 | Logout from Music Gate using Chrome on Mac | | Passed |
| T41 | Logout from Music Gate using Safari on Mac | | Passed |
| T42 | Logout from Music Gate using Firefox on Mac | | Passed |
| T43 | Logout from Music Gate using Microsoft Edge on Mac | | Passed |
| T44 | Logout from Music Gate using Chrome Incognito Mode on Mac | | Passed |
| T45 | Logout from Music Gate using Safari Private Browsing on Mac | | Passed |
| T46 | Logout from Music Gate using Firefox Private Mode on Mac | | Passed |
| T47 | Logout from Music Gate using Microsoft Edge InPrivate Mode on Mac | | Passed |

## Sign-In & Registration:

**Sign-In & Registration** 8

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T97 | Validate Successful Login Process (Mac, Chrome) | | Passed |
| T98 | Test Forgotten Password Flow (Mac, Chrome) | | Passed |
| T99 | Verify Unsuccessful Login Attempt with Invalid Credentials (Mac, Chrome) | | Passed |
| T100 | Ensure Successful User Registration (Mac, Chrome) | | Passed |
| T101 | Verify Email Validation During User Registration (Mac, Chrome) | | Passed |
| T102 | Test Password Strength Requirement on Registration (Mac, Chrome) | | Passed |
| T103 | Ensure User Can Log Out Successfully (Mac, Chrome) | | Failed |
| T106 | Verify Account Lockout After Multiple Failed Logins (Mac, Chrome) | | Failed |

## Login - Validations:

**Login - Validations** 4

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T104 | Confirm Successful User Login (Mac, Chrome) | | Passed |
| T107 | Check the "Forgot Password" Link Behavior (Mac, Chrome) | | Passed |
| T119 | Test Registration with Mismatched Passwords (Mac, Chrome) | | Passed |
| T122 | Verify Registration with Special Characters in Password (Mac, Chrome) | | Passed |

## Login - Security:

**Login - Security** 5

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T105 | Ensure Error Message for Invalid Credentials (Mac, Chrome) | | Failed |
| T108 | Test Login Form with Empty Fields (Mac, Chrome) | | Failed |
| T112 | Validate Error for Empty Password Field on Login (Mac, Chrome) | | Passed |
| T113 | Validate Password Masking on Sign-In Page (Chrome, Mac) | | Passed |
| T114 | Validate "Forgotten Password" Flow (Mac, Chrome) | | Passed |

## Registration – Valid & Invalid:

**Registration - Valid & Invalid** 3

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T109 | Validate Successful Registration with Strong Password (Mac, Chrome) | | Passed |
| T110 | Test Sign-Up with Already Registered Email (Mac, Chrome) | | Passed |
| T111 | Check Registration with Invalid Email Format (Mac, Chrome) | | Failed |

## Login - Functional:

**Login – Functional** 1

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T115 | Check Behavior of "Sign In" Button When Fields Are Empty (Mac, Chrome) | | Failed |

## Registration – Validations:

**Registration – Validations** 4

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T116 | Verify Password Strength Validation on Registration Page (Mac, Chrome) | | Passed |
| T118 | Ensure Registration with Already Registered Email Displays Error (Mac, Chrome) | | Passed |
| T120 | Verify Required Fields Validation on Registration Form (Mac, Chrome) | | Passed |
| T121 | Ensure "Password Strength" Indicator Appears on Registration (Mac, Chrome) | | Passed |

## Registration – Valid Cases:

**Registration – Valid Cases** 1

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T117 | Registration with Valid Unique Email and Strong Password (Mac, Chrome) | | Passed |

## Session Management:

**Session Management** 6

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T123 | Verify User Can Log Out Successfully (Mac, Chrome) | | Passed |
| T124 | Verify Session Expiry After Period of Inactivity (Mac, Chrome) | | Failed |
| T125 | Test Simultaneous Login from Two Browsers on Mac | | Passed |
| T126 | Verify Session Handling After Password Change (Mac, Chrome) | | Passed |
| T127 | Ensure Secure Cookie Handling and Session Data Storage (Mac, Chrome) | | Passed |
| T128 | Verify Session Persistence After Browser Refresh (Mac, Chrome) | | Passed |

## Cart & Checkout:

**Cart & Checkout** 0

| ID | Title | Assigned To | Status |
|---|---|---|---|

## Cart Functionality:

**Cart Functionality** 6

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T129 | Ensure Cart Icon Updates After Adding Product (Mac, Chrome) | | Failed |
| T130 | Validate Product Visibility in Cart After Addition (Mac, Chrome) | | Passed |
| T131 | Verify Cart Icon Reflects Correct Product Count (Mac, Chrome) | | Failed |
| T132 | Test Cart Behavior When Item is Removed (Mac, Chrome) | | Failed |
| T133 | Check Empty Cart Behavior (Mac, Chrome) | | Passed |
| T134 | Verify Cart Updates When Quantity is Adjusted (Mac, Chrome) | | Failed |

## Checkout Flow:

**Checkout Flow** 5

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T135 | Ensure User Can Proceed to Checkout Page (Mac, Chrome) | | Passed |
| T136 | Test Address Form Submission During Checkout (Mac, Chrome) | | Passed |
| T137 | Check Payment Method Selection Process (Mac, Chrome) | | Passed |
| T138 | Verify Cart Summary Before Payment (Mac, Chrome) | | Passed |
| T139 | Validate Successful Order Confirmation After Checkout (Mac, Chrome) | | Untested |

## Cart & Checkout – Edge Cases:

**Cart & Checkout – Edge Cases** 3

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T140 | Test Cart Persistence After Page Refresh (Mac, Chrome) | | Passed |
| T141 | Verify Cart Behavior With Multiple Item Types (Mac, Chrome) | | Passed |
| T142 | Ensure Cart Behavior After Logging In and Out (Mac, Chrome) | | Passed |

## Homepage:

**Homepage** 0

| ID | Title | Assigned To | Status |
|---|---|---|---|

## Banner & Hero Components:

**Banner & Hero Components** 8

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T62 | Verify homepage banner slider in Chrome (Regular Mode, Mac) | | Passed |
| T63 | Verify homepage banner slider in Firefox (Regular Mode, Mac) | | Passed |
| T64 | Verify homepage banner slider in Safari (Regular Mode, Mac) | | Passed |
| T65 | Verify homepage banner slider in Edge (Regular Mode, Mac) | | Passed |
| T66 | Verify homepage banner slider in Chrome Incognito Mode (Mac) | | Passed |
| T67 | Verify homepage banner slider in Firefox Private Mode (Mac) | | Passed |
| T68 | Verify homepage banner slider in Safari Private Browsing (Mac) | | Passed |
| T69 | Verify homepage banner slider in Microsoft Edge InPrivate Mode (Mac) | | Passed |

## Product Previews & Cards:

**Product Previews & Cards** 5

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T70 | Verify all links in the navigation menu (Mac, Chrome) | | Passed |
| T71 | Verify Search Bar Functionality on the Homepage (Mac, Chrome) | | Passed |
| T72 | Verify Promo Section Functionality on the Homepage (Mac, Chrome) | | Passed |
| T73 | Verify Footer Links on the Homepage (Mac, Chrome) | | Failed |
| T74 | Verify Product Previews on the Homepage (Mac, Chrome) | | Passed |

## Add to Card Functionality:

**Add to Cart Functionality** 5

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T75 | Verify Responsive Design on the Homepage (Mac, Chrome) | | Passed |
| T76 | Verify Cart Functionality on the Homepage (Mac, Chrome) | | Failed |
| T77 | Verify Homepage Load Time Performance (Mac, Chrome) | | Passed |
| T78 | Verify Product Detail Page Functionality (Mac, Chrome) | | Passed |
| T79 | Verify Navigation Menu Dropdown Functionality (Mac, Chrome) | | Passed |

## Responsive Scroll & Behavior:

**Responsive Layout & Scroll Behavior** 4

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T80 | Verify Product Image Click Behavior on the Homepage (Mac, Chrome) | | Passed |
| T81 | Verify Homepage Responsiveness on Window Resize (Mac, Chrome) | | Passed |
| T82 | Verify Search Bar Visibility on Homepage | | Passed |
| T83 | Verify Homepage Promo Banner Click Redirect | | Blocked |

## Search & Autocomplete:

**Search & Autocomplete** 6

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T84 | Verify Homepage Hero Text Is Visible and Correct | | Blocked |
| T85 | Verify Add to Cart Button Visibility on Homepage Previews | | Passed |
| T86 | Validate Functionality of Product Preview Button on Homepage (Mac, Chrome) | | Passed |
| T87 | Ensure Homepage Promo Banner Is Responsive on All Screen Sizes (Mac, Chrome) | | Blocked |
| T88 | Validate Product Preview Visibility on Homepage (Mac, Chrome) | | Passed |
| T89 | Verify Footer Link Functionality (Mac, Chrome) | | Passed |

## Search Functionality:

**Search Functionality** 3

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T143 | Verify Filters Functionality on Search Results (Mac, Chrome) | | Passed |
| T144 | Test Empty Search Results Handling (Mac, Chrome) | | Passed |
| T145 | Verify Search Results Relevance (Mac, Chrome) | | Passed |

## Product Filters:

**Product Filters** 3

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T146 | Test Filter by Price Range (Mac, Chrome) | | Passed |
| T147 | Test Filter by Category (Mac, Chrome) | | Blocked |
| T148 | Test Filter by Brand (Mac, Chrome) | | Blocked |

## Sorting:

**Sorting** 2

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T149 | Check Sorting by Price (Low to High) (Mac, Chrome) | | Blocked |
| T150 | Check Sorting by Popularity (Mac, Chrome) | | Failed |

## Empty Results:

**Empty Results** 2

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T151 | Test Sorting Behavior When No Products Match Filters (Mac, Chrome) | | Passed |
| T152 | Verify Filtered Search Results Return No Products When Filters Are Too Restrictive (Mac, Chrome) | | Failed |

## Footer & Utility Links:

**Footer & Utility Links** 7

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T90 | Validate Smooth Scrolling on Homepage (Mac, Chrome) | | Passed |
| T91 | Check Autocomplete Functionality in Search Bar (Mac, Chrome) | | Passed |
| T92 | Assess Homepage Load Time Efficiency (Mac, Chrome) | | Passed |
| T93 | Confirm Visibility of "Add to Cart" Button on Product Previews (Mac, Chrome) | | Failed |
| T94 | Verify Mobile Responsiveness of Homepage (Mac, Chrome) | | Passed |
| T95 | Verify Cart Icon Updates When Adding Products (Mac, Chrome) | | Failed |
| T96 | Ensure Promo Banner Visibility on Mobile Devices (Mac, Chrome) | | Blocked |

## Footer Links:

**Footer Links** 4

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T153 | Verify "Contact Us" Link in the Footer Works Correctly (Mac, Chrome) | | Passed |
| T154 | Check "Privacy Policy" Link Functionality in Footer (Mac, Chrome) | | Passed |
| T155 | Test "Terms & Conditions" Link in Footer (Mac, Chrome) | | Passed |
| T156 | Validate "FAQ" Link Functionality in Footer (Mac, Chrome) | | Failed |

## Social Media Integration:

**Social Media Integration** 2

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T157 | erify Social Media Buttons Link to Correct Platforms (Mac, Chrome) | | Failed |
| T158 | Ensure Social Media Links Open in a New Tab (Mac, Chrome) | | Failed |

## Utility Navigation:

**Utility Navigation** 2

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T159 | Test "Site Map" Link in Footer (Mac, Chrome) | | Passed |
| T160 | Verify "Help" Link Redirects to the Support Page (Mac, Chrome) | | Failed |

## Legal & Policies:

**Legal & Policies** 1

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T161 | Confirm Presence and Functionality of "Terms of Service" Link in Footer (Mac, Chrome) | | Passed |

## Responsiveness:

**Responsiveness** 1

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T162 | Validate Responsiveness of Footer Links on Window Resize (Mac, Chrome) | | Passed |

