

E2E Test Cases Document - Safe Mode

Pandashop.md E-commerce Platform

Document Information

- **Project:** Pandashop.md E2E Test Cases (Safe Mode)
 - **Version:** 2.0
 - **Date:** August 2025
 - **Author:** QA Team
 - **Status:** Ready for Implementation
 - **Framework:** Playwright + TypeScript + Cucumber/Gherkin
 - **Total E2E Cases:** 45
 - **Testing Mode:** SAFE MODE - No real orders, No SMS
-

Document Purpose

This document contains End-to-End test cases that check complete user journeys on Pandashop.md safely. These test cases validate business flows from a user perspective without creating real orders or using SMS services.

Safety Rules:

- Tests stop at order review (no real order creation)
 - Email-only registration (no SMS testing)
 - Form validation only (no real payments)
 - Safe test data only (@example.com emails)
-

Critical Path E2E Test Cases

Test Suite: TS-E2E-001 - Product Shopping Journey (Safe Mode)

TC-E2E-001: Complete Shopping Flow to Order Review

Priority: Critical

Test Type: End-to-End (Safe Mode)

User Story: US-001, US-004, US-008

Automation: High Priority

Estimated Time: 4-5 minutes

Test Goal

Check that a customer can complete the entire shopping process from product search to order review safely (without placing real orders).

Before Starting


- Website works at <https://pandashop.md>
- Test environment is stable
- Safe test data is ready

Test Data


typescript

```
const testData = {  
  searchTerm: "iPhone",  
  customerInfo: {  
    firstName: "Тест",  
    lastName: "Покупатель",  
    email: "test.buyer@example.com",  
    phone: "+37360123456"  
  },  
  deliveryAddress: {  
    city: "Кишинев",  
    street: "ул. Штефан чел Маре",  
    building: "124",  
    apartment: "15",  
    postalCode: "MD-2012"  
  }  
};
```

Test Steps

Step	Action	Expected Result	Automation Help
1	Go to homepage	Pandashop homepage loads	<code>page.goto('https://pandashop.md')</code>
2	Search for "iPhone"	Search box accepts input	<code>page.fill('[placeholder*="Поиск"]', 'iPhone')</code>
3	Click search button	Search results show iPhone products	<code>page.click('button[type="submit"]')</code>
4	Click on first product	Product detail page opens	<code>.product-item:first-child</code>
5	Check product info	All details visible (price, cashback, stock)	<code>.product-details</code>
6	Click "В корзину"	Product added to cart, counter updates	<code>button:has-text("В корзину")</code>
7	Go to cart	Cart page shows added product	Cart navigation
8	Check cart contents	Product appears with correct details	<code>.cart-item</code>
9	Click "Оформить заказ"	Checkout process starts	<code>button:has-text("Оформить заказ")</code>
10	Fill customer info	Form accepts all required data	Customer form fields
11	Fill delivery address	Address form validation passes	Delivery address fields
12	Choose delivery method	Delivery options show with costs	Delivery method selection
13	Choose payment method	Payment options available	Payment method selection
14	Review order details	All order information correct	<code>.order-review</code>
15	STOP HERE	Order review complete - DO NOT SUBMIT	 SAFE MODE END

Expected Results

- Shopping flow works completely up to order review
- All product information displays correctly
- Cart calculations are accurate
- Forms validate input properly
- Order review shows all details correctly
-  **NO REAL ORDER IS CREATED**

Gherkin Scenario

gherkin

Feature: Complete Product Shopping Journey (Safe Mode)

@critical @smoke @safe

Scenario: Customer completes shopping flow to order review

Given I am on Pandashop homepage

When I search for "iPhone"

And I select the first product

And I add it to cart

And I go to cart page

And I proceed to checkout

And I fill all delivery information

And I select delivery and payment methods

And I review my order

Then I should see complete order summary

And all details should be correct

But I should NOT place a real order

Pass Criteria

- All 15 steps work successfully
- Order review shows correct information
- No real order is created
- No errors in browser console
- Page loads in under 3 seconds each

TC-E2E-002: Email Registration Flow (Safe Mode)

Priority: High

Test Type: End-to-End (Safe Mode)

User Story: US-005

Automation: High Priority

Estimated Time: 3 minutes

Test Goal

Check that a new user can register using email only (no SMS verification needed).

