

Step IT Academy s.r.o. Opletalova 1418/23, 110 00 Nové Město

Diploma Work

Functional and load testing of the website of the railway company
CD.cz

Автор: Olena Malinovska

Руководитель: Vitaliy Zamirovskiy

2025

Contents

1. Relevance of the study
2. Purpose and functions of the programme under test
3. Architecture of the system under test
4. Purpose and objectives of the thesis
5. Test environment
6. Methodology
7. Test cases and test scenarios
8. Testing tools used
9. Checklist
 - Functionality
 - Usability
 - Performance
 - Security
 - Content
10. Practical part of the work.
11. Conclusion

1. Relevance of the Study

In everyday life, the digitalization of transportation services has become a key direction in infrastructure development. Railway transport is actively integrating online services that allow passengers to plan routes, purchase tickets, and obtain travel information at any time. The reliability and correct functioning of such services directly affect customer satisfaction and the company's image.

The Czech railway network connects to neighboring countries such as Austria, Germany, Poland, and Slovakia, among others. The website **CD.cz** is the official online platform of České dráhy (Czech Railways) and provides functions that are critically important for users: route planning, displaying schedules, ticket price calculation, and issuing travel documents. Errors in these functions can lead to inconvenience, time loss, or financial damage for passengers.

Conducting **functional and load testing** of the CD.cz website allows for an assessment of its stability, accuracy of calculations, and ability to operate under high user activity. This is especially relevant during seasonal peaks, timetable changes, and other critical situations when the service must remain accessible and function correctly.

2. Purpose and Features of the Tested Application

The **CD.cz** website is intended to provide passengers of České dráhy with up-to-date travel information, as well as tools for independently planning routes and purchasing tickets. The main functions of the website include:

- Searching for routes between stations, taking into account departure date and time;
- Displaying train schedules;
- Calculating ticket prices based on the route, passenger category, train type, and service class;
- Online ticket purchasing and reservation;
- Providing information about promotions, discounts, and special offers;
- Enabling communication with technical support and submitting requests via a contact form;
- A **Fanshop** section — an online store offering branded ČD products and souvenirs.

These features are a vital part of the company's digital infrastructure and require regular verification for correctness and stability.

3. Architecture of the Tested System

It is important to note that the **CD.cz** web application is available in Czech and English, ensuring convenience for both local and international users. The mobile application supports three languages: Czech, English, and German, which broadens its accessibility for international passengers.

To provide a visual representation of the site and mobile application interfaces, relevant screenshots are included.

Figure 1 — Interface of the CD.cz mobile application

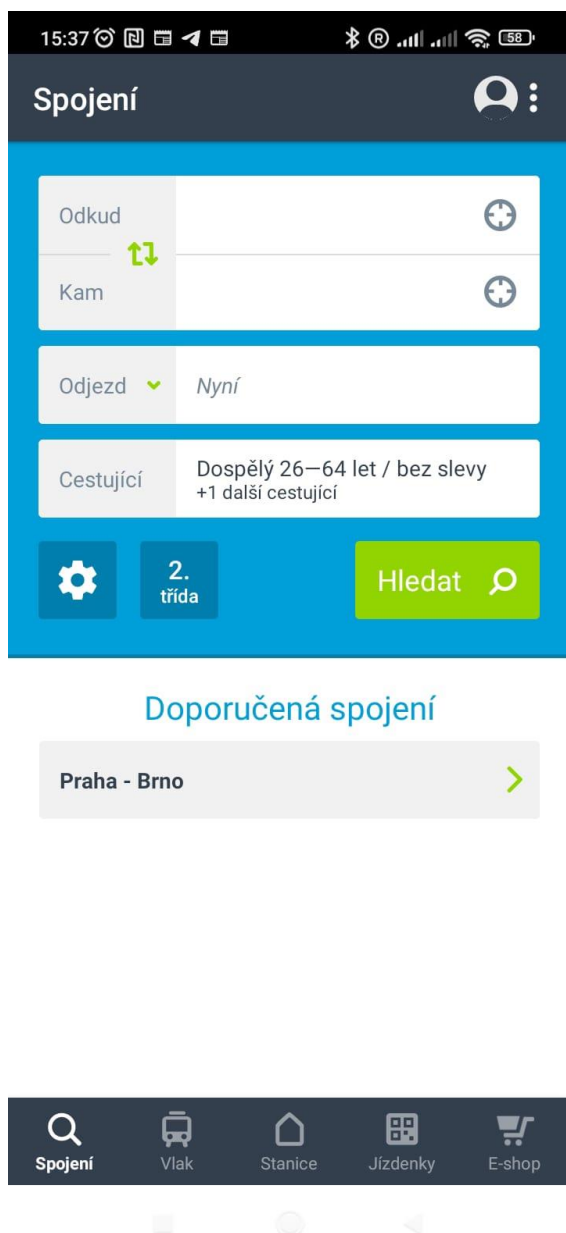
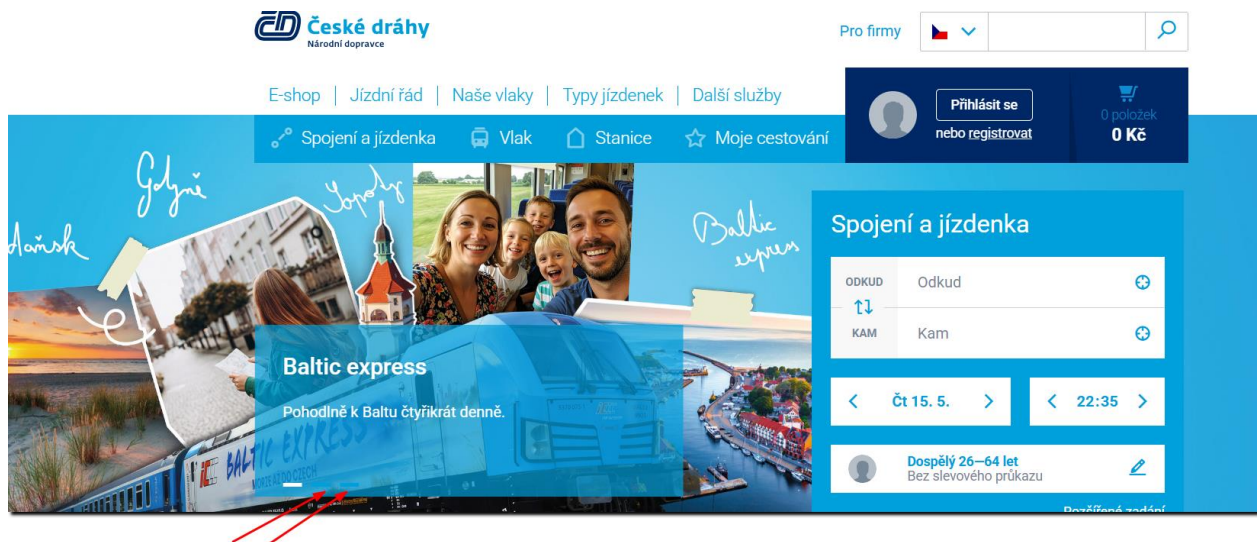


Figure 2 — Homepage of the CD.cz website



In addition to the website, České dráhy also offers an official mobile app, available for Android and iOS platforms. The mobile app provides the same core functions as the website: route searching, ticket purchasing, schedule display, and access to a personal user account. The application is designed for use on the go and allows electronic tickets to be stored without the need for printing.