## Categories:

**Categories** 6

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T48 | Navigate to a product category using Chrome on Mac | | Passed |
| T49 | Navigate to a product category using Safari on Mac | | Passed |
| T50 | Navigate to a product category using Firefox on Mac | | Passed |
| T51 | Access a product category using Safari Private Browsing on Mac | | Passed |
| T52 | Access a product category using Firefox Private Mode on Mac | | Passed |
| T53 | Access a product category using Microsoft Edge InPrivate Mode on Mac | | Passed |

## UI Product View:

**UI Product View** 8

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T54 | View product detail page from category using Chrome on Mac | | Passed |
| T55 | View product detail page from category using Safari on Mac | | Passed |
| T56 | View product detail page from category using Firefox on Mac | | Passed |
| T57 | View product detail page from תופים category using Microsoft Edge on Mac | | Passed |
| T58 | View product detail page from category using Chrome Incognito Mode on Mac | | Passed |
| T59 | View product detail page from category using Safari Private Browsing on Mac | | Passed |
| T60 | View product detail page from category using Firefox Private Mode on Mac | | Failed |
| T61 | View product detail page from category using Microsoft Edge InPrivate Mode on Mac | | Failed |

## Performance & Load:

**Performance & Load** 3

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T163 | Verify Homepage Load Time is Under 2 Seconds | | Passed |
| T164 | Measure Load Time of "Guitars" Category Page | | Failed |
| T165 | Evaluate Load Time After Applying Multiple Product Filters | | Failed |

## Accessibility:

**Accessibility** 3

| ID | Title | Assigned To | Status |
|---|---|---|---|
| T166 | Verify All Interactive Elements Have Accessible Labels | | Failed |
| T167 | Check Color Contrast for Text in Banners and Buttons | | Passed |
| T168 | Verify Full Keyboard Navigation for Cart and Checkout | | Failed |

# Severity & Priority Graph Status

## Severity:

The **Severity Graph** illustrates the distribution of **reported bugs based on their impact level on the system**. Severity reflects the technical seriousness of the defect, regardless of how frequently it occurs. Here's a breakdown of the data:

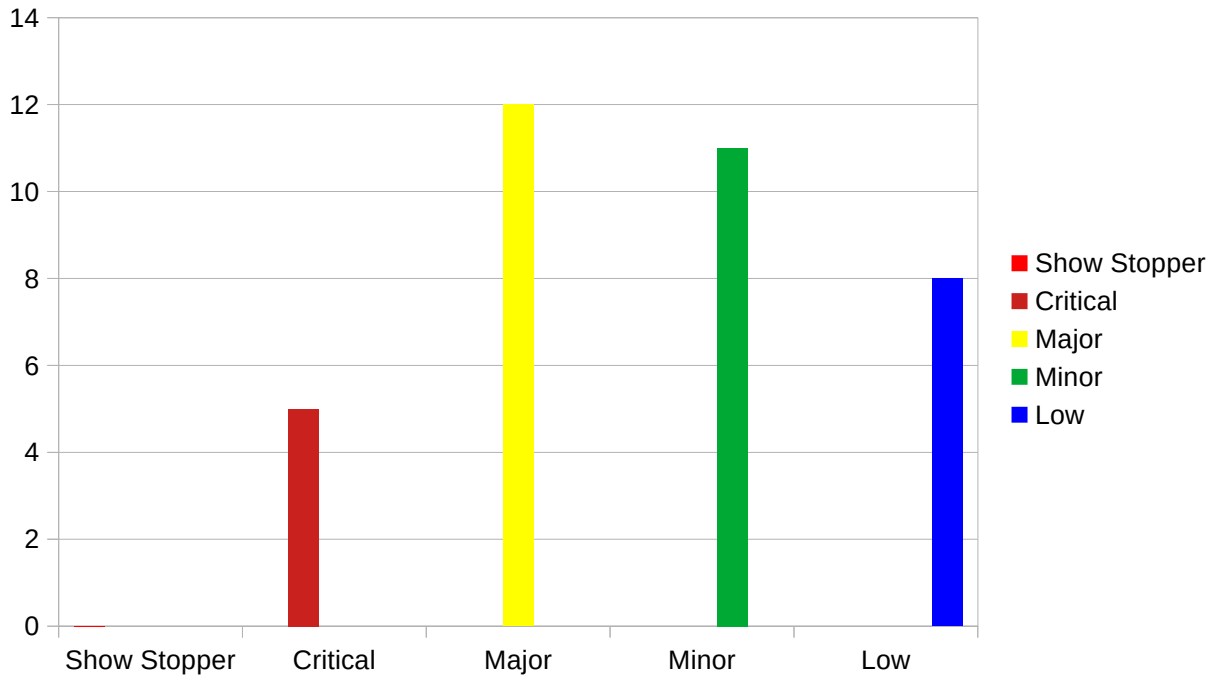
**Show Stopper (0 bugs):**
No bugs were found that completely prevent further testing or system usage. This indicates good system stability at a core level.

**Critical (5 bugs):**
A few high-severity bugs were identified that could cause system crashes, data corruption, or major feature failures. These require urgent attention.

**Major (12 bugs):**
The highest number of bugs fall under the "Major" category. These defects affect significant functionality but do not block the entire system. Fixing them is a priority for maintaining product quality.

**Minor (11 bugs):**
These bugs cause limited disruption and typically involve specific features or edge cases. They can be addressed in future sprints or lower-priority releases.

**Low (8 bugs):**
These are mostly cosmetic issues or minor inconsistencies that do not affect the user experience or functionality significantly.



## Priority:

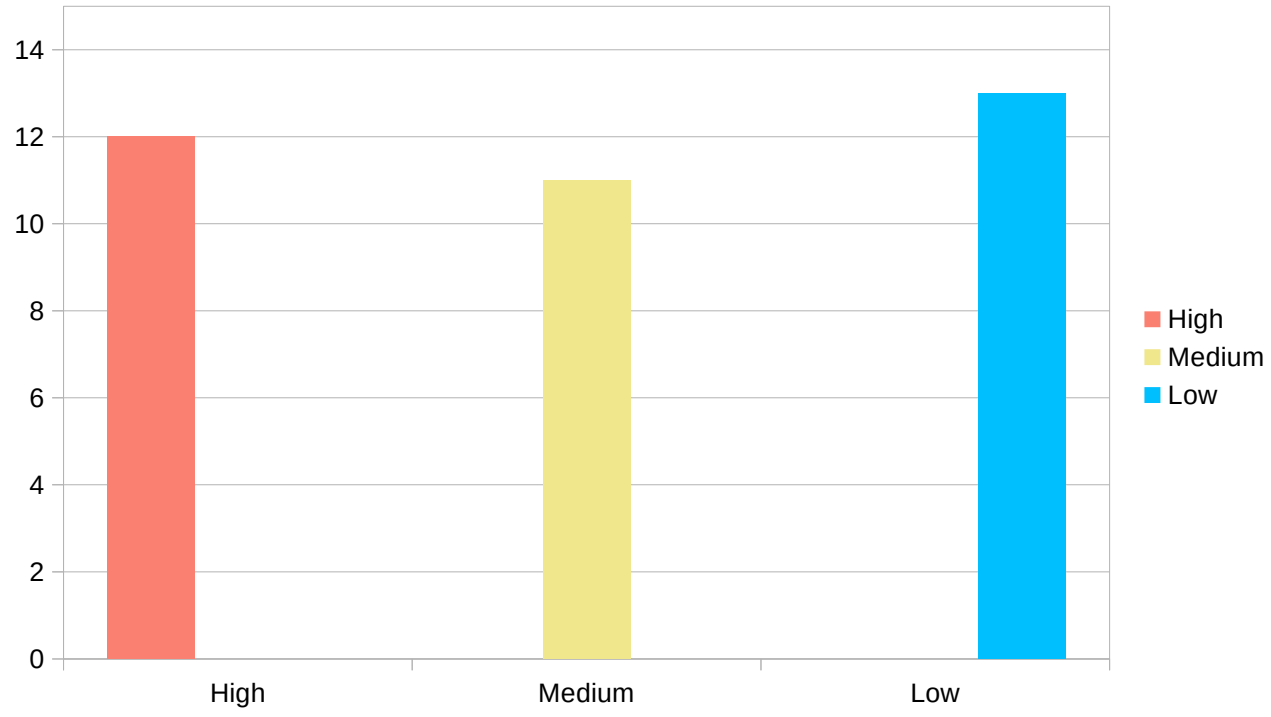
The **Priority Graph** reflects how each bug was ranked in terms of **urgency and business impact**, helping guide the development team's response order. Unlike severity, which measures technical depth, priority considers how quickly a bug should be fixed.

Here's the breakdown of the data:

**High (12 bugs):**
These bugs require immediate attention due to their business-critical impact. They might affect key user flows, system stability, or release deadlines.

**Medium (11 bugs):**
Bugs in this category are important but not urgent. They impact user experience or functionality in noticeable ways but don't require immediate fixes.

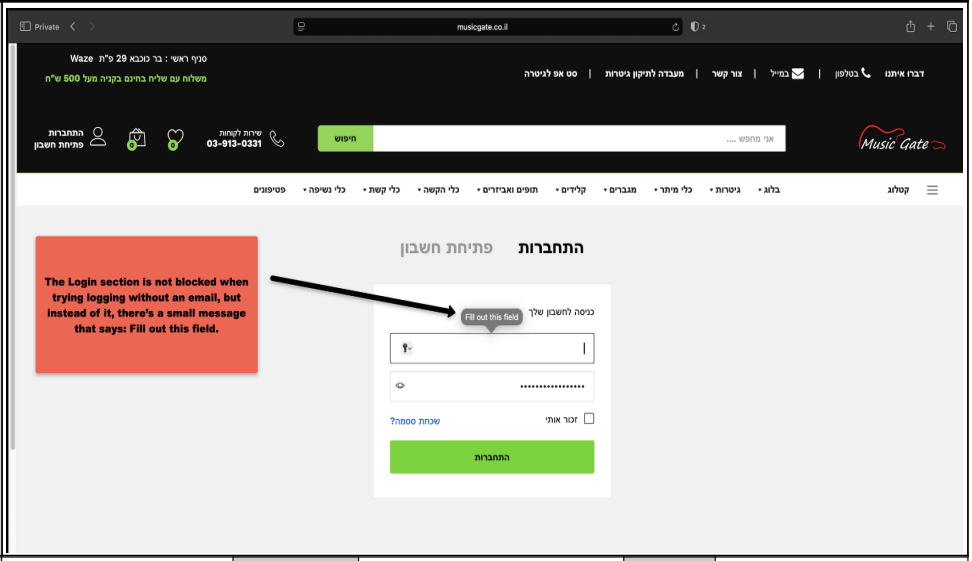
**Low (13 bugs):**
These are the least urgent and may include UI inconsistencies, content issues, or edge-case behaviors. They can be scheduled for later releases without harming core functionality.