Before Starting

- User is not logged in

- Registration page is accessible
- **No SMS services active**

Test Data

typescript

```
const registrationData = {
  email: () => `test.user.${Date.now()}@example.com`,
  password: "SecurePass123!",
  confirmPassword: "SecurePass123!",
  firstName: "Новый",
  lastName: "Пользователь",
  phone: "+37368234567"
};
```

Test Steps

Step	Action	Expected Result	Safety Check
1	Go to homepage	Homepage loads correctly	Standard page load
2	Click "Регистрация"	Registration form appears	Registration page opens
3	Fill email field	Email input accepts valid format	Email validation works
4	Fill password fields	Password validation works	Strong password required
5	Fill name fields	Name fields accept text	Form fields work
6	Fill phone field	Phone format validation works	+373 format only
7	Check terms checkbox	Checkbox can be selected	Agreement required
8	Click "Зарегистрироваться"	Account created successfully	No SMS verification
9	Check welcome message	Success message appears	Registration confirmed
10	Check auto-login	User logged in automatically	Login state active
11	Check welcome bonus	100 lei bonus appears	Bonus system works

Expected Results

- Registration works with email only
- **No SMS verification required**
- User automatically logged in
- Welcome bonus appears in account
- All form validation works correctly

Gherkin Scenario

gherkin

Feature: Safe Email Registration

@registration @safe @no-sms

Scenario: New user registers with email only

Given I am on the homepage

When I click "Регистрация"

And I fill registration form with valid email data

And I submit the form

Then my account should be created immediately

And I should be logged in automatically

And I should receive 100 lei welcome bonus

But I should NOT need SMS verification

TC-E2E-003: Language Switching with Cart Preservation

Priority: High

Test Type: End-to-End

User Story: US-007, US-004

Automation: High Priority

Estimated Time: 3 minutes

Test Goal

Check that switching between Russian and Romanian keeps cart items and works properly.

Before Starting

- Website loads in Russian by default
- Multiple products available
- Cart functionality works

Test Data

typescript

```
const languageTest = {
  products: ["Smartphone", "Laptop", "Headphones"],
  languages: ["ru", "ro"],
  expectedTranslations: {
    "В корзину": "În coș",
    "Корзина": "Coș",
    "Оформить заказ": "Plasați comanda"
  }
};
```

Test Steps

Step	Action	Expected Result	Language Check
1	Check site language	Interface shows in Russian	Russian is default
2	Add 3 products to cart	Cart counter shows 3 items	Cart works in Russian
3	Go to cart page	All 3 products visible with Russian names	Cart content correct
4	Note prices and totals	Record all amounts in Russian	Amount tracking
5	Click "Ro" language switch	Interface changes to Romanian	Language switch works
6	Check page content	All text elements translated	Romanian translation
7	Go to cart page	Cart still has 3 products	Cart preserved
8	Check cart totals	Amounts and quantities same	Calculations preserved
9	Check product names	Names shown in Romanian	Product translation
10	Switch back to "Py"	Interface returns to Russian	Reverse switch works
11	Final cart check	All items and totals still there	Complete preservation

Expected Results

- Cart items stay during language switches
- All interface elements translate properly
- Prices and calculations stay identical
- No data is lost or corrupted
- User can continue shopping normally

Gherkin Scenario

gherkin

Feature: Language Switching with Cart Preservation

@multilingual @cart @critical

Scenario: Language switching preserves cart state

Given I have 3 items in cart in Russian

When I switch language to Romanian

Then cart should keep all items

And interface should be in Romanian

And prices should stay the same

When I switch back to Russian

Then everything should still be preserved

Mobile Experience E2E Test Cases

TC-E2E-004: Mobile Shopping with Click-to-Call

Priority: High

Test Type: End-to-End Mobile

User Story: US-009

Automation: High Priority

Estimated Time: 4 minutes

Test Goal

Check that complete shopping works on mobile devices with proper responsive design and phone call features.