The **CD.cz** website is a multi-level web system with both client-side and server-side components. The front end is developed using HTML, CSS, JavaScript, and modern frameworks to enable dynamic user interaction. The back end processes routes, fares, and bookings using proprietary algorithms and databases.

Key architectural components include:

- **User Interface (Frontend):** route search forms, ticket selection and order forms, as well as feedback forms;
- **Application Server (Backend):** logic for processing routes, calculating prices, checking seat availability, handling form submissions;
- **Database:** storage of information about routes, schedules, prices, customers, and booking operations;
- **Third-Party Services:** payment systems, user authentication, APIs for real-time data and weather display.

This architecture requires comprehensive testing of both individual components and their interaction under load and real user activity.

4. Objective and Tasks of the Thesis

The objective of this thesis is to conduct **functional and load testing** of the website of the Czech railway company **CD.cz**. During the course of this work, the following tasks were formulated:

- Verify the correctness of route generation;
- Evaluate the website's performance under high load conditions;
- Test the ticket price calculation functionality under various input parameters.

5. Testing Environment

The testing was carried out in an environment closely resembling real user conditions, with the following configurations:

- **Operating systems:** Windows 10 PRO 22H2
- **Browsers:** Google Chrome (versions 123–124), Mozilla Firefox (version 124), Microsoft Edge (136.0.3240.76)
- **Screen resolution:** 1920x1080 (Full HD)
- **Devices:** laptop **DESKTOP-NLC59KB**, smartphone **Redmi 9C NFC** with Android 11 (RP1A.200720.011)
- **Network connection:** wired 100 Mbps, Wi-Fi

To test the mobile version of the website, Chrome DevTools emulator and real Android devices were used. Automated tests were executed in the Chrome browser using the latest version of **ChromeDriver**.

6. Methodology

A combined methodology was applied during the testing process, incorporating both **manual and automated testing** approaches. For manual testing, checklists and test cases based on user scenarios and core site functions were used. For automated testing, the **Selenium** library in **Python** was utilized, enabling the simulation of user actions and verification of system stability during operations.

Testing of forms, route generation, and ticket price calculation was conducted using various input parameters such as trip type, date, passenger age, and status. The tests

were developed according to the principles of **positive and negative testing**, including boundary value analysis and input of invalid data.

To test form submissions, the **Postman** tool was used to emulate HTTP requests, thereby complementing UI testing with server-side validation checks. Test results were documented, and in case of discrepancies, **bug reports** were created and recorded.

7. Test Cases and Test Scenarios

To systematically verify the key functionalities of the **CD.cz** website, test cases were developed. Each test case described input data, expected results, and actual system behavior. The main focus was placed on the following categories:

- Route generation between stations;
- Ticket price calculation based on parameters (date, passenger, direction);
- Contact form submission;
- System behavior when invalid data is entered;
- Verification of discounts (e.g., for children, students);
- Functionality of the navigation menu and internal links.

The test cases were structured in tables and included a title, reproduction steps, test data, expected results, pass/fail status, and comments. Both **positive and negative scenarios** were covered, including **boundary values**. Examples of the test cases are provided in the appendix.

8. Testing Tools Used

- **Jira** – a project management and bug tracking system used for maintaining bug reports, tracking test statuses, and logging defects found during testing.
- **JMeter** – a load testing tool used to assess website performance under high user activity. It allows the creation of scenarios that simulate multiple simultaneous requests and helps analyze server behavior under stress.
- **Lighthouse** – a tool by Google for evaluating the quality of web pages, including performance, accessibility, SEO, and best practices. It was used to analyze page load speed and detect potential frontend bottlenecks.
- **Selenium WebDriver** – a browser automation library used to write automated tests in Python. It enabled the simulation of user actions on the

website, including entering routes, selecting dates, calculating prices, and submitting forms.

- **Google Chrome DevTools** – built-in developer tools used to analyze network requests, DOM structure, and debug the site's interface.
- **Manual testing tools** – checklists, spreadsheets, and scenario tables used for step-by-step manual verification of functionality.
- **Postman** – one of the most popular tools for testing APIs. It was used to send requests and validate server responses.
- **TestRail** – a professional platform for organizing, planning, and documenting software tests.
- **SSL Labs** – a tool for analyzing the security of HTTPS connections.
- **SecurityHeaders.com** – an online tool for checking the presence and proper configuration of HTTP security headers.
- **GitHub** – a code hosting platform.
- **BrowserStack** – a platform for cross-browser and real-device testing.

9. Checklist

A **website testing checklist** is a tool that helps structure and organize the verification process for a web resource. It helps avoid missing critical aspects of testing and ensures a higher-quality result. Using a checklist is especially beneficial for beginners like myself, as it provides a structured workflow and prevents overlooking key areas.

Website testing includes many aspects, ranging from functionality to security. Each of these must be carefully reviewed to ensure the best possible user experience. A checklist not only helps organize the testing process but also makes it more transparent and manageable. This is particularly important for teams working on large projects, where coordination and clarity of actions are essential.

Key Elements of My Checklist:

Creating an effective checklist requires understanding the core categories that must be tested. Here are the key areas I focused on:

Functionality

The functionality of a website refers to its core ability to perform its declared operations. Functional testing includes multiple aspects such as:

- **Navigation:** Checking all site links and menus. Ensuring that all links work correctly and lead to the intended pages. Navigation should be intuitive and user-friendly.
- **Forms:** Testing all forms for correct data entry and submission. Verifying that all fields work properly and data is sent correctly. This includes checking required fields, input formats, and error messages.
- **Buttons and Links:** Verifying that all buttons and links function as expected. Ensuring that buttons perform their designated actions and links lead to the correct destinations. This is especially critical for buttons related to purchases or form submissions.

Usability

Usability refers to how convenient and intuitive a website is for users. A well-designed website should be easy to navigate and interact with. The following points were included in the checklist:

- **Interface:** Assessing how easy the site is to use. Checking how easily users can find the information they need and perform key actions.
- **Accessibility:** Verifying that content is accessible to all users, including people with disabilities. This includes checking compatibility with screen readers and other assistive technologies.
- **Mobile Version:** Testing the site's responsiveness on various devices. It's essential to ensure that the site displays and functions correctly on smartphones and tablets.