# Bug Reports Introduction & Reports

**This section presents a structured collection of software bugs identified during testing of the Music Gate platform. Each bug was documented using a standardized reporting template, covering key QA fields such as severity, priority, environment, steps to reproduce, expected versus actual results, and technical notes. The reports reflect a variety of issues found across multiple sections of the website – including UI behavior, functionality, accessibility, responsiveness, and console-related errors. By analyzing each bug independently and providing clear evidence (screenshots and/or video links), this section demonstrates the tester's ability to identify, isolate, and communicate real issues in a professional QA context.**

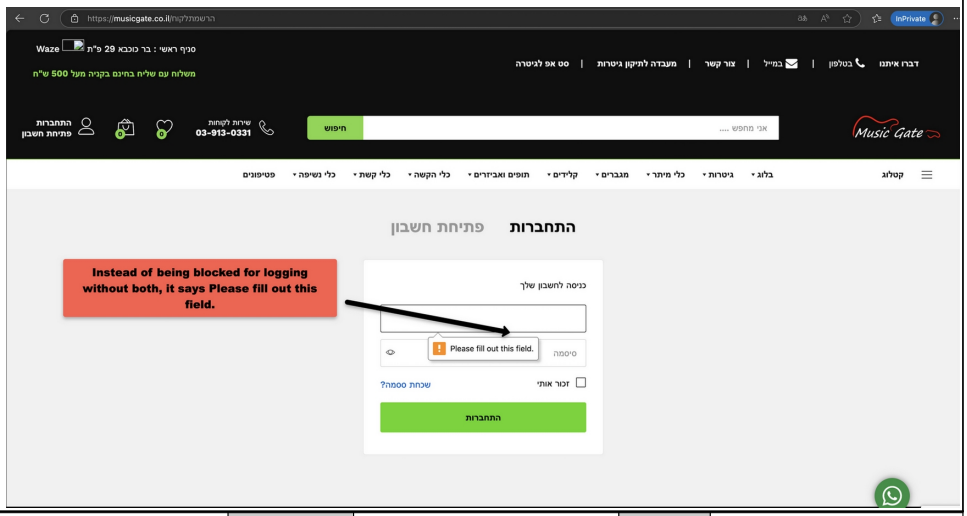
**From pages 28-64, there will be all thirty six bug reports that I've found during the project, the order of the bug reports in the order according to the section of the testing tree as you've seen in page 11.**

| | | | |
|---|---|---|---|
| **Bug Name & Number:** | B37- Login fails silently with empty email using Safari Private Browsing | | |
| **Description:** | Login fails without showing any error message when the email field is left empty in Safari Private Browsing mode. | | |
| **Status:** | Open | **Project:** | Music Gate STR Project |
| **Attachment:** |  | | |

| | | | | | |
|---|---|---|---|---|---|
| **Severity:** | Major | **Priority:** | High | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Safari | **Version:** | 18.5 (20621.2.5.11.8) | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | System should display a validation error like “Email is required” and block login. | | | | |
| **Actual Result:** | Login fails without any message: no visual feedback is provided to the user. | | | | |

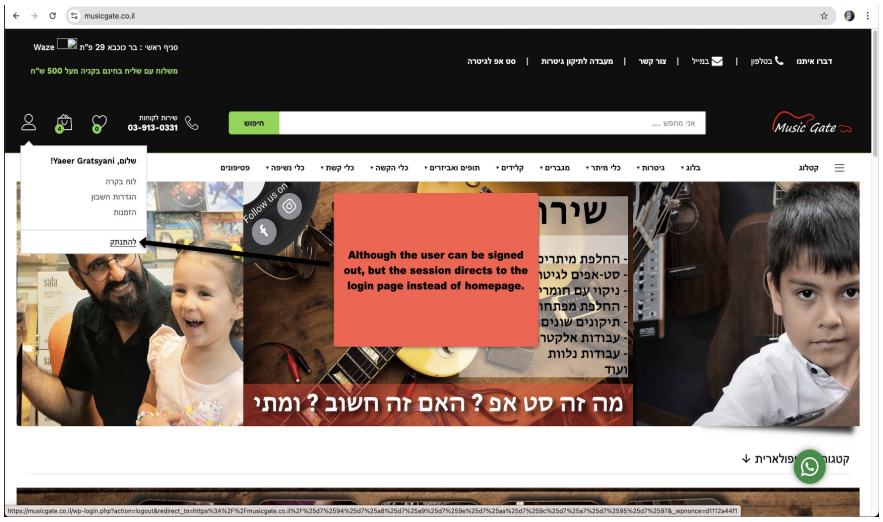
## Steps to Reproduce:

1. **Open Safari in Private Browsing mode**
2. **Navigate to https://musicgate.co.il/**
3. **Click the "התחברות" button**
4. **Leave the email field empty**
5. **Enter a vaild password**
6. **Click "התחברות"**
7. **Notice: No error is shown and login fails silently**

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B39 – Login fails silently with both fields empty using Edge InPrivate on Mac. | | | | |
| **Description:** | Login fails with no error message when both email and password fields are left empty in Edge InPrivate mode. | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** |  | | | | |
| **Severity:** | Major | **Priority:** | High | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Microsoft Edge | **Version:** | 138.0.3351.65 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | System displays an error like "Email and password are required" and blocks login. | | | | |
| **Actual Result:** | Login is rejected silently, no error message is displayed to the user. | | | | |

## Steps to Reproduce:

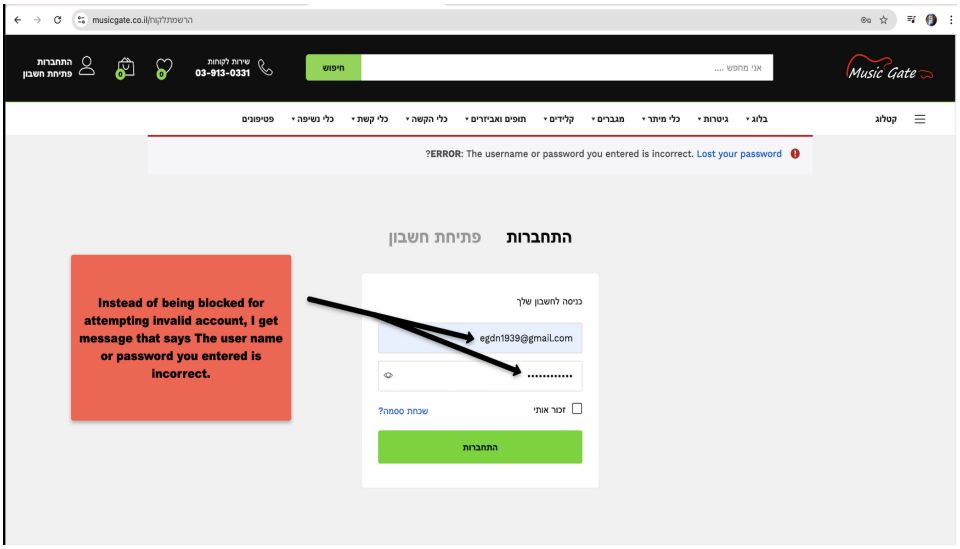
1. **Open Microsoft Edge in InPrivate Browsing mode**
2. **Navigate to https://musicgate.co.il/**
3. **Click the "התחברות" button**
4. **Leave both email and password field empty**
5. **Enter a vaild password**
6. **Click "התחברות"**
7. **Observe: No error appears, and login fails silently**

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B103 – User is not logged out properly using Chrome on Mac.<br><br>**Note:** You need to have an account to investigate the bug. | | | | |
| **Description:** | After clicking the “Logout” button, the user is not properly signed out and is still able to access protected pages without reauthentication. | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** |  | | | | |
| **Severity:** | Minor | **Priority:** | Medium | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | After logging out, the user should be fully signed out and redirected to the homepage. | | | | |
| **Actual Result:** | User is signed out, but instead of being redirected to the homepage, the session ends at the login page. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to** https://musicgate.co.il/
3. **Click the "התחברות" button**
4. **Login to your account**
5. **Click "התנתק"**
6. **Observe: After you logged out, it directs you to the login page instead to the homepage**

**Note:** if this bug is not understandable yet, I’ll send a video of this bug.

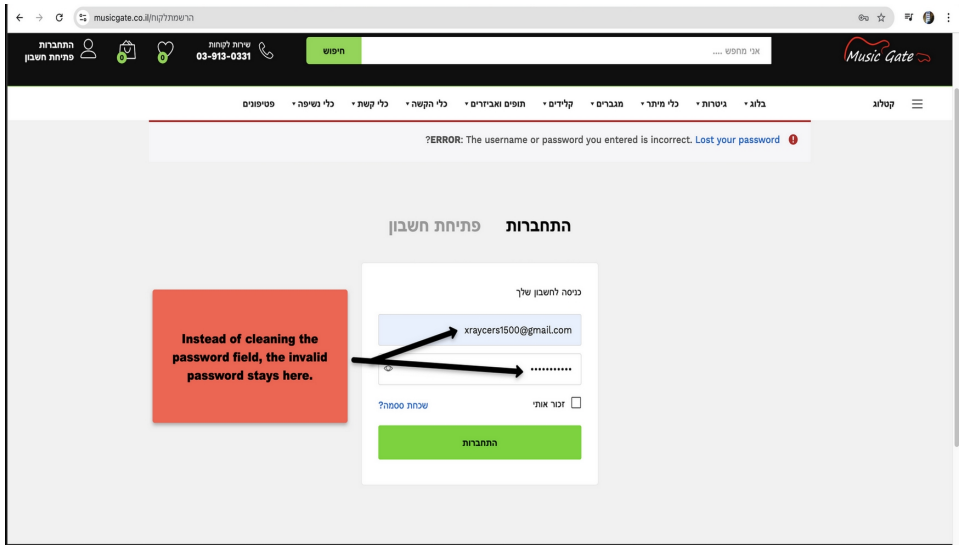
| | | | |
|---|---|---|---|
| **Bug Name & Number:** | B106 – No lockout triggered after multiple failed login attempts in Chrome on Mac | | |
| **Description:** | User not blocked after 5 wrong logins; no CAPTCHA or lockout triggered. | | |
| **Status:** | Open | **Project:** | Music Gate STR Project |
| **Attachment:** |  | | |

| | | | | | |
|---|---|---|---|---|---|
| **Severity:** | Major | **Priority:** | Medium | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | After 5 failed attempts, user should be locked out or shown CAPTCHA prompt. | | | | |
| **Actual Result:** | User can repeatedly attempt login without restriction; no lockout is enforced. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to https://musicgate.co.il/**
3. **Click the "התחברות" button**
4. **Enter invalid credentials and click התחברות**
5. **Repeat the invaild login process 5 times in a row**
6. **Observe the behavior after 5th attempts**
7. **Confirm that user is not locked out or prompted with CAPTCHA**

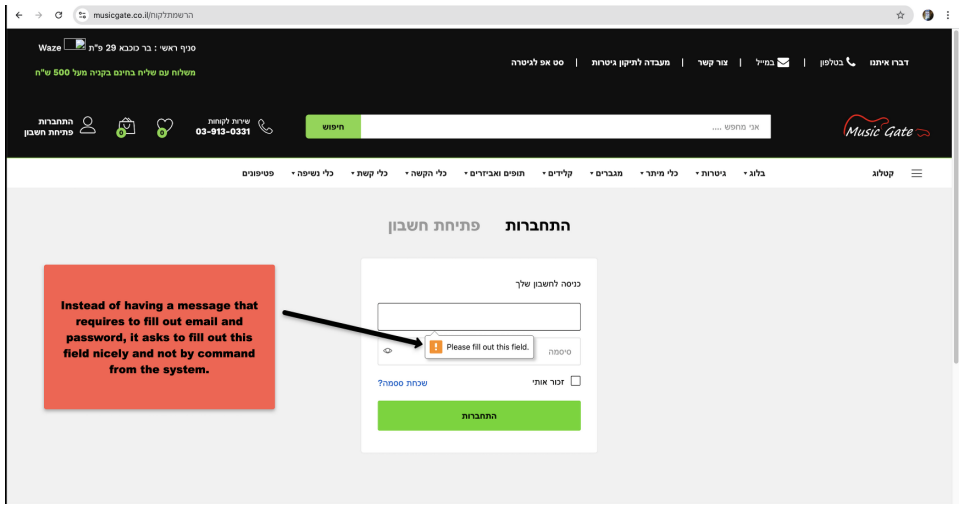
**Note:** if this bug is not understandable yet, I'll send a video of this bug.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B105 – Intermittent login error message issue on Chrome (Mac) | | | | |
| **Description:** | Error message doesn't always appear after invalid login; user gets no feedback. | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** |  | | | | |
| **Severity:** | Major | **Priority:** | High | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Invalid login always shows error; password field is cleared. | | | | |
| **Actual Result:** | Sometimes no message shown; user stays on page with fields filled. | | | | |