Before Starting

- Test runs on mobile device or mobile simulation
- Phone click simulation available

Test Data

typescript


```
const mobileTestData = {  
  device: "iPhone 13",  
  viewport: { width: 375, height: 812 },  
  phoneNumbers: [  
    "022 815-819",  
    "079 815-819",  
    "060 815-819"  
  ]  
};
```

Test Steps

Step	Action	Expected Result	Mobile Features
1	Open site on mobile	Responsive layout loads correctly	Mobile optimization
2	Test hamburger menu	Menu expands with all categories	Touch navigation
3	Search for product	Mobile search works smoothly	Mobile search
4	Browse product details	Images zoom, text readable	Mobile product view
5	Test phone number clicks	Phone dialer opens for each number	Click-to-call works
6	Add product to cart	Mobile cart works well	Touch-friendly buttons
7	Go through mobile checkout	Forms work well on mobile	Mobile form optimization
8	Fill forms on mobile	Right keyboards appear	Mobile input help
9	Complete to order review	Review works without computer	Mobile checkout complete

Expected Results

- All features work on mobile device
- Touch interactions respond well
- Text readable without zooming
- Forms work well for mobile
- Phone numbers start phone calls

Gherkin Scenario

gherkin

Feature: Mobile Shopping Experience

@mobile @click-to-call @responsive

Scenario: Complete mobile shopping flow

Given I am using a mobile device

When I browse products on mobile

And I tap phone numbers to start calls

And I add products to cart

And I complete checkout forms on mobile

Then all features should work perfectly

And phone calls should start correctly

Form Validation E2E Test Cases

TC-E2E-005: Checkout Form Validation (Safe Mode)

Priority: Medium

Test Type: End-to-End Form Validation

User Story: US-008

Automation: Medium Priority

Estimated Time: 3 minutes

Test Goal

Check that all checkout forms validate input correctly and show proper error messages.

Test Data

typescript

```
const formTestData = {
  invalidEmail: "not-an-email",
  invalidPhone: "123",
  emptyFields: "",
  validData: {
    email: "test.form@example.com",
    phone: "+37360123456",
    city: "Кишинев"
  }
};
```

Test Steps

Step	Action	Expected Result	Validation Check
1	Start checkout with cart items	Checkout form loads	Forms available
2	Try to submit empty form	Error messages appear	Required field validation
3	Enter invalid email	Email error shows	Email format validation
4	Enter invalid phone	Phone error shows	Phone format validation
5	Fill all fields correctly	Errors disappear	Validation clears
6	Continue to next step	Form accepts valid data	Progress works
7	Test each required field	All validations work	Complete checking

Expected Results

- All required fields show errors when empty
- Email and phone format validation works
- Error messages clear when fixed
- Forms only accept valid data
- Error messages in correct language

Cross-Browser E2E Test Cases

TC-E2E-006: Cross-Browser Shopping Flow

Priority: Medium

Test Type: End-to-End Compatibility

User Story: Multiple

Automation: Medium Priority

Estimated Time: 20 minutes (across 4 browsers)

Test Goal

Make sure core shopping features work the same across different browsers.

Browser Coverage

- Google Chrome (latest)
- Mozilla Firefox (latest)
- Safari (latest)
- Microsoft Edge (latest)

Test Matrix

Browser	Shopping Flow	Language Switch	Mobile View	Form Validation
Chrome	✓	✓	✓	✓
Firefox	✓	✓	✓	✓
Safari	✓	✓	✓	✓
Edge	✓	✓	✓	✓

Automation Code Example

```
typescript
const browsers = ['chromium', 'firefox', 'webkit'];
browsers.forEach(browserName => {
  test.describe(`${browserName} - Core Shopping`, () => {
    test('Complete shopping flow works', async ({ page }) => {
      // Test the main shopping flow in each browser
    });
  });
});
```

Performance E2E Test Cases

TC-E2E-007: Page Load Performance Check

Priority: Medium
Test Type: End-to-End Performance
User Story: Multiple
Automation: Medium Priority
Estimated Time: 5 minutes

Test Goal

Check that important pages load fast enough for good user experience.