Performance

Website performance refers to its ability to handle user requests quickly and efficiently.

- **Page Load Speed:** Measuring how quickly website pages load. Fast loading times are important for retaining users and improving SEO.
- **Stress Testing:** Checking the website's behavior under high load. Ensuring that the site can handle a large number of simultaneous users without crashing.

Security

Website security refers to its ability to protect user data and prevent attacks. The following items were included in the checklist:

- **SSL Certificate:** Verifying the presence and correct configuration of the SSL certificate. Ensuring that all pages are served over a secure connection.
- **Data Protection:** Evaluating how user data is secured. Checking whether data is stored and transmitted securely, and whether the site is protected against common attacks such as **SQL injection** and **XSS**.

Content

Website content includes all textual and multimedia elements. The following items were included in the checklist:

- **Spelling and Grammar:** Reviewing the text for errors. Ensuring that all content is written correctly and professionally.
- **Media Files:** Testing the loading and display of images and videos. Verifying that all media content loads properly and is displayed correctly across devices.

10. Practical Part of the Work

Functional Testing of Key Scenarios

Functional testing was conducted to verify the following key scenarios:

- Entering departure and arrival locations;
- Selecting the travel date;
- Displaying available routes;
- Displaying and calculating ticket prices;
- Submitting feedback via the contact form.

Tests were performed using both **manual** and **automated** methods (via the Selenium library).

Functional Testing of the Authorization Form

One of the key elements of the website is the login form, which provides access to the user's personal account. As part of functional testing, the correctness of the login form was tested using both valid and invalid data.

Testing Goal:

To verify correct form behavior in different input scenarios:

- With valid credentials;
- With incorrect username and/or password.

Test Case: Correct Login

TestRail Česká dráhy

Add Search Working On Olena lion Admin

České dráhy

Project Overview To-Do Test Cases Overview Details Tests & Results Defects History

Test Runs & Results Milestones Reports

In section Česká dráhy.

People & Status Unassigned Change

Created Olena lion

C6 Корректный логин

České dráhy

Type	Other	Priority	High	Assigned To	None	Estimate	None
References	None	Automation Type	None				

Preconditions

Selenium WebDriver на языке Python, а также браузер Google Chrome

Steps

1. Ввод действующего логина и пароля.
2. Нажатие кнопки входа.

Expected Result

Пользователь перенаправляется в личный кабинет.
После входа открылась страница профиля, подтверждающая успешную авторизацию.

Automation Code

```
import pytest
import time
from selenium import webdriver
from selenium.webdriver.common.by import By
```

```
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.chrome.service import Service
from selenium.webdriver import ActionChains
from webdriver_manager.chrome import ChromeDriverManager
from config import URL, correct_login, correct_password
options = webdriver.ChromeOptions()
options.add_argument("--disable-blink-features=AutomationControlled")
options.add_argument("--window-size=1920,1080")

driver =
webdriver.Chrome(service=Service(ChromeDriverManager().install()),
, options=options)
wait = WebDriverWait(driver, 10)

def test_correct_login():
    try:
        driver.get(URL)

        wait.until(EC.element_to_be_clickable((By.ID,
"consentBtnall"))).click()
        wait.until(EC.invisibility_of_element_located((By.ID,
"consentBtnall")))

        login_buttons = wait.until(
            EC.presence_of_all_elements_located((By.XPATH,
"//button[contains(@class,'userbox--toggle')]"))
        )

        visible_login_button = next((b for b in login_buttons if
b.is_displayed()), None)

        if visible_login_button:
            actions = ActionChains(driver)

actions.move_to_element(visible_login_button).perform()
            print("Навели на видимую кнопку входа")
```

```
driver.execute_script("arguments[0].scrollIntoView({block:
'center'});", visible_login_button)
    visible_login_button.click()
    print("Кликнули на Přihlásit se")
    time.sleep(3)

    email_field =
wait.until(EC.visibility_of_element_located((By.ID, "emailId")))
    password_field =
wait.until(EC.element_to_be_clickable((By.ID, "passwordId")))

    email_field.clear()
    email_field.send_keys(correct_login)
    print("Ввели email")

    password_field.clear()
    password_field.send_keys(correct_password)
    print("Ввели правильный пароль")

    submit_buttons =
wait.until(EC.presence_of_all_elements_located(
        (By.CSS_SELECTOR, "button.btn.btn--blue.btn--
filled.btn--full-width")
    ))

    submit_btn = next((b for b in submit_buttons if
b.is_displayed()), None)

    if submit_btn:

driver.execute_script("arguments[0].scrollIntoView({block:
'center'});", submit_btn)
        driver.execute_script("arguments[0].click();",
submit_btn)

        print(" Клик по кнопке 'Přihlásit' выполнен")
        time.sleep(3)
    else:
        print(" Кнопка отправки входа не найдена!")
```

```
except Exception as e:
    print(f" Ошибка во время выполнения теста: {e}")
finally:
    driver.quit()
```

```
✓ Tests passed: 1 of 1 test – 12 sec 297 ms

correct_login.py::test_correct_login PASSED [100%]Навели на видимую кнопку входа
Кликнули на Přihlásit se
Ввели email
Ввели правильный пароль
Клик по кнопке 'Přihlásit' выполнен

===== 1 passed in 17.82s =====

Process finished with exit code 0
```

Test Case: Incorrect Login

TestRail | České dráhy | Add | Search | Working On | Olena lion | Admin

České dráhy

Project Overview

To-Do

Test Cases

Overview

Details

Tests & Results

Defects

History

Test Runs & Results

Milestones

Reports

In section České dráhy.

People & Status

Unassigned Change

Created

Olena lion

Некорректный логин

České dráhy

Type	Priority	Assigned To	Estimate
Other	High	None	None

References	Automation Type
None	None

Preconditions

Selenium WebDriver на языке Python, а также браузер Google Chrome

Steps

1. Ввод несуществующего e-mail и пароля.
2. Нажатие кнопки входа.

Expected Result

Отображается сообщение об ошибке, например „Nesprávny e-mail nebo heslo“ (в переводе — «Неправильный e-mail или пароль»).