## Steps to Reproduce:

1. Open Google Chrome
2. Navigate to https://musicgate.co.il/
3. Click the "התחברות" button
4. Enter a vaild email address
5. Enter an incorrect password
6. Click the "התחברות" button
7. Sometimes no error is shown; user stays on the page silently

**Note:** if this bug is not understandable yet, I'll send a video of this bug.

| | | | |
|---|---|---|---|
| **Bug Name & Number:** | B108 – Login form allows empty fields on Chrome without warning or error | | |
| **Description:** | User can submit login form with empty fields; no error or feedback shown. | | |
| **Status:** | Open | **Project:** | Music Gate STR Project |
| **Attachment:** |  | | |

| | | | | | |
|---|---|---|---|---|---|
| **Severity:** | Major | **Priority:** | Medium | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Login button should be disabled or show a clear message about required fields. | | | | |
| **Actual Result:** | Clicking “Sign In” with empty fields gives no error; login fails silently. | | | | |

## Steps to Reproduce:

1. Open Google Chrome
2. Navigate to https://musicgate.co.il/
3. Click the "username" **icon**
4. Enter a vaild email address
5. Click the "התחברות" button
6. Confirm login does not proceed and user remains on page

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B111 – Invalid email format not detected on registration page | | | | |
| **Description:** | User can register with wrong email format; no error is shown. | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** | I'm not posting a video or an image because of privacy.<br><br>**Note:** For privacy reasons, a functional test email was used in the format of example@gmail.com without a valid structure. | | | | |
| **Severity:** | Critical | **Priority:** | High | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | System should show "Invalid email format" and block registration when email is malformed. | | | | |
| **Actual Result:** | Form is accepted without email validation; registration process continues without error. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to** https://musicgate.co.il/
3. **Click the "username" icon**
4. **Switch and choose the "פתיחת חשבון" button**
5. **Enter invalid email (e.g., "user@")**
6. **Fill in remaining required fields**
7. **Submit form**
8. **Observe system does not validate email format**

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B115 – Registration form accepts invalid email format without error | | | | |
| **Description:** | User can register with invalid email format (e.g., "user@") ; system does not vaildate and proceeds without warning. | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** | I'm not posting a video or an image because of privacy.<br><br>**Note:** For privacy reasons, a functional test email was used in the format of example@gmail.com without a valid structure. | | | | |
| **Severity:** | Critical | **Priority:** | High | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | System should block submission and display "Invalid email format" error. | | | | |
| **Actual Result:** | Form is accepted and registration continues without validating email format. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to** https://musicgate.co.il/
3. **Click the "username" icon**
4. **Switch and choose the "**פתיחת חשבון**" button**
5. **Enter an invalid email format (e.g.,** `g.user@`**)**
6. **Fill other required fields**
7. **Submit the form**
8. **Observe that no validation error is shown and registration proceeds**

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B124 – Session does not expire after inactivity period (Mac, Chrome) | | | | |
| **Description:** | User remains logged in after inactivity timeout; no logout or warning appears. | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** | This is clear and professional.<br>Let me know if you'd like to add a footnote or internal tester note indicating the exact time measured (e.g. 10 minutes), or if you prefer to leave it general. | | | | |
| **Severity:** | Critical | **Priority:** | High | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | User should be logged out after inactivity timeout.<br>A message should appear and user should be redirected to the login page.<br>Any action should require login again. | | | | |
| **Actual Result:** | User remains logged in after timeout.<br>No session expiration message appears; actions still work without re-login. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to** https://musicgate.co.il/
3. **Click the "username" icon**
4. **Log in using valid credentials**
5. **Click "התחברות" button**
6. **Leave session inactive for ~10 minutes**
7. **Attempt to interact with the page (e.g., click link or cart)**
8. **Observe that user is still logged in with no message or redirect**

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B129 – Cart icon doesn't update after adding product (Mac, Chrome) | | | | |
| **Description:** | User adds item, but cart icon doesn't reflect the change visually. | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** | Video included to demonstrate visual behavior (cart icon not updating).<br>Note: The bug is also fully described below for clarity, even without watching the video.<br><br>https://www.youtube.com/watch?v=8fxYcrYI4Fw | | | | |
| **Severity:** | Major | **Priority:** | High | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Cart icon should visually update to reflect the correct number of added items. | | | | |
| **Actual Result:** | Cart icon remains unchanged after adding a product; no visual feedback appears. | | | | |

## Steps to Reproduce:

1. Open Google Chrome
2. Navigate to https://musicgate.co.il/
3. **Go to the Music Gate homepage**
4. Locate any visible product preview
5. **Click the "Add to Cart" button**
6. Watch the cart icon after clicking
7. Observe that the icon doesn't reflect item count change

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B131 – Cart icon doesn't reflect correct product count (Mac, Chrome) | | | | |
| **Description:** | Cart icon fails to update item count when multiple products are added. | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** | This video was also used in bug B129. It demonstrates related but distinct behavior - item quantity increases, but cart icon count does not update<br><br>https://www.youtube.com/watch?v=8fxYcrYI4Fw | | | | |
| **Severity:** | Major | **Priority:** | Medium | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Cart icon should show updated item count as each product is added to the cart. | | | | |
| **Actual Result:** | Cart icon stays the same after adding products; count does not increase. | | | | |

## Steps to Reproduce:

1. Open Google Chrome
2. Navigate to https://musicgate.co.il/
3. **Go to the Music Gate homepage**
4. Locate any visible product preview
5. **Click the "Add to Cart" button**
6. Add another product to the cart
7. **Notice the cart icon count does not reflect the correct total**
8. Observe that the icon doesn't reflect item count change

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B132 – Cart icon doesn't update when item is removed (Mac, Chrome) | | | | |
| **Description:** | After removing item from cart, icon count stays the same and incorrect. | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** | You can watch the video that I'm removing a product from the cart and quantity isn't updating itself, It's pretty hard to explain it verbally.<br>https://www.youtube.com/watch?v=BC9fGj769L0 | | | | |
| **Severity:** | Major | **Priority:** | Medium | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Product should be removed from cart and icon should reflect updated count. | | | | |
| **Actual Result:** | Item is removed, but cart icon still shows incorrect product count. | | | | |

## Steps to Reproduce:

1. Open Google Chrome
2. Navigate to https://musicgate.co.il/
3. **Go to the Music Gate homepage**
4. **Click the "Username" icon**
5. **Navigate to the cart page**
6. **Click the "Remove" button next to any product**
7. **Confirm that item disappears from cart**
8. **Observe the cart icon in the header**
9. **Notice the count is not updated correctly**

**Note**: You need to have a valid account to add products. Without it, sometimes it won't add or will take some time to add.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B134 – Cart icon fails to update after quantity adjustment (Mac, Chrome) | | | | |
| **Description:** | Cart icon and price don’t update when item quantity is changed. | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** | No video/image attached due to time and consistency constraints. | | | | |
| **Severity:** | Major | **Priority:** | High | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Cart and price update correctly after quantity is changed. | | | | |
| **Actual Result:** | Cart icon and price remain unchanged after adjusting quantity. | | | | |

## Steps to Reproduce:

1. Open Google Chrome
2. Navigate to https://musicgate.co.il/
3. Go to the Music Gate homepage
4. Click the "Username" icon
5. Login to your account
6. Navigate to the cart page
7. Adjust product quantity (e.g., from 1 to 2)
8. Observe price and cart icon count
9. Icon and price don’t reflect updated quantity

**Note:** You need to have a valid account to add products. Without it, sometimes it won’t add or will take some time to add.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B73 – Facebook Footer link fail is broken on homepage (Mac, Chrome) | | | | |
| **Description:** | Clicking the Facebook link in the footer leads to a broken or empty page. | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** | No video/image attached due to low severity and clear reproduction. | | | | |
| **Severity:** | Low | **Priority:** | Low | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | All footer links should open valid social pages without error. | | | | |
| **Actual Result:** | Facebook link fails to load; Instagram works correctly. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to https://musicgate.co.il/**
3. **Go to the Music Gate homepage**
4. **Scroll to the footer section**
5. **Click the Facebook icon/link**
6. **Observe the error or failure to load**
7. **Click the Instagram link – it works properly**

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B76 – Cart icon fails to update after product added (Mac, Chrome) | | | | |
| **Description:** | "Add to Cart" updates product list but cart icon doesn't change. | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** | This issue follows the same behavior as earlier cart icon update bugs. No video is attached. All relevant steps and actual behavior are documented in detail below. | | | | |
| **Severity:** | Major | **Priority:** | High | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Cart icon count should match added products. | | | | |
| **Actual Result:** | Cart icon fails to reflect new product added. | | | | |

## Steps to Reproduce:

1. Open Google Chrome
2. Navigate to https://musicgate.co.il/
3. Wait for homepage to load
4. Find a product preview
5. Click "Add to Cart"
6. Wait for cart to update (expect count to rise)
7. Observe the cart icon – it stays unchanged
8. Click cart icon and verify the product is in the cart

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B83 – Promo banner is missing from homepage (Mac, Chrome) | | | | |
| **Description:** | Promo banner does not appear on homepage. | | | | |
| **Status:** | Blocked | **Project:** | Music Gate STR Project | | |
| **Attachment:** | No screenshot attached – banner is not visible on the page. | | | | |
| **Severity:** | Low | **Priority:** | Low | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Banner should appear and link to a valid landing page. | | | | |
| **Actual Result:** | No banner is displayed on the homepage. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to** https://musicgate.co.il/
3. **Wait for homepage to load completely**
4. **Scroll through homepage**
5. **Observe absence of any promo banner**

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B84 – Hero section text is not displayed on homepage (Mac, Chrome) | | | | |
| **Description:** | Hero title text is missing from the homepage. | | | | |
| **Status:** | Blocked | **Project:** | Music Gate STR Project | | |
| **Attachment:** | No screenshot attached – hero text not visible. | | | | |
| **Severity:** | Low | **Priority:** | Low | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Hero section should display aligned and readable heading text. | | | | |
| **Actual Result:** | No banner is displayed on the homepage. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to https://musicgate.co.il/**
3. **Locate the hero section at the top of the page**
4. **Verify visibility of heading and subheading text**
5. **Confirm spelling and alignment – no text found**

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B87 – Promo banner responsiveness test blocked due to missing banner (Mac, Chrome) | | | | |
| **Description:** | No promo banner appears on homepage, test blocked on all screen sizes. | | | | |
| **Status:** | Blocked | **Project:** | Music Gate STR Project | | |
| **Attachment:** | No image/video attached since no banner is present. | | | | |
| **Severity:** | Low | **Priority:** | Low | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Banner should scale smoothly across devices and remain visible and readable. | | | | |
| **Actual Result:** | No promo banner exists on the homepage. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to** https://musicgate.co.il/
3. **Look for a promo banner on the homepage**
4. **Attempt to resize the screen to various widths**
5. **Confirm that no promo banner is rendered**

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B147 – Category filter doesn't affect search results (Mac, Chrome) | | | | |
| **Description:** | Applying category filter can't be applied in the search results, it doesn't exist. | | | | |
| **Status:** | Blocked | **Project:** | Music Gate STR Project | | |
| **Attachment:** | No video/image attached – test is blocked due to missing functionality. | | | | |
| **Severity:** | Low | **Priority:** | Low | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Search results should only show products matching the selected category. | | | | |
| **Actual Result:** | Category filter has no effect — unrelated products are still shown. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to** https://musicgate.co.il/
3. **Enter a search term (e.g., "smartphone")**
4. **Apply category filter (e.g., "Category: Electronics")**
5. **Observe filtered results**

**Note:**
Severity and Priority are both marked as Low, as the specific category used in this test ("Electronics") does not exist on the website and is not relevant to a music-focused store. This test may have been misaligned with the domain of the website.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B148 – Brand filter doesn't exist or doesn't affect search results (Mac, Chrome) | | | | |
| **Description:** | Brand filter has no effect — unrelated brands are still shown or the filter doesn't exist at all. | | | | |
| **Status:** | Blocked | **Project:** | Music Gate STR Project | | |
| **Attachment:** | No screenshot or video provided due to test irrelevance and redundancy across filter cases. | | | | |
| **Severity:** | Minor | **Priority:** | Low | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Only selected brand products should appear, updated instantly without errors. | | | | |
| **Actual Result:** | Filter missing or broken – unrelated brand results still appear. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to** https://musicgate.co.il/
3. **Enter a search term (e.g., "laptop") in the search bar**
4. **Apply brand filter (e.g., "Brand: Dell")**
5. **Observe the filtered results**