Performance Targets

Page Type	Target Load Time	Maximum OK
Homepage	< 2 seconds	3 seconds
Category Pages	< 2.5 seconds	3.5 seconds
Product Details	< 2.5 seconds	3.5 seconds
Cart Page	< 2 seconds	3 seconds
Checkout	< 3 seconds	4 seconds

Test Steps

Step	Action	Performance Check	OK Criteria
1	Load homepage	Measure load time	< 2 seconds
2	Go to category	Time to interactive	< 2.5 seconds
3	Open product details	Page load complete	< 2.5 seconds
4	Add to cart and view	Cart update time	< 1 second
5	Start checkout	Checkout load time	< 3 seconds

Automation Example

```
typescript

test('Page load performance', async ({ page }) => {
  const startTime = Date.now();
  await page.goto('https://pandashop.md');
  await page.waitForLoadState('networkidle');
  const loadTime = Date.now() - startTime;
  expect(loadTime).toBeLessThan(3000);
});
```

Error Handling E2E Test Cases

TC-E2E-008: Network Error Recovery

Priority: Low

Test Type: End-to-End Error Handling

User Story: Multiple

Automation: Low Priority

Estimated Time: 3 minutes

Test Goal

Check that the website handles network problems well and gives helpful error messages.

Error Scenarios

Scenario	Error Type	Expected Behavior	Recovery Action
Slow Connection	Network Delay	Loading indicator shows	User can wait or retry
Connection Lost	Network Failure	Error message appears	Retry button available
Server Error	500 Error	Friendly error message	Suggest trying again later
Page Not Found	404 Error	Clear "not found" message	Navigation to homepage

Expected Results

- Users get clear, helpful error messages
- Error messages not technical or scary
- Users can recover from errors easily
- Website stays usable during problems

E2E Test Suite Summary

Test Coverage Overview

Test Category	Critical	High	Medium	Total E2E
Shopping Flow	1	0	0	1
User Registration	0	1	0	1
Language Features	0	1	0	1
Mobile Experience	0	1	0	1
Form Validation	0	0	1	1
Cross-Browser	0	0	1	1
Performance	0	0	1	1
Error Handling	0	0	1	1
TOTAL	1	3	4	8

Automation Priority

Priority	Count	When to Automate	How Often to Run
Critical	1	First - Most Important	Every code change
High	3	Second - Very Important	Every day
Medium	4	Third - Important	Every week

Test Execution Strategy

Smoke Test Suite (Critical + High Priority)

- TC-E2E-001: Complete Shopping Flow (Safe Mode)
- TC-E2E-002: Email Registration (No SMS)
- TC-E2E-003: Language Switching
- TC-E2E-004: Mobile Shopping

Time: 12-15 minutes

When: Every time code changes

Full E2E Suite (All Tests)

- All 8 E2E test cases
- Cross-browser testing
- Performance checking

Time: 45-60 minutes

When: Every week or before big releases

Safe Mode Implementation

Safety Features Built In

- **Order Prevention:** All tests stop at order review
- **No SMS Testing:** Email registration only
- **Safe Test Data:** Only @example.com emails
- **Form Validation Only:** No real payments processed

Safety Checking

```
typescript
```

```
export class SafeTestUtils {
  static async ensureNoOrderPlacement(page: Page) {
    // Make sure no real orders can be created
    await page.addInitScript() => {
      // Block order submission forms
    };
  }

  static generateSafeTestEmail(): string {
    return `test.safe.${Date.now()}@example.com`;
  }
}
```

Test Data Management

Safe Test Data

- **Email Addresses:** Only @example.com domain
- **Phone Numbers:** Valid Moldova format (+373) for validation only
- **Addresses:** Real Moldova locations for form testing
- **Products:** Use existing catalog without changes

Data Cleanup

- Clear test carts after each test
- Remove test user accounts if created
- No real data affected by tests

Reporting and Monitoring

Test Reports Include

- Screenshots on test failures
- Video recordings of test runs
- Performance timing information
- Cross-browser comparison results
- Safety validation confirmation

Failure Analysis

- Detailed error logs for debugging

- Network request monitoring
 - Console error checking
 - Page performance metrics
-

Implementation Notes for GitHub Copilot

Framework Setup

- Playwright + TypeScript for reliable testing
- Cucumber/Gherkin for readable test scenarios
- Page Object Model for maintainable code
- Safe mode utilities to prevent real orders

Best Practices

- Clear test data separation
- Comprehensive error handling
- Cross-browser compatibility
- Mobile-first responsive testing
- Performance monitoring built in

Safety First

- All tests designed to be completely safe
 - No risk of real orders or SMS charges
 - Educational value without business impact
 - Perfect for learning and practice
-

Document Approval

Role	Name	Date
QA Lead		
Safety Officer		
Automation Engineer		

This document provides comprehensive E2E test specifications for safe testing of Pandashop.md using simple B2-level English.