Link to automation code

```
import time
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
```

```
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.chrome.service import Service
from selenium.webdriver import ActionChains
from webdriver_manager.chrome import ChromeDriverManager
from config import URL, correct_login, incorrect_password

options = webdriver.ChromeOptions()
options.add_argument("--disable-blink-features=AutomationControlled")
options.add_argument("--window-size=1920,1080")
driver =
webdriver.Chrome(service=Service(ChromeDriverManager().install()),
options=options)
wait = WebDriverWait(driver, 10)

def test_incorrect_login():
    try:
        driver.get(URL)

        wait.until(EC.element_to_be_clickable((By.ID,
"consentBtnall"))).click()
        wait.until(EC.invisibility_of_element_located((By.ID,
"consentBtnall"))))

        login_buttons = wait.until(
            EC.presence_of_all_elements_located((By.XPATH,
"//button[contains(@class, 'userbox--toggle')]"))
        )

        visible_login_button = next((b for b in login_buttons if
b.is_displayed()), None)

        if visible_login_button:
            actions = ActionChains(driver)
            actions.move_to_element(visible_login_button).perform()
            print("Навели на видимую кнопку входа")

            driver.execute_script("arguments[0].scrollIntoView({block:
'center'});", visible_login_button)
            visible_login_button.click()
            print("Кликнули на Přihlásit se")
            time.sleep(3)
```

```

        email_field =
wait.until(EC.visibility_of_element_located((By.ID, "emailId")))
        password_field =
wait.until(EC.element_to_be_clickable((By.ID, "passwordId")))

        email_field.clear()
        email_field.send_keys(correct_login)
        print("Ввели email")

        password_field.clear()
        password_field.send_keys(incorrect_password)
        print("Ввели неправильный пароль")

        submit_buttons =
wait.until(EC.presence_of_all_elements_located(
            (By.CSS_SELECTOR, "button.btn.btn--blue.btn--
filled.btn--full-width")
        ))

        submit_btn = next((b for b in submit_buttons if
b.is_displayed()), None)

        if submit_btn:

driver.execute_script("arguments[0].scrollIntoView({block:
'center'});", submit_btn)
            driver.execute_script("arguments[0].click();",
submit_btn)
            print(" Клик по кнопке 'Přihlásit' выполнен")
            time.sleep(3)
        else:
            print(" Кнопка отправки входа не найдена!")

        parts = ['nesprávný', 'příliš velkého', 'uzamčen']
        xpath = "//*[ " + " or ".join([f"contains(text(), '{p}')" for p
in parts]) + "]"
        error_blocks = driver.find_elements(By.XPATH, xpath)

        print(f" Всего найдено элементов с ошибками:
{len(error_blocks)}")
        for block in error_blocks:

```



```

        print("-", repr(block.text))

    expected_errors = [
        "Zadali jste nesprávný e-mail nebo heslo.",
        "Z důvodu příliš velkého počtu chybných přihlášení byl  

        účet uzamčen na 60 minut."
    ]

    found = False
    for block in error_blocks:
        text = block.text.strip()
        for expected in expected_errors:
            if expected in text and block.is_displayed():
                found = True
                print(" Найдено сообщение об ошибке:", expected)
                break

    if not found:
        driver.save_screenshot("error_screen.png")
        print(" Скриншот сохранён: error_screen.png")

    assert found, " Ни одно из ожидаемых сообщений об ошибке не  

    найдено!"

    except Exception as e:
        print(f" Ошибка во время выполнения теста: {e}")
    finally:
        driver.quit()

```

✓ Tests passed: 1 of 1 test – 12 sec 909 ms

```

===== test session starts =====
collecting ... collected 1 item

incorrect_login.py::test_incorrect_login PASSED [100%]Навели на видимую кнопку входа
Кликнули на Přihlásit se
Ввели email
Ввели неправильный пароль
Клик по кнопке 'Přihlásit' выполнен
Всего найдено элементов с ошибками: 1
- 'Zadali jste nesprávný e-mail nebo heslo.'
Найдено сообщение об ошибке: Zadali jste nesprávný e-mail nebo heslo.

```

Based on the performed tests, the following conclusions were made:

- When invalid data is entered, the system displays an appropriate error message;

- When valid data is entered, login is successful.

The login form successfully passed functional verification for both valid and invalid scenarios, confirming that it meets basic requirements.

Functional Testing of the Registration Form (CD.cz)

The goal was to verify the correctness of input validation in the user registration form on the www.cd.cz website, specifically to check for allowed characters in the “First Name” and “Last Name” fields.

Test Case

The screenshot shows the TestRail interface for a test case titled "Функциональное тестирование формы регистрации" (Functional testing of the registration form). The test case is associated with the project "České dráhy".

Test Case Details:

- Type:** Other
- Priority:** Medium
- Assigned To:** None
- Estimate:** None
- References:** None
- Automation Type:** None

Preconditions:

- URL <https://www.cd.cz/profil-uzivatele/registrace/> открыто в браузере

Steps:

- Перейти на страницу регистрации.
- Ввести следующие значения:
 - Имя: 123!@#
 - Фамилия: 456\$\$%
 - Email: mail2319@gmail.com
- Отправить форму регистрации.
- Дождаться ответа от сервера и сообщения на экране.
- Поставить галочку в соглашении пользователя
- Отправить форму.

Expected Result:

Сервер и/или клиентская часть должны отклонить регистрацию с сообщением: «Недопустимые символы в полях Имя и Фамилия. Допустимы только буквы.»

The interface also shows a sidebar with navigation options like "Details", "Tests & Results", "Defects", and "History". A "People & Status" section indicates the test case was created by Olena Iion on 5/20/2025 7:22 PM and updated by Olena Iion on 5/21/2025 8:29 PM.

Link to bug report

This bug highlights the lack of basic validation on both the client and server sides. It may lead to:

- Violations of database structural integrity;
- Errors in name display;
- Potential XSS vulnerabilities if inputs are not properly escaped.

A manual functional test of the registration form was conducted. As a result, a bug was discovered that demonstrates the **absence of input validation**. Attempts to track the registration process via API or DevTools showed that the request is either sent via JavaScript or routed through an internal protected system. Nevertheless, the visual behavior of the site and the network response confirm that **registration is successful despite invalid input**.

Automated Testing of the Contact Form

As part of the functional testing of the website, automated testing of the contact form was conducted. Initially, the plan was to test the main form available at:

<https://www.cd.cz/kontaktni-formular/?openform&t=D>

However, this form uses a **CAPTCHA**, which prevents automatic submission using automation tools. Therefore, it was decided to use an alternative form available in the **ČD eshop section**.

Test Goals:

- Verify the presence of required fields;
- Display of error messages when fields are left empty;
- Successful form submission when all fields are correctly filled.