**Note:**
Severity and Priority are both **Low**, because the specific brand used in this test doesn't exist and isn't relevant to a music store.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B149 – Price sorting option is missing or not functioning (Mac, Chrome) | | | | |
| **Description:** | Sorting option “Price: Low to High” is missing or does not apply sorting correctly. | | | | |
| **Status:** | Blocked | **Project:** | Music Gate STR Project | | |
| **Attachment:** | No screenshot or video needed – functionality is missing or unavailable for this filter. | | | | |
| **Severity:** | Minor | **Priority:** | Low | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Products should appear in ascending price order, cheapest items shown first. | | | | |
| **Actual Result:** | Sorting option is missing or doesn’t apply any change to product order. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to https://musicgate.co.il/**
3. **Enter a search term (e.g., "headphones") in the search bar**
4. **Try to apply sorting option “Price: Low to High”**
5. **Observe if any sorting occurs**

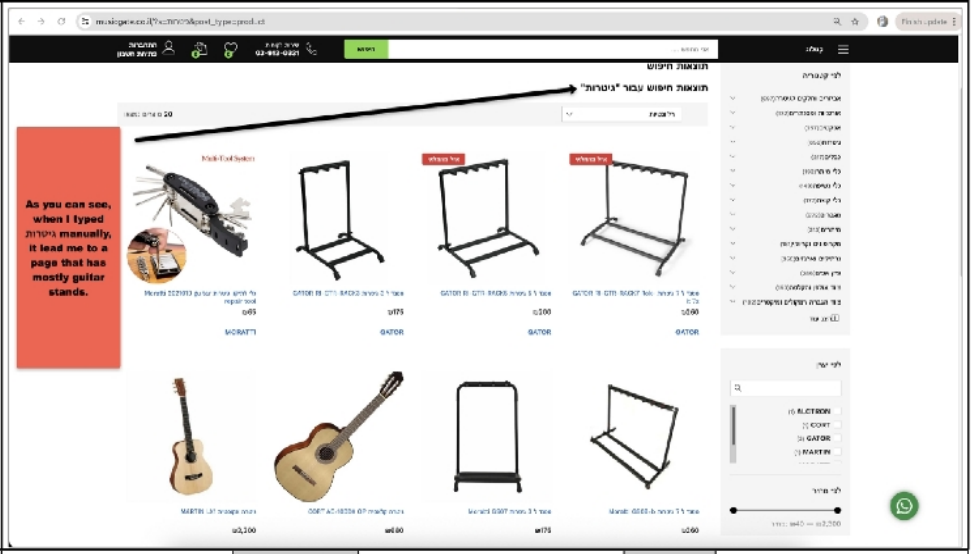
**Note:**
Severity and Priority are both **low**, because the sorting option is missing and likely irrelevant due to limited product inventory.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B150 – Popularity sorting doesn't apply or has no effect (Mac, Chrome) | | | | |
| **Description:** | Sorting by popularity does not change product order - results remain the same regardless of selection. | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** | No screenshot or video provided – product list remains static with no visual indication of popularity-based sorting. | | | | |
| **Severity:** | Minor | **Priority:** | Low | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Products should be sorted by popularity, showing top-rated items first only. | | | | |
| **Actual Result:** | No visible sorting occurs — product list appears unchanged regardless of sorting selection. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to** https://musicgate.co.il/
3. **Enter a search term (e.g., "smartphone") in the search bar**
4. **Apply the sorting option "Popularity" in Hebrew**
5. **Observe the order of products**

**Note:**
Popularity filter has no visible effect — sorting logic may be missing or inactive on front end.

| | | | |
|---|---|---|---|
| **Bug Name & Number:** | B152 – Price range filter is available but irrelevant due to inaccurate product targeting (Mac, Chrome) | | |
| **Description:** | Price filter (slider) is displayed but fails to accurately limit product types — e.g., search for “guitar” returns unrelated accessories. | | |
| **Status:** | Open | **Project:** | Music Gate STR Project |
| **Attachment:** |  | | |

| | | | | | |
|---|---|---|---|---|---|
| **Severity:** | Major | **Priority:** | Medium | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | User should be able to filter product search by **`type and price`** — e.g., “guitars” between ₪1000–₪2000 should return actual guitars only. | | | | |
| **Actual Result:** | Price slider appears, but search results include unrelated accessories (cases, racks, interfaces), ignoring product type context. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to https://musicgate.co.il/**
3. **Type "גיטרות" in the search bar**
4. **Apply price range filter (e.g., ₪1000–₪2000)**
5. **Observe if non-guitar items (cases, stands, audio gear) appear**

**Note:**
This is a **filter misalignment bug** — price slider is functional but **search logic** does not limit results to relevant product types.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B93 – “Add to Cart” Button Missing or Not Functional on Product Page (Mac, Chrome) | | | | |
| **Description:** | Button missing or fails to update cart when clicked or an item is out of stock. | | | | |
| **Status:** | Closed – a logistical / design issue | **Project:** | Music Gate STR Project | | |
| **Attachment:** | It’s pretty hard to explain verbally but you can watch the video and understand that the bug is a logistical or design issue:<br>https://www.youtube.com/watch?v=OSzeI4mC7MU | | | | |
| **Severity:** | Minor | **Priority:** | Medium | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Homepage should display “Add to Cart” button with cart icon update. | | | | |
| **Actual Result:** | No “Add to Cart” button appears on homepage product previews. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to https://musicgate.co.il/**
3. **Scroll through homepage product previews**
4. **Observe missing or inactive “Add to Cart” button**

**Note:**
This is not a technical bug. The missing button is a **logistical decision or design limitation**, possibly intentional by the site owner.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B95 – Cart icon doesn’t update after adding product (Mac, Chrome) | | | | |
| **Description:** | Cart icon doesn’t reflect item count after clicking "Add to Cart". | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** | No screenshot or video provided – visual issue observed consistently across multiple products. | | | | |
| **Severity:** | Major | **Priority:** | High | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Cart icon should update item count and show the correct item in cart. | | | | |
| **Actual Result:** | Clicking "Add to Cart" doesn’t update the icon or item count. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to** https://musicgate.co.il/
3. **Locate any product with visible "Add to Cart" button**
4. **Click the button**
5. **Observe the cart icon in the header**

**Note:**
This is a visual and functional issue. A short video is highly recommended to capture the lack of cart icon update. This ensures the issue is clearly understood since no error is shown and behavior appears static.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B96 – Promo banner missing completely in mobile view (Mac, Chrome) | | | | |
| **Description:** | Promo banner is hidden or unreadable in mobile screen simulation. | | | | |
| **Status:** | Blocked | **Project:** | Music Gate STR Project | | |
| **Attachment:** | No screenshot attached – test blocked due to lack of mobile-specific promo banner rendering. | | | | |
| **Severity:** | Low | **Priority:** | Low | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | **Google Chrome** (DevTools mobile view: 375px) | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Promo banner should appear on mobile, properly scaled and aligned. | | | | |
| **Actual Result:** | No promo banner is displayed at all on mobile view. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to** https://musicgate.co.il/
3. **Open DevTools and simulate screen size of 375px**
4. **Look for the promo banner**

**Note:**
Test blocked — no promo banner exists for mobile or desktop view.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B156 – FAQ link missing entirely from footer (Mac, Chrome) | | | | |
| **Description:** | FAQ link is missing from the footer section on homepage. | | | | |
| **Status:** | Failed | **Project:** | Music Gate STR Project | | |
| **Attachment:** | No screenshot attached – issue verified manually by scrolling to footer. | | | | |
| **Severity:** | Minor | **Priority:** | Low | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | **Google Chrome** (DevTools mobile view: 375px) | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Footer should display a working FAQ link, styled and functional. | | | | |
| **Actual Result:** | No FAQ link appears in the footer on homepage. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to** https://musicgate.co.il/
3. **Scroll down to the bottom of the homepage**
4. **Look for the FAQ link in the footer**

**Note:**
FAQ link is completely missing - cannot verify functionality of the FAQ page.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B157 – Facebook button redirects to invalid or incorrect page (Mac, Chrome) | | | | |
| **Description:** | Facebook button fails to redirect to the official Facebook page. | | | | |
| **Status:** | Failed | **Project:** | Music Gate STR Project | | |
| **Attachment:** | No screenshot attached – tested manually; Facebook redirect failed. | | | | |
| **Severity:** | Low | **Priority:** | Low | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | **Google Chrome** (DevTools mobile view: 375px) | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Clicking the Facebook button should open the official Facebook page in a new tab. | | | | |
| **Actual Result:** | Facebook button redirects to an invalid or unrelated page. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to https://musicgate.co.il/**
3. **Scroll to the footer section**
4. **Click the Facebook icon**
5. **Observe that the redirect is incorrect or broken**

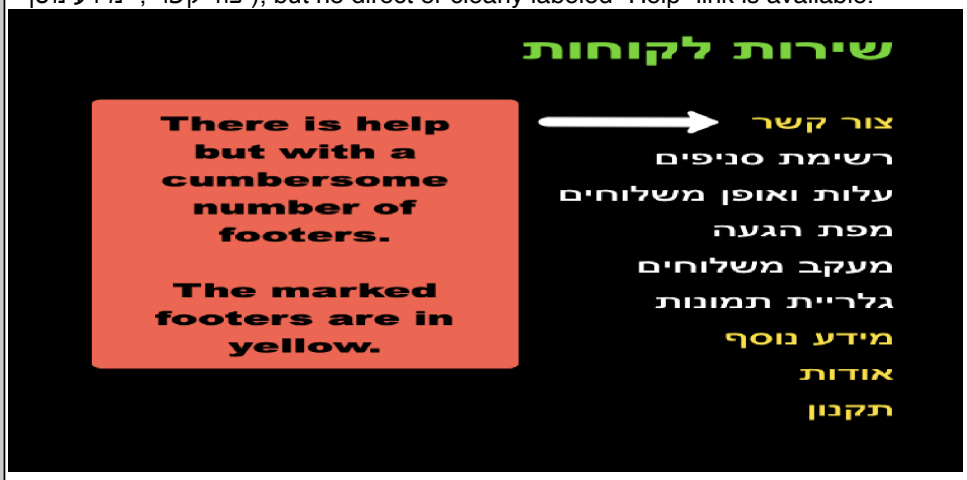
**Note:**
Test failed — Instagram redirect works as expected. Facebook link leads to the wrong location or does not load properly.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B158 – Facebook link in footer is broken (Mac, Chrome) | | | | |
| **Description:** | Facebook social icon redirects to invalid or broken link. | | | | |
| **Status:** | Failed | **Project:** | Music Gate STR Project | | |
| **Attachment:** | No screenshot attached – issue confirmed through manual test; Facebook link leads to a 404 or unrelated page. | | | | |
| **Severity:** | Low | **Priority:** | Low | **Type:** | Functional |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Both Facebook and Instagram links should open in a new tab and redirect to their correct official pages. | | | | |
| **Actual Result:** | Facebook link opens in a new tab but leads to a broken or invalid destination. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to** https://musicgate.co.il/
3. **Scroll down to the footer**
4. **Click on the Facebook icon**
5. **Observe that the link opens in a new tab but doesn't redirect correctly**