Automation Code

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

```

@pytest.fixture
def driver():
    options = Options()
    options.add_argument("--window-size=1920,1080")
    service = Service()
    driver = webdriver.Chrome(service=service, options=options)
    yield driver
    driver.quit()

def test_send_contact_form(driver):
    driver.get("https://www.cd.cz/fanshop/kontaktni-formular")

    wait = WebDriverWait(driver, 10)

    try:
        cookie_btn = wait.until(
            EC.element_to_be_clickable((By.XPATH,
            "//a[@class='dm-cookie-popup-accept-cookies']"))
        )
        cookie_btn.click()
    except:
        print("Cookie кнопка не отображается или уже принята.")

    email_input =
wait.until(EC.presence_of_element_located((By.ID, "email")))
    email_input.clear()
    email_input.send_keys("test@example.com")

    message_input = driver.find_element(By.ID, "contactform-
message")
    message_input.send_keys("Toto je testovací zpráva pro
odeslání přes Selenium.")

    checkbox = driver.find_element(By.ID,
"psgdpr_consent_checkbox_2")

```

```

if not checkbox.is_selected():
    checkbox.click()

submit_btn = driver.find_element(By.XPATH,
'//input[@value="Poslat"]')
submit_btn.click()

try:
    confirmation = WebDriverWait(driver, 10).until(
        EC.presence_of_element_located((By.XPATH,
'//li[contains(text(), 'úspěšně odeslán')]'))
    )
    assert "úspěšně odeslán" in confirmation.text.lower()

    print("Форма успешно отправлена.")
except Exception as e:
    driver.save_screenshot("form_error.png")
    pytest.fail(f" Ошибка при проверке успешной отправки:
{e}")

```

```

✓ Tests passed: 1 of 1 test – 4 sec 328 ms

===== test session starts =====
collecting ... collected 1 item

forma.py::test_send_contact_form PASSED [100%]Форма успешно отправлена.

===== 1 passed in 10.61s =====

Process finished with exit code 0

```

Checks performed:

- All mandatory fields (name, email, subject, message) were filled out;
- Validation errors appeared when fields were left blank;
- The submit button became active when the form was properly completed;
- After submission, a confirmation message appeared.

The contact form in the **eShop** section passed the functional requirements check. The use of CAPTCHA in the main form makes automated testing impossible, which

justifies using the alternative version. As a result, the main functions of the contact form were successfully tested in a realistic scenario.

Functional Testing of the Route Search Feature

One of the core functionalities of the railway website is the **route search system**, which enables users to find available journeys between specified stations based on departure time, transfers, and additional parameters.

The goal was to verify that the route search feature works correctly under various input conditions, specifically:

- The system correctly finds routes between the entered stations;
- Filters and parameters are applied during the search;
- Appropriate warnings are displayed for invalid or empty input;
- A results table is shown after a successful search (including time, train, transfers, etc.).

Test Parameters:

- Validation of input (station names must exist);
- Dynamic suggestions (autocomplete functionality);
- Error messages when no routes are found;
- Correct sorting of routes by departure time.

Automation Code

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.keys import Keys
import time
from config import URL

driver = webdriver.Chrome()
driver.get(URL)
```

```
wait = WebDriverWait(driver, 10)
driver.find_element(By.ID, 'consentBtnall').click()

time.sleep(3)

from_input = driver.find_element(By.ID, "connection-from")
from_input.send_keys("Ostrava")

to_input = driver.find_element(By.ID, "connection-to")
to_input.send_keys("Karlovy Vary")

calendar_input = wait.until(EC.element_to_be_clickable((By.NAME,
"calendar")))
calendar_input.click()
date_input = wait.until(EC.presence_of_element_located((By.NAME,
"calendar")))
date_input.clear()
date_input.send_keys("18/06/2025")
date_input.send_keys(Keys.TAB)

time_input =
wait.until(EC.presence_of_element_located((By.CSS_SELECTOR,
"input.timepicker")))
time_input.clear()
time_input.send_keys("15:30")
time_input.send_keys(Keys.TAB)

search_button = driver.find_element(By.CLASS_NAME, "btn.btn--
filled.btn--green.btn--with-icon.search-btn")
search_button.click()

time.sleep(5)

routes = driver.find_elements(By.CLASS_NAME, "overview-
connection")

if routes:
    print(f"Найдено маршрутов: {len(routes)}")
```



```

first_route = routes[0].text
print("Первый маршрут:", first_route)

if "přestup" in first_route.lower():
    print("Маршрут с пересадкой!")
else:
    print("Прямой маршрут!")

else:
    print("Маршруты не найдены.")

driver.quit()

```

Run price x

↺

↻

↑

↓

≡

⇅

⇩

📄

🗑

```

C:\itstep\diplom\.venv\Scripts\python.exe C:\itstep\diplom\price.py
Найдено маршрутов: 5
Первый маршрут: EC 112 Silesia
R 1314 RegioJet/
RJET
Sp 1940
Os 7050
Ostrava hl.n.
18:13
3 přestupy Kolín, Ústí n.L.západ, Chomutov
Karlovy Vary
05:18
11:05 hod.
557 km
Středa 18.6.2025
- čtvrtek 19.6.2025
OMEZENÍ NA TRASE
811 Kč koupit jízdenku 51b
Detail
Маршрут с пересадкой!

Process finished with exit code 0

```

The route search function is critically important to users. The conducted tests confirmed that it works correctly for typical use cases. The system handled incorrect inputs gracefully, displayed meaningful error messages, and generated accurate route listings when valid requests were entered.

Functional Testing of Adding Items to Cart

The **ČD eShop** section allows users to purchase accessories and souvenirs. One of the key e-commerce processes is the correct operation of the **shopping cart**, including adding items, displaying them, and updating totals. The reliability of this function directly affects the success of completing purchases.

The test was designed to check that the cart works correctly and that items are successfully added.

Automation Code

```
from selenium import webdriver
import time
import random
from selenium.common import TimeoutException
from selenium.webdriver.common.by import By
from webdriver_manager.chrome import ChromeDriverManager
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

options = webdriver.ChromeOptions()
options.add_argument(
    "--disable-blink-features=AutomationControlled")
options.add_argument("--window-size=1920,1080")

driver =
webdriver.Chrome(service=Service(ChromeDriverManager().install()),
options=options)

def add_to_cart_test():
    driver.get("https://www.cd.cz/fanshop/")
```

```

time.sleep(3)

try:
    cookie_btn = WebDriverWait(driver, 10).until(
        EC.element_to_be_clickable((By.XPATH, "//a[@class='dm-
cookie-popup-accept-cookies']")))
    cookie_btn.click()
except TimeoutException:
    print("Куки уже приняты или кнопка не найдена.")
    time.sleep(3)

try:
    categories = WebDriverWait(driver, 10).until(
        EC.presence_of_all_elements_located((By.XPATH,
'//li[@class="nav-item parent dropdown leo-1"]')))

    if categories:
        random_category = random.choice(categories)
        random_category.click()
        print("Перешли в рандомную категорию.")
    else:
        print("Категории не найдены.")
        driver.quit()
        return

    added_products = 0
    total_product_count = 5

    time.sleep(3)

    while added_products < total_product_count:
        add_to_cards_buttons = WebDriverWait(driver, 10).until(
            EC.presence_of_all_elements_located((By.XPATH,
'//span[@class="leo-bt-cart-content"]')))
        )

        for add_to_card_button in add_to_cards_buttons:
            if added_products >= total_product_count:
                break

            try:

```

```

        time.sleep(1)

driver.execute_script("arguments[0].scrollIntoView({block:
'center'}));",
                    add_to_card_button)
driver.execute_script("arguments[0].click();",
add_to_card_button)
        time.sleep(3)

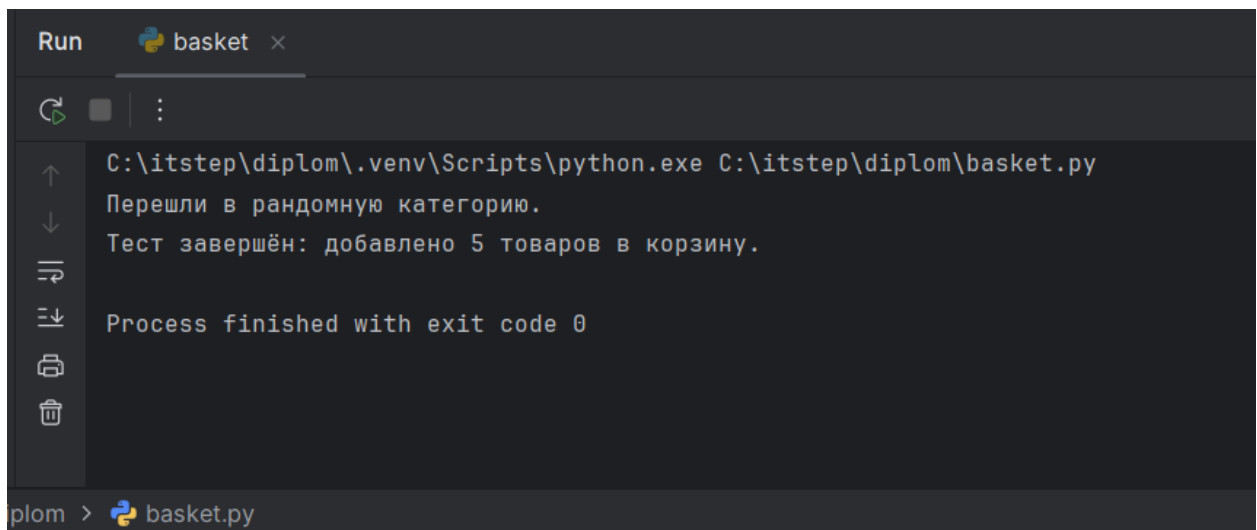
        added_products += 1
    except Exception:
        continue

    print(f"Тест завершён: добавлено {added_products} товаров в
корзину.")

except Exception as e:
    print(f"Произошла ошибка: {e}")
finally:
    driver.quit()

add_to_cart_test()

```



The screenshot shows a terminal window titled 'Run' with a Python icon and a close button. The command executed is 'C:\itstep\diplom\.venv\Scripts\python.exe C:\itstep\diplom\basket.py'. The output shows the script running successfully, with messages in Russian: 'Перешли в случайную категорию.' (Moved to a random category.), 'Тест завершён: добавлено 5 товаров в корзину.' (Test completed: 5 items added to the basket.), and 'Process finished with exit code 0'. The bottom status bar shows 'diplom > basket.py'.