**Note:**
Instagram link behaves as expected. Only the Facebook button is broken — the destination page is not valid or does not exist.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B160 – Help link missing from footer (Mac, Chrome) | | | | |
| **Description:** | The "Help" link is not present in the footer, reducing support accessibility. | | | | |
| **Status:** | Failed | **Project:** | Music Gate STR Project | | |
| **Attachment:** | Screenshot attached showing footer section. Various support-related links exist (e.g., "צור קשר","מידע נוסף"), but no direct or clearly labeled “Help” link is available.<br>  | | | | |
| **Severity:** | Minor | **Priority:** | Medium | **Type:** | Usability |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | A visible “Help” link should appear in the footer and redirect to a centralized support/help page. | | | | |
| **Actual Result:** | The footer contains several related links, but no clearly marked “Help” link exists. Users may find it hard to locate support. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to** https://musicgate.co.il/
3. **Scroll down to the footer**
4. **Look for a “Help” or support-related link**
5. **Observe the absence of a direct “Help” label**

**Note:**
Test failed – no centralized help link. Although there are multiple related links, none of them clearly indicate general help, which may affect user experience and support discovery.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B60 – Product Detail Page Layout Broken in Firefox Private Mode | | | | |
| **Description:** | Product detail page displays incomplete layout when in Private Mode. | | | | |
| **Status:** | Failed | **Project:** | Music Gate STR Project | | |
| **Attachment:** | Issue involves visual layout failure (e.g., missing “Add to Cart” button) that cannot be effectively communicated with words or static screenshots.<br>Video required due to UI distortion:<br>https://www.youtube.com/watch?v=_fp2cTmUhzQ | | | | |
| **Severity:** | Minor | **Priority:** | Medium | **Type:** | Usability |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Firefox | **Version:** | 140.0.4 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Product layout should include title, image, price, description, button. | | | | |
| **Actual Result:** | Some product pages are missing cart button or price in Private Mode. | | | | |

## Steps to Reproduce:

1. **Launch Firefox in Private Mode on Mac**
2. **Navigate to https://musicgate.co.il/**
3. **Click on a visible product category**
4. **Select a product**
5. **Wait for product detail page to load fully**
6. **Observe missing layout elements (e.g., price or “Add to Cart”)**

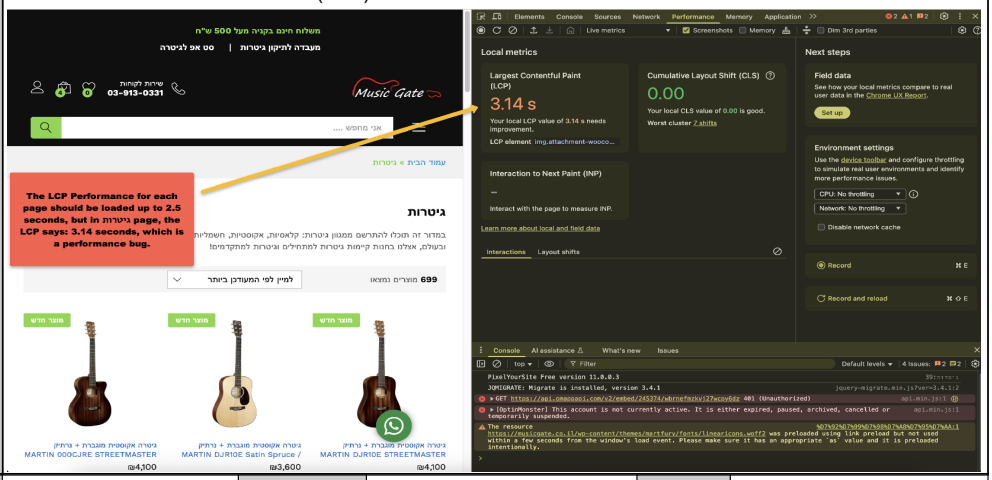
**Note:**
Product layout missing essential UI elements due to out-of-stock items or design gaps.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B61 – Product layout broken in Edge InPrivate for "כלים ומתאמים" | | | | |
| **Description:** | Missing layout elements on product page in private Edge session | | | | |
| **Status:** | Failed | **Project:** | Music Gate STR Project | | |
| **Attachment:** | Decided not to add an attachment for this UI issue. | | | | |
| **Severity:** | Minor | **Priority:** | Low | **Type:** | UI / Usability |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Microsoft Edge | **Version:** | 138.0.3351.83 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | All elements (title, image, price, description, CTA) should render on product page. | | | | |
| **Actual Result:** | Some key product elements missing due to website rendering issue, not browser mode. | | | | |

## Steps to Reproduce:

1. **Launch Microsoft Edge (InPrivate mode) on Mac**
2. **Navigate to** https://musicgate.co.il/
3. **Look for "כלים ומתאמים" in the homepage top menu.**
4. **Confirm its absence.**

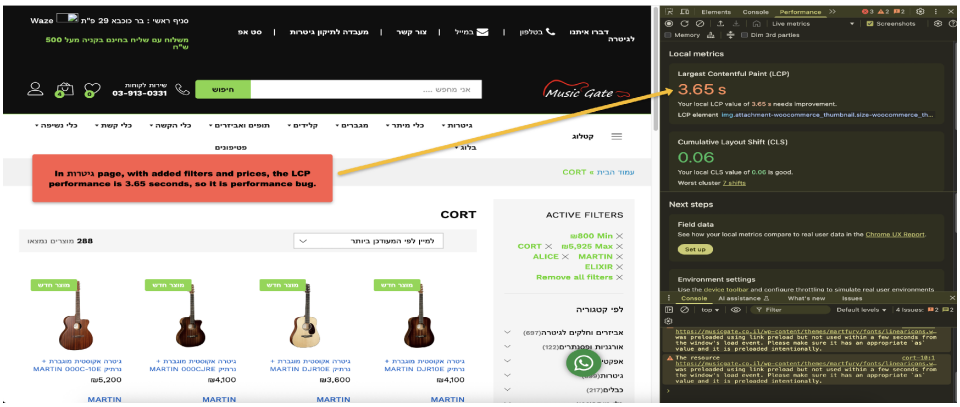
**Note:**
The issue seems to originate from the website itself, not the browser's private mode and also category may exist on site but is missing from homepage.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B164 – “Guitars” Category Page Loads Slowly | | | | |
| **Description:** | Guitars category takes over 5 seconds to load fully. | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** | Screenshot of the LCP for the "Guitars" page:<br>Included is a screenshot showing the DevTools performance panel and the page itself, highlighting the measured Load Contentful Paint (LCP) time of **3.14 seconds**.<br>  <br> | | | | |
| **Severity:** | Major | **Priority:** | Medium | **Type:** | Performance |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Page should load all products and UI components within 2.5 seconds. | | | | |
| **Actual Result:** | Full content appears after 3.1–5.4 seconds with minor image lag. | | | | |

## Steps to Reproduce:

1. **Open Chrome browser**
2. **Navigate to** https://musicgate.co.il/
3. **Click “גיטרות” from homepage navigation**
4. **Open DevTools → Network**
5. **Measure full load time of category page**

**Note:**
Delays occur despite fast and stable internet. Affects UX

| | | | |
|---|---|---|---|
| **Bug Name & Number:** | B165 – Filtered product results take too long to load | | |
| **Description:** | Applying multiple filters results in slow product updates | | |
| **Status:** | Open | **Project:** | Music Gate STR Project |
| **Attachment:** | Screenshot of the same page but with added filters and prices:<br>LCP performance is 3.65 seconds after filters are applied — screenshot includes filtered guitars page and Chrome DevTools (Network + Performance tab) showing slow rendering after applying filters (e.g., brand, price, rating).<br>  <br> | | |

| | | | | | |
|---|---|---|---|---|---|
| **Severity:** | Minor | **Priority:** | Medium | **Type:** | Performance |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | Filtered results should appear within 2 seconds after filters applied | | | | |
| **Actual Result:** | Page responds slowly after filters; update takes ~3–7 seconds | | | | |

## Steps to Reproduce:

1. **Open Chrome browser**
2. **Navigate to** https://musicgate.co.il/
3. **Choose a category (e.g., גיטרות)**
4. **Apply filters: Brand = Yamaha, Price = 1000–2000, Rating = 4+**
5. **Open DevTools → Network tab**
6. **Measure loading time after filters**
7. **Note the slow rendering of updated results**

## Note:

Filter delay persists despite stable internet and fast device. The page takes longer than expected to update filtered results, affecting user experience.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B166 – Accessibility navigation via keyboard is inconsistent | | | | |
| **Description:** | TAB key skips some elements on homepage during keyboard navigation | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** | You can watch the video that demonstrates the website is not accessible to people with physical disabilities:<br><br>https://www.youtube.com/watch?v=3f51JV9yVO0 | | | | |
| **Severity:** | Critical | **Priority:** | High | **Type:** | Accessibility |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | All homepage elements should be reachable using the TAB key | | | | |
| **Actual Result:** | Some elements are skipped and not accessible with the keyboard | | | | |

## Steps to Reproduce:

1. **Open Chrome browser**
2. **Navigate to https://musicgate.co.il/**
3. **Use TAB key for navigation**
4. **Note that some items (e.g., links, banners) are skipped**

**Note:**
Severity is Critical as this lacks of accessibility for users with physical disabilities.
*A supporting video was included to emphasize the real-world impact of the issue on users with physical disabilities.*

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | B168 – Checkout Flow Not Fully Accessible via Keyboard Navigation Only | | | | |
| **Description:** | Keyboard-only users can't complete checkout flow; some elements are unreachable or lack visual focus. | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** | Here's another video of an accessibility bug when I'm trying to fully navigate on the website:<br><br>https://www.youtube.com/watch?v=0KpAw6ZoC4A | | | | |
| **Severity:** | Critical | **Priority:** | High | **Type:** | Accessibility |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | All checkout steps should be accessible using Tab/Shift+Tab, with clear visual focus indicators. | | | | |
| **Actual Result:** | Tab navigation skips actions like "remove item" and "proceed to checkout" unexpectedly. | | | | |

## Steps to Reproduce:

**1. Open Chrome browser**

**2. Navigate to https://musicgate.co.il/**

**3. Add item to cart**

**4. Navigate to cart with Tab**

**5. Try removing item using Tab/Enter**

**6. Attempt full checkout flow without mouse**

**7. Observe unreachable or skipped steps**

**Note:**
*This issue is critical because keyboard-only users (e.g., users with physical disabilities) cannot complete the checkout process on their own due to skipped or inaccessible steps. A supporting video demonstrates the navigation failure clearly.*

# Console Introduction

**In modern web applications, many bugs are not visible through the user interface but occur behind the scenes in the browser's processing engine. The Developer Console, often referred to simply as the console, is a built-in browser tool that allows testers and developers to view real-time logs, JavaScript errors, network failures, and system warnings. It acts as a diagnostic window into the internal behavior of a website during runtime.**

**For QA testers, the console is a powerful tool that helps uncover hidden issues such as failed API calls, loading errors, deprecated functions, or security policy violations — all of which may go unnoticed during manual UI testing.**

### Accessing the Developer Console:

The console can be opened in most browsers by pressing F12 or using the shortcut Ctrl + Shift + I (on Windows) or Cmd + Option + I (on macOS). After opening the developer tools panel, select the **Console** tab to begin monitoring log messages and runtime activity.

**By including console analysis in QA workflows, testers gain better visibility into application performance and stability. Identifying and documenting console bugs allows for early detection of technical problems, clearer communication with developers, and ultimately, a smoother user experience.**

# Console Bugs

## What are Console Bugs and What will be shown in this section?

Console bugs are issues that appear in the browser's Developer Console but may not be visible on the screen or affect the UI directly. These bugs usually include JavaScript errors, failed API calls, CORS issues, security warnings, and missing resources (like images or files that failed to load).

Even though these bugs might not stop a user from using the site, they often point to deeper problems in the application that can affect performance, reliability, or future features.

### In this section, we will present two real examples of console bugs that were discovered during testing:

1. One bug found on the Homepage
2. One bug found on the Guitars Category Page

### Each bug report includes:

- A clear title and description
- Screenshot of the console output
- Severity and priority level
- Steps to reproduce the issue
- Expected vs. actual results
- A short technical note explaining why the bug matters

These examples highlight the importance of using the Developer Console as part of every QA process - not just for debugging, but for improving the quality and stability of the product.