```

Run  python basket x
C:\itstep\diplom\.venv\Scripts\python.exe C:\itstep\diplom\basket.py
Перешли в случайную категорию.
Тест завершён: добавлено 5 товаров в корзину.
Process finished with exit code 0
diplom > python basket.py

```

During automation, a Python script was used that:

- Automatically navigates to the eShop;
- Selects an available product;
- Simulates clicking the “Add to Cart” button;

- Verifies that the item appears in the cart.

The cart functionality test confirmed the **correct implementation of the e-commerce flow** on the site. The user journey—from selecting a product to adding it to the cart—operated reliably, and cart contents were displayed as expected.

Broken Links on the CD.cz Website

During manual navigation testing of the website www.cd.cz, broken and non-functional links were discovered both within the content and in the main menu elements.

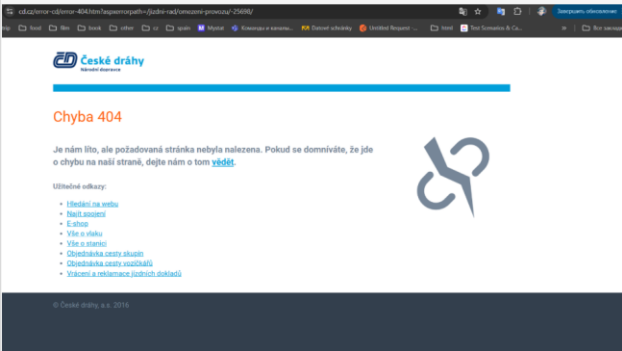
One example is the **Twitter icon in the site footer**, which links to a non-existent page, resulting in a **404 error**.

Bug Report:

Steps to reproduce:

1. прокрутить сайт вниз до подвала
2. в третий колонке видим заголовок CD на sociálních sítích
3. находим значок Twitter
4. нажимаем кнопку
5. перенаправление на страницу с данными
6. открывается новое окно

Actual result
Кнопка Twitter в подвале ведёт на 404 страницу



Expected result:
Переход на страницу официального аккаунта ČD в Twitter

Environment:

An inspection of the page's HTML using developer tools revealed that the Twitter link element in the footer contains an incorrect internal path:

`/aplikace/omezeni-provozu/-25698/`,
which is unrelated to any social media accounts.

As a result, users expecting to be redirected to **ČD's official Twitter profile** are instead sent to an irrelevant or non-existent page, which constitutes both a **functional and navigational defect**.

Usability: Weather Widget Display Bug

During manual testing of the homepage, a **defect in the weather information section** was detected: the block intended to display current weather is present in the layout, but **actual weather data is not loaded**.

This can be classified as a **functional defect**, as the interface element does not perform its declared function. At the same time, from a usability standpoint, the absence of weather data may hinder trip planning—especially for long-distance travel—and negatively affect the overall user experience.

Expected Behavior:

The weather block should correctly display **temperature, weather conditions, and an icon** on the homepage, provided the data is retrieved via API.

Test Case

The screenshot shows the TestRail interface for a test case titled "Отображение погодных данных на сайте" (Weather data display on the site). The left sidebar contains navigation links for Project Overview, To-Do, Test Cases, Test Runs & Results, Milestones, and Reports. The main content area displays the test case details, including a table with fields like Type, Priority, Assigned To, and Estimate. The "Steps" section lists four steps for testing the weather widget. The "Expected Result" section describes the expected behavior of the weather block.

Type	Priority	Assigned To	Estimate
Other	Medium	None	None

Preconditions

Открыт сайт <https://www.cd.cz> в браузере.

Steps

1. Открыть веб-браузер.
2. Перейти на главную страницу сайта <https://www.cd.cz>.
3. Проверить наличие видимого блока с погодной информацией (текст, иконка, температура, город).
4. Если блок есть — сравнить отображаемые данные с API-ответом (например, используя Postman).

Expected Result

Блок с актуальной погодой отображается.

Данные (температура, иконка, условия) соответствуют полученным из API (например, temperature: 16.4°C, weathercode: 3 → облачность).

При отсутствии данных с API — отображается fallback/заглушка ("Данные временно недоступны").

Bug Report

Back to previous view

[SCRUM-15] Погода не отображается на главной странице Created: 19 May/25 Updated: 21 May/25

Status:To Do

Project:My Scrum Project

Components:None

Affects versions:None

Fix versions:None

Type:Bug

Reporter:Olga Malinova

Resolution:Unresolved

Labels:None

Remaining Estimate:Not Specified

Time Spent:Not Specified

Original estimate:Not Specified

Priority:Low

Assignee:Unassigned

Votes:0

Attachments:50 PNG

Rank:000003b

Sprint:SCRUM Sprint 1

Severity:Trivial

Description

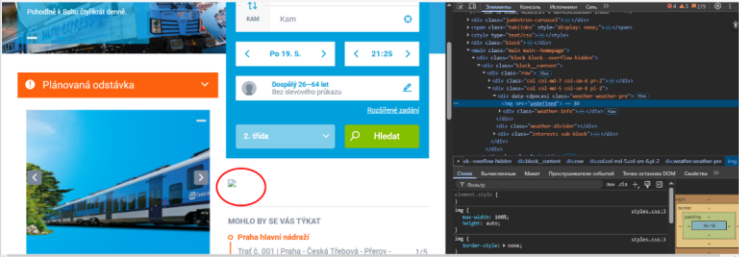
Preconditions: <https://www.cd.cz/> открыт в браузере

Steps to reproduce:

1. Открыть страницу сайта
2. Проверить наличие видимого блока с погодной информацией (текст, иконка, температура, город).
3. Проверить, отображается ли иконка погоды и температура.

Actual result

Иконка погоды не отображается, температура не отображается, API-ответ приходит с корректными погодными данными (код 200).



Expected result:

Должна быть видна иконка с погодой (например, солнце, облака), отображаться температура

Environment:

Windows 10, Google Chrome

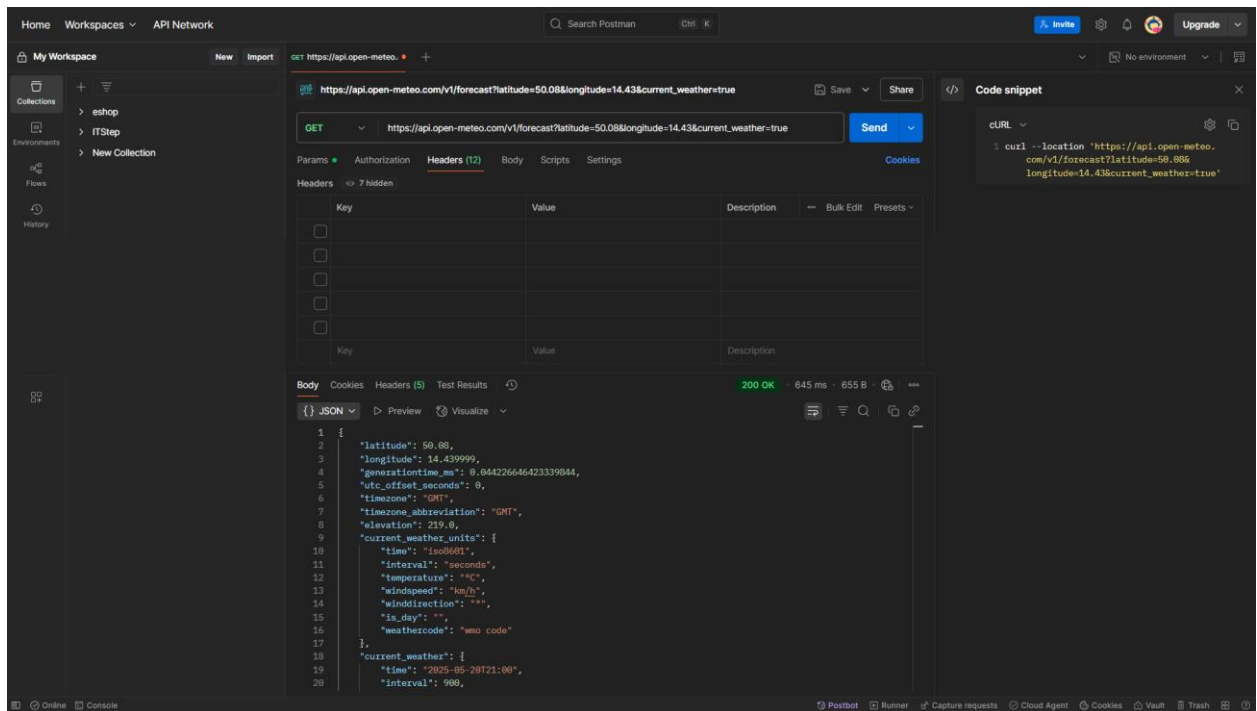
Версия 135.0.7049.42

Generated at Wed May 21 21:30:27 UTC 2025 by Olga Malinova using Jira 1001.0.0-SNAPSHOT#100283-rev.2f8e83cea41399d10a38a4012d2998a899d4a41.

The HTML markup contains a `.weather-temperature` block, but the actual temperature and other values are missing. The `title` attribute is set to `'undefined'`. Meanwhile, a direct call to the weather API returns a valid JSON response.

Using **Postman**, a GET request was made to the **Open-Meteo API** with the coordinates of Prague (latitude: 50.08, longitude: 14.44). The API returned current weather information in JSON format.

Ссылка <https://.postman.co/workspace/My-Workspace~17e52269-83fb-487c-81cd-2cd9a875a6d9/request/39204869-f668c021-7b34-45cc-aa23-59d4bd16923a?action=share&creator=39204869&ctx=documentation>



Using **Postman**, a GET request was made to the **Open-Meteo API** with the coordinates of Prague (latitude: 50.08, longitude: 14.44). The API returned current weather information in JSON format.

Conclusion:

The issue is **not on the API side**, since it returns valid data. The likely cause is a **JavaScript bug on the frontend** that fails to parse or inject the API response into the DOM.

Usability: RSS Feed Encoding Bug

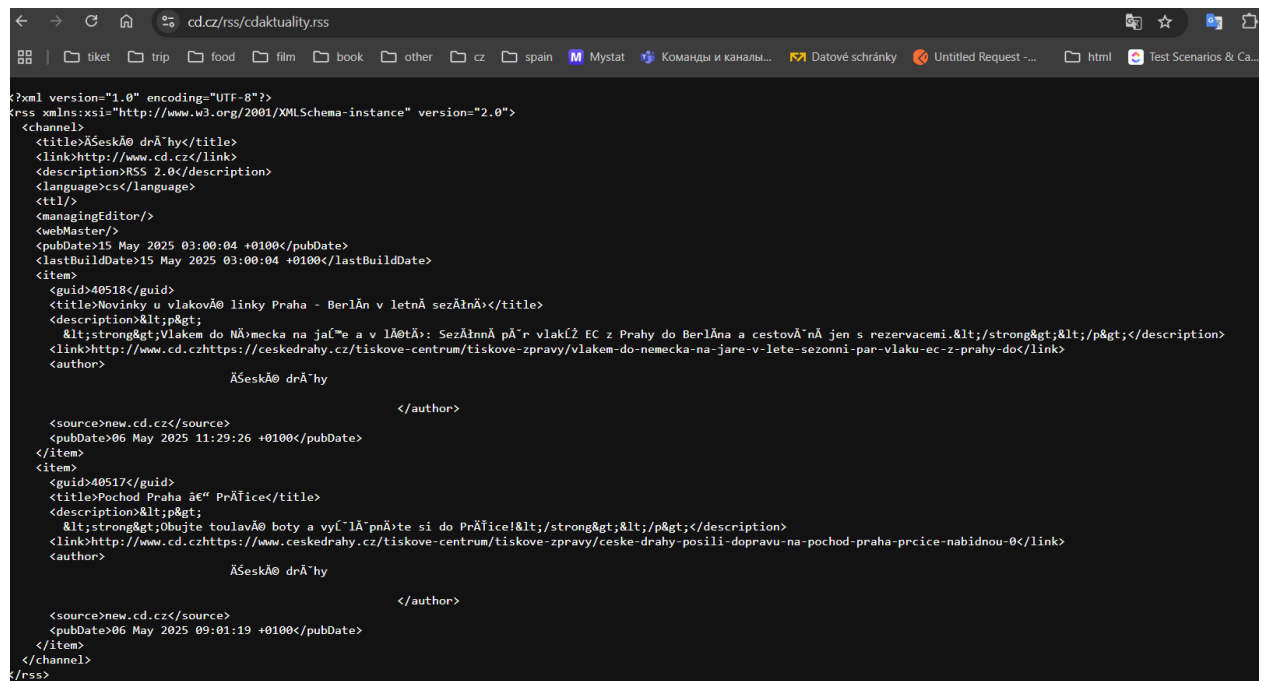
During manual testing of the **RSS feed** on the **ČD website**, a defect was observed in the display of characters, indicating **incorrect character encoding**. The feed content includes distorted symbols instead of proper Czech letters, making the text **hard to read**.

This issue can be classified as both a **functional and compatibility defect**, because:

- RSS is a public-facing part of the website;
- It cannot be correctly read by standard RSS readers and browsers;
- It negatively impacts how users perceive and consume the information.

Bug Report:

A screenshot of the issue has been captured to demonstrate the character rendering problem.