There will be two console bugs that severity and priority of the console bug reports will be explained:

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | BC001 – Server Error 500 on Resource Load | | | | |
| **Description:** | The browser fails to load a resource from the server (`musicgate.co.il`), displaying a `500 Internal Server Error`. This indicates a server-side malfunction which may prevent full page functionality or loading of specific backend assets. | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** | Screenshot clearly showing the error message:<br>  <br> | | | | |
| **Severity:** | Major | **Priority:** | Medium | **Type:** | Server-side Bug |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | All server resources should load properly without throwing backend errors. | | | | |
| **Actual Result:** | At least one server request fails with status code 500, indicating an internal error. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to** https://musicgate.co.il/
3. **Open DevTools → Console tab**
4. **Reload the page**
5. **Observe the error message:** `Failed to load resource: the server responded with a status of 500 ()`

**Notes:**
A 500 error usually means that the server code or configuration failed to handle the request. Depending on which resource is affected (JS, API, etc.), this can cause serious functionality issues or make parts of the site unreliable.

## Let's break down Severity and Priority as they appear in the bug report (BC001):

### **Severity**: Major

### What it means:

Severity refers to the technical impact of the bug on the system. **It's usually determined by the QA team:**

1. **Major Severity** means the bug causes a serious issue — it may block resources, affect system stability, or interfere with important backend processes.

2. In BC001, the browser displays a 500 Internal Server Error when trying to load a resource from musicgate.co.il.

3. This indicates a server-side failure, which prevents proper communication between the client and server.

4. The error may break certain parts of the page or cause delays, even if the UI appears to load partially.

5. Since the issue is not cosmetic and can affect functionality, the severity is marked as Major.

### **Priority**: Medium

### What it means:

Priority refers to the urgency of fixing the bug**. It's usually determined by the product manager or the development team:**

1. **Medium Priority** means the bug should be fixed soon, preferably within an upcoming sprint, but it's not blocking the release.

2. In this case:

A. The error occurs on the **server-side**, outside the control of the frontend.

B. The bug **may impact reliability**, but doesn't completely crash the system or UI.

C. If the failed resource is not essential (e.g., analytics), the fix can wait; but if it's related to core data or scripts, priority may need to increase.

3. Therefore, the priority is set to **Medium**, pending deeper investigation of which resource is affected.

| | | | | | |
|---|---|---|---|---|---|
| **Bug Name & Number:** | BC002 – Console TypeError on LCP Property (Uncaught in Promise) | | | | |
| **Description:** | When the page loads, a `TypeError` is thrown in the browser console. The error originates from `wpr-beacon.js:435` and states:<br>`Cannot read properties of undefined (reading 'lcp')`<br>This indicates that the script attempts to read the `lcp` property from an object that is not properly initialized (undefined), and the error is unhandled within a Promise. | | | | |
| **Status:** | Open | **Project:** | Music Gate STR Project | | |
| **Attachment:** | Screenshot showing the console error:<br>`Uncaught (in promise) wpr-beacon.js:435`<br>`TypeError: Cannot read properties of undefined (reading 'lcp')`<br>`at y.init (wpr-beacon.js:435:65)` | | | | |
| **Severity:** | Minor | **Priority:** | Low | **Type:** | Console Bug |
| **Reporter:** | Yaeer Gratsyani | **Assignee:** | | | |
| **Browser:** | Google Chrome | **Version:** | 138.0.7204.101 | **Device:** | MacBook Pro M3 Max |
| **Expected Result:** | No unhandled errors should appear in the console during normal page load. | | | | |
| **Actual Result:** | A JavaScript error appears due to an unhandled promise that attempts to access an undefined `lcp` property. | | | | |

## Steps to Reproduce:

1. **Open Google Chrome**
2. **Navigate to** https://musicgate.co.il/
3. **Open DevTools → Console tab**
4. **Reload the page**
5. **Observe the error:**

```
Uncaught (in promise) TypeError: Cannot read properties of undefined (reading 'lcp')
```

**Notes:**

The error may relate to performance analytics that depend on the `lcp` metric before it's available. Although it does not directly affect user experience or UI behavior, it could interfere with internal monitoring or analytics tool

## Let's break down Severity and Priority as they appear in the bug report (BC002):

**Severity**: Minor

### What it means:

**Severity measures the technical impact on the system:**

1. The error is **limited to the console** and doesn't affect any user-visible functionality.

2. IIt occurs during a **background script execution** and relates to a missing `lcp` value.

3. Since the bug is silent for users and affects only internal metrics, it's classified as **Minor**.

**Priority**: Low

### What it means:

**Priority defines how soon the issue should be fixed:**

1. The bug does not block user flows or major features.
2. It is related to **non-critical scripts** (likely analytics or performance tracking).
3. It can be resolved in a **future maintenance release** with minimal urgency.

# Python & Selenium Introduction

## So, what is Python?

**Python** is a powerful, high-level programming language known for its readability and clean syntax.
It was created by **Guido van Rossum** in the **late 1980s** and officially released in **1991**. His goal was to design a language that would be easy to learn and intuitive to use — a language that felt almost like English.
The name *Python* wasn't inspired by the snake — but by the British comedy group **Monty Python**, reflecting van Rossum's desire to keep programming fun and not overly serious.

Over the decades, Python has become one of the **most widely-used languages in the world**. It powers a huge range of technologies, including:

- Web development (with frameworks like Django and Flask)
- Machine learning and AI (with TensorFlow, PyTorch, scikit-learn)
- Data analysis (with pandas and NumPy)
- Automation and scripting
- And of course, **QA and test automation**

Thanks to its simplicity, large ecosystem of libraries, and supportive community, **Python is often the first language new developers and testers learn — just like me.**

# So, what is Selenium?

**Selenium** is an open-source framework for automating web browsers.
It was first developed in **2004** by **Jason Huggins**, a test engineer at **ThoughtWorks**, who wanted a way to automate repetitive browser tasks while testing web applications.

Initially, it started as a simple tool called **Selenium Core**, which injected JavaScript into pages to simulate user actions.
As web applications became more complex, Selenium evolved into a more advanced and flexible toolset, including:

- **Selenium WebDriver** – the most popular part today, which lets you control browsers like Chrome, Firefox, Safari, and Edge using real programming languages like Python, Java, or C#
- **Selenium Grid** – for running tests in parallel on multiple machines
- **Selenium IDE** – a beginner-friendly record-and-playback tool (not commonly used in professional QA)

What made Selenium so popular is that it supports **real browser automation** — opening websites, clicking elements, filling out forms, navigating pages — just like a real user would.

Today, Selenium is considered the **industry standard for UI testing**, and it integrates with many **QA and CI/CD tools**.
It's supported by a huge global community and actively maintained by the **Selenium Project** under the **OpenJS Foundation**.

## Why I Chose Them?

In my project, I chose **Python** for its beginner-friendly nature and easy-to-read syntax.
It allowed me to focus on learning automation without getting overwhelmed by complex code.

I combined it with **Selenium WebDriver**, which gave me the ability to simulate real user interactions on websites — like clicking, typing, and navigating — all through code.

Even as someone new to programming, I was able to build working test scripts, understand how automation flows, and start thinking like a real **QA engineer**.

# Automated Tests with Python & Selenium

Automated testing means using code to simulate how a real user would interact with a website. In my project, I used Python because it's beginner-friendly and easy to understand, and I combined it with Selenium WebDriver, a tool that allows me to control a real browser. Together, they let me open web pages, click buttons, fill out forms, and check that everything works as expected. Each automated test verifies a specific part of the site, helping to catch bugs early without repeating the same manual steps. I wrote 5–7 automated test cases for the Music Gate website, focusing on important user flows. Even though I'm new to programming, I built and ran these tests independently while using ChatGPT for guidance.

To be honest, I'm more passionate about automation than manual testing, and this project gave me the motivation and confidence to keep growing in that direction.

# Loading the Music Gate Website

## Purpose:

This automated test script is designed to verify that the homepage of Music Gate loads successfully and includes expected keywords in the page title.

```python
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.support.ui import WebDriverWait
import logging

logging.basicConfig(level=logging.INFO, format='[%(levelname)s] %(message)s')

def setup_driver():
    service = Service(executable_path="/Users/yaeergratsyani/Desktop/chromedriver")
    options = Options()
    options.add_argument("--start-maximized")
    return webdriver.Chrome(service=service, options=options)

def validate_homepage_title(driver, url, keywords):
    driver.get(url)
    WebDriverWait(driver, 10).until(lambda d: d.title.strip() != "")
    title = driver.title.strip()
    logging.info(f"Page title is: {repr(title)}")
    for word in keywords:
        if word in title:
            logging.info(f"Title matched keyword: '{word}'")
            return
    raise AssertionError(f"Title '{title}' does not contain any of: {keywords}")

def run_test():
    driver = setup_driver()
    try:
        validate_homepage_title(driver, "https://musicgate.co.il", ["Music", "Gate", "מיוזיק"])
        logging.info("Homepage validation passed.")
    except AssertionError as ae:
        logging.error(f"Assertion failed: {ae}")
    except Exception as e:
        logging.error(f"Unexpected error: {e}")
    finally:
        driver.quit()
        logging.info("Browser closed.")

run_test()
```

## Result:

The test was executed using Selenium and passed successfully. The expected keyword ("Music Gate") was found in the title of the homepage.

```
Python 3.13.5 (v3.13.5:6cb20a219a8, Jun 11 2025, 12:23:45) [Clang 16.0.0 (clang-1600.0.26.6)] on darwin
Enter "help" below or click "Help" above for more information.
>>>
============ RESTART: /Users/yaeergratsyani/Desktop/homepage_test.py ===========
[INFO] Page title is: 'כלי נגינה - חנות מוזיקה אונליין - תזמינו באתר - Music Gate'
[INFO] Title matched keyword: 'Music'
[INFO] Homepage validation passed.
[INFO] Browser closed.
>>>
```

# Loading the Guitar Category Page

## Purpose:

This automated test script is designed to verify that the Guitar Category page of Music Gate loads successfully and includes expected keywords in the page title.

```python
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.chrome.options import Options
import logging
import time

logging.basicConfig(level=logging.INFO, format='[%(levelname)s] %(message)s')

GUITAR_CATEGORY_URL = "https://musicgate.co.il/product-category/%d7%92%d7%99%d7%98%d7%a8%d7%95%d7%aa"
EXPECTED_KEYWORDS = ["גיטרות", "Guitar"]

def setup_driver():
    service = Service(executable_path="/Users/yaeergratsyani/Desktop/chromedriver")
    options = Options()
    options.add_argument("--start-maximized")
    return webdriver.Chrome(service=service, options=options)

def validate_title(driver, expected_words):
    time.sleep(2)
    title = driver.title
    for word in expected_words:
        if word in title:
            logging.info(f"Title validated: '{title}' (matched: '{word}')")
            return
    raise AssertionError(f"Title '{title}' does not contain any of: {expected_words}")

def run_guitar_category_test():
    driver = setup_driver()
    try:
        logging.info(f"Opening page: {GUITAR_CATEGORY_URL}")
        driver.get(GUITAR_CATEGORY_URL)
        validate_title(driver, EXPECTED_KEYWORDS)
        logging.info("Guitar Category homepage loaded and title validated successfully.")
    except AssertionError as ae:
        logging.error(f"Assertion failed: {ae}")
    except Exception as e:
        logging.error(f"Unexpected error: {e}")
    finally:
        driver.quit()
        logging.info("Browser closed.")

run_guitar_category_test()
```

## Result:

The test was executed using Selenium and passed successfully. The expected keyword ("Guitar") was found in the title of the page.

```
Python 3.13.5 (v3.13.5:6cb20a219a8, Jun 11 2025, 12:23:45) [Clang 16.0.0 (clang-1600.0.26.6)] on darwin
Enter "help" below or click "Help" above for more information.
>>>
========= RESTART: /Users/yaeergratsyani/Desktop/guitarcategory_test.py ========
[INFO] Opening page: https://musicgate.co.il/product-category/%d7%92%d7%99%d7%98%d7%a8%d7%95%d7%aa
[INFO] Title validated: 'מגוון גיטרות שונות המובילות בארץ ובעולם במחירים לכל כיס – מיוזיק גייט' (matched: 'גיטרות')
[INFO] Guitar Category homepage loaded and title validated successfully.
[INFO] Browser closed.
>>>
```

# Loading the Guitar Product Page

## Purpose:

This automated test script is designed to verify that the Guitar Category page of Music Gate loads successfully and includes expected keywords in the page title.

```python
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.chrome.options import Options
import time
import logging

logging.basicConfig(level=logging.INFO, format='[%(levelname)s] %(message)s')

GUITAR_URL = "https://musicgate.co.il/product/%d7%92%d7%99%d7%98%d7%a8%d7%94-%d7%90%d7%a7%d7%95%d7%a1%d7%98%d7%99%d7%aa-%d7%a9%d7%97%d7%95%d7%a8%d7%94-%d7%a0%d7%a8%d7%aa%d7%99%d7%a7-martin-spdx1blk-dx1-black-hpl"
EXPECTED_KEYWORDS = [
    "MARTIN SPDX1BLK",
    "גיטרה אקוסטית",
    "מרטין שחורה"
]

def setup_driver():
    service = Service(executable_path="/Users/yaeergratsyani/Desktop/chromedriver")
    options = Options()
    options.add_argument("--start-maximized")
    return webdriver.Chrome(service=service, options=options)

def validate_page_title(driver, expected_keywords):
    time.sleep(2)
    title = driver.title
    for word in expected_keywords:
        if word in title:
            logging.info(f"Title validated: '{title}' (matched: '{word}')")
            return
    raise AssertionError(f"Title '{title}' does not contain any of: {expected_keywords}")

def run_guitar_test():
    driver = setup_driver()
    try:
        driver.get(GUITAR_URL)
        validate_page_title(driver, EXPECTED_KEYWORDS)
        logging.info("Guitar product page loaded successfully.")
    except AssertionError as ae:
        logging.error(f"Assertion failed: {ae}")
    finally:
        driver.quit()

run_guitar_test()
```

## Result:

The test was executed using Selenium and passed successfully. The expected keyword ("Guitar") was found in the title of the page.

```
Python 3.13.5 (v3.13.5:6cb20a219a8, Jun 11 2025, 12:23:45) [Clang 16.0.0 (clang-1600.0.26.6)] on darwin
Enter "help" below or click "Help" above for more information.
>>>
========= RESTART: /Users/yaeergratsyani/Desktop/specificguitar_test.py ========
[INFO] Title validated: 'גיטרה אקוסטית שחורה + נרתיק MARTIN SPDX1BLK DX1 BLACK HPL' (matched: 'MARTIN SPDX1BLK')
[INFO] Guitar product page loaded successfully.
>>>
```

# An Automation Search with Python & Selenium

## Purpose:

This automated test script is designed to verify that the Guitar Category page of Music Gate loads successfully and includes expected keywords in the page title.

```
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import logging

logging.basicConfig(level=logging.INFO, format='[%(levelname)s] %(message)s')

URL = "https://musicgate.co.il"
EXPECTED_TITLE_KEYWORDS = ["Music Gate", "מנגנת גיטרות"]
SEARCH_TERM = "גיטרה"
SEARCH_INPUT_SELECTOR = "input.search-field"

def setup_driver():
    service = Service(executable_path="/Users/yaeergratsyani/Desktop/chromedriver"
    options = Options()
    options.add_argument("--start-maximized")
    return webdriver.Chrome(service=service, options=options)

def open_homepage(driver):
    logging.info("Opening homepage...")
    driver.get(URL)
    WebDriverWait(driver, 10).until(EC.title_is(driver.title))
    page_title = driver.title
    if not any(word in page_title for word in EXPECTED_TITLE_KEYWORDS):
        raise AssertionError(f"Unexpected page title: {page_title}")
    logging.info("Homepage title validated.")

def perform_search(driver, term):
    logging.info("Performing search...")
    search_box = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.CSS_SELECTOR, SEARCH_INPUT_SELECTOR))
    )
    search_box.clear()
    search_box.send_keys(term)
    search_box.submit()

def validate_search_result(driver, term):
    WebDriverWait(driver, 10).until(
        EC.text_to_be_present_in_element((By.TAG_NAME, "body"), term)
    )
    page_source = driver.page_source
    if term not in page_source:
        raise AssertionError(f"Search term '{term}' not found in page source")
    logging.info("Search result validated.")

def run_test():
    driver = setup_driver()
    try:
        open_homepage(driver)
        perform_search(driver, SEARCH_TERM)
        validate_search_result(driver, SEARCH_TERM)
        logging.info("Search test passed successfully.")
    except AssertionError as ae:
        logging.error(f"Assertion failed: {ae}")
    except Exception as e:
        logging.error(f"Unexpected error: {e}")
    finally:
        driver.quit()
        logging.info("Browser closed.")

run_test()
```

## Result:

The test was executed using Selenium and passed successfully. The expected keyword ("Guitar") was found in the title of the page.

```
Python 3.13.5 (v3.13.5:6cb20a219a8, Jun 11 2025, 12:23:45) [Clang 16.0.0 (clang-1600.0.26.6)] on darwin
Enter "help" below or click "Help" above for more information.
>>>

========== RESTART: /Users/yaeergratsyani/Desktop/mgsearch_test.py ==========
[INFO] Opening homepage...
[INFO] Homepage title validated.
[INFO] Performing search...
[INFO] Search result validated.
[INFO] Search test passed successfully.
[INFO] Browser closed.
>>>
```

# Criteria Status: Manual & Automation

**This section presents a focused comparison between the manual and automated testing efforts throughout the project, explaining which functionalities were tested manually or automated - and why.**

During the automation phase, I developed and executed 3 basic-level and 1 intermediate-level test scenarios using Selenium. These focused on stable, high-priority flows that benefited from speed and consistency — such as navigation, product loading, and form submissions. Some scripts were upgraded during the project to improve reliability and coverage.

Manual testing was intentionally applied to areas with dynamic UI behavior, conditional visibility, or inconsistent rendering — where automation alone proved unreliable. One key scenario even involved a failed automated test that was later clarified through manual validation, confirming a genuine system-level bug.

The combination of both methods enabled broader coverage, practical efficiency, and realistic decision-making under real-world constraints.

**Despite the technical challenges**, this dual approach strengthened my confidence with Python, Selenium, and QA test design, while sharpening my ability to evaluate when automation is worthwhile versus when manual testing remains essential.

# Conclusions & Recommendations

## Conclusions:

Throughout this project, I gained hands-on experience in designing, implementing, and executing both manual and automated test cases using real-world scenarios on a live website. The use of Python and Selenium enabled me to simulate user behavior and validate system functionality effectively.

Although certain challenges were encountered — especially during the automation of dynamic web elements — each issue contributed to a deeper understanding of synchronization, DOM structure, and test stability. As part of the learning process, I iteratively upgraded all four automated tests to improve locator accuracy and robustness. One test initially failed due to an assertion error, but after manual investigation, the failure turned out to be a valid bug, showcasing how manual testing can complement automation in validating real issues.

The successful execution of 3 basic and 1 intermediate-level automated tests, combined with a wide range of manual validations, reflects a well-balanced and adaptive testing strategy. This project helped me strengthen my technical confidence, attention to detail, documentation practices, and test planning abilities.

## Recommendations:

**1. Expand automation coverage** for stable and repetitive test scenarios, especially those with high business value, to reduce manual effort and increase testing efficiency. Iterative improvement of locators and script reliability is essential.

**2. Implement robust error handling and logging** within automation scripts to better identify whether failures are due to script limitations or actual bugs, allowing for quicker triage and resolution.

**3. Leverage manual testing as a complement**, particularly when debugging failed test cases or validating dynamic UI behavior that might not be fully supported through automation. This hybrid approach proved valuable in uncovering a true bug that was initially suspected to be a script error.

**4. Continue skill development** in Selenium's advanced features (e.g., dynamic waits, page object model, test frameworks like `pytest`) to scale and maintain tests more effectively.

**5. Maintain clear and detailed documentation** for both manual and automated tests to ensure transparency, traceability, and smoother knowledge transfer across teams.

## Release Recommendation:

Based on the test results, I believe the website can be released **with caution**. While the core functionalities are stable and most critical flows operate as expected, one automated test initially failed due to a locator issue that disguised a real bug. Manual verification successfully uncovered the root cause, leading to both a code fix and a test script upgrade.

This highlights the importance of combining automation with selective manual investigation, especially when dynamic behavior is involved. Several areas—particularly in **error handling**, **UI responsiveness**, and **form validation**—still require improvements to ensure robust user experience across devices and edge cases.

A solid QA infrastructure must be in place to continue refining the product post-release. Further regression testing is strongly recommended to prevent future changes from reintroducing previously resolved bugs.

# Final Project Summary

**This project marked a significant milestone in my professional development as a QA tester. Over the course of several weeks, I planned, executed, and documented a comprehensive test strategy for a live website, combining both manual and automated testing approaches.**

I began by designing comprehensive manual test cases, eventually identifying and documenting 36 distinct bugs. These covered a wide range of categories — from UI/UX issues and broken links to missing validations and system errors. Each bug report was accompanied by detailed reproduction steps, expected results, and supporting evidence such as screenshots or videos.

Alongside the manual effort, I implemented automated tests using Python and Selenium, covering 3 basic-level scripts and 1 intermediate-level scenario. These tests verified critical flows such as website loading, category navigation, and contact form functionality. One automated test initially failed, which was suspected to be a script error. However, manual inspection revealed an actual bug in the system. This finding not only validated the automation process but also led to a targeted script enhancement, demonstrating the importance of cross-validation between manual and automated methods.

I faced challenges automating dynamic web elements — particularly those with delayed rendering, unstable locators, or conditional visibility. These required workarounds such as `WebDriverWait`, adjusted selectors, and repeated trial-and-error to ensure reliability.

Despite the technical hurdles, the project greatly expanded my hands-on experience in QA work. It strengthened my skills in test planning, synchronization, documentation, and debugging, while deepening my understanding of the tester mindset — balancing efficiency, curiosity, and persistence.

Ultimately, I emerged with stronger technical confidence and a deeper appreciation for blending structured analysis with flexible problem-solving. This project was not only a technical achievement, but also a personal step forward in professional growth and resilience.

# Project Closure Note

This document concludes a full-scale Quality Assurance project that integrates manual and automated testing practices into a cohesive and structured workflow.

Over the course of the project, I executed a comprehensive manual test plan, reported a total of 36 unique bugs, and developed 4 automated test cases using Python and Selenium — three at a basic level and one at an intermediate level. These scripts were designed to simulate real user behavior and validate key website flows such as search functionality, product selection, and contact form submission. During the project, all automated scripts were iteratively upgraded to improve reliability, element selection, and synchronization. One automation test that initially failed due to a suspected assertion error was manually investigated and revealed a genuine bug in the system — showcasing the practical synergy between manual and automated QA efforts.

Significant technical challenges were encountered along the way, particularly in the automation phase. Dynamic DOM elements, timing delays, incorrect locators, and interaction errors required detailed analysis, trial and error, and continuous debugging. These challenges, rather than slowing the process, became learning opportunities — helping build confidence in problem-solving and deeper understanding of both the front-end structure and the tools involved.

The testing process was supported by clear documentation: each manual test case was written with defined steps, expected results, and supporting media where necessary. The automated scripts were validated, and their limitations were acknowledged and addressed through code iterations and locator adjustments.

With over 80 pages of content, this project reflects real effort, attention to detail, and an ability to carry out a testing project from planning through execution to reflection. It demonstrates not only familiarity with QA principles and automation tools, but also persistence, adaptability, and structured thinking — all essential qualities for future QA roles in real-world environments.