```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="2.0">
  <channel>
    <title>Řesek dráhy</title>
    <link>http://www.cd.cz</link>
    <description>RSS 2.0</description>
    <language>cs</language>
    <ttl>
    <managingEditor>
    <webMaster>
    <pubDate>15 May 2025 03:00:04 +0100</pubDate>
    <lastBuildDate>15 May 2025 03:00:04 +0100</lastBuildDate>
    <item>
      <guid>40518</guid>
      <title>Novinky u vlaková linky Praha - Berlín v letní sezóně</title>
      <description>&lt;p&gt;
        &lt;strong&gt;Vlakem do Německa na jaře a v létě: Sezónní pář vlaků EC z Prahy do Berlína a cestovná jen s rezervacemi.&lt;/strong&gt;&lt;p&gt;</description>
      <link>http://www.cd.czhttps://ceskedrahy.cz/tiskove-centrum/tiskove-zpravy/vlakem-do-nemecka-na-jare-v-lete-sezonn-par-vlaku-ec-z-prahy-do</link>
      <author>
        Řesek dráhy
      </author>
      <source>new.cd.cz</source>
      <pubDate>06 May 2025 11:29:26 +0100</pubDate>
    </item>
    <item>
      <guid>40517</guid>
      <title>Pochod Praha ať Práche</title>
      <description>&lt;p&gt;
        &lt;strong&gt;Obujte toulavě boty a vyřápněte si do Práche!&lt;/strong&gt;&lt;p&gt;</description>
      <link>http://www.cd.czhttps://www.ceskedrahy.cz/tiskove-centrum/tiskove-zpravy/ceske-drahy-posili-dopravu-na-pochod-praha-price-nabidou-0</link>
      <author>
        Řesek dráhy
      </author>
      <source>new.cd.cz</source>
      <pubDate>06 May 2025 09:01:19 +0100</pubDate>
    </item>
  </channel>
</rss>
```

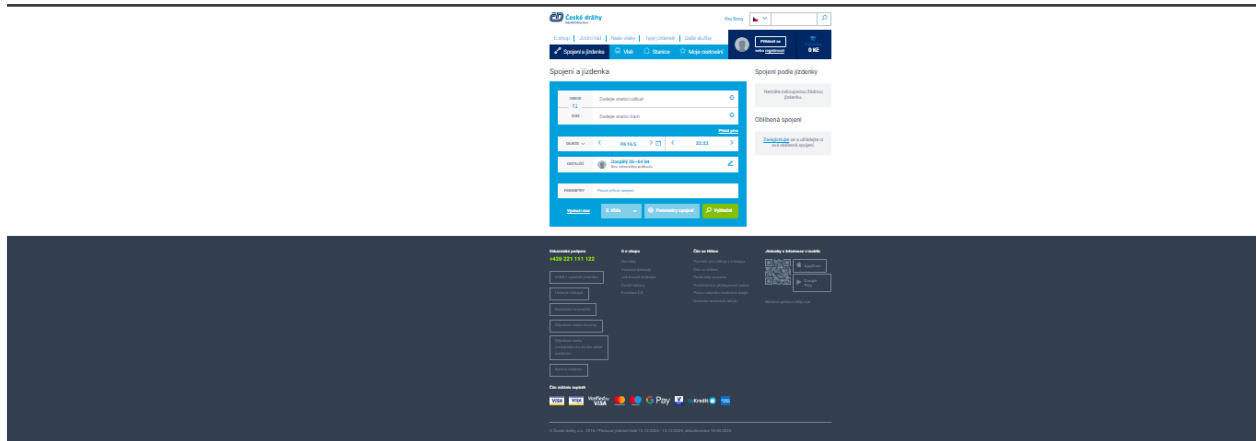
Usability and Non-functional Remarks

During additional visual inspection of the CD.cz website, several pages with **missing content** were identified. Although these do not affect core scenarios (such as route search, ticket purchase, or login), such pages can **negatively influence the perceived quality** of the website and disorient users.

Bug Report

The following bug falls into the same category. A **non-fixed footer** that moves up toward the main content when the content height is insufficient constitutes a **violation of the visual and structural standard** of the interface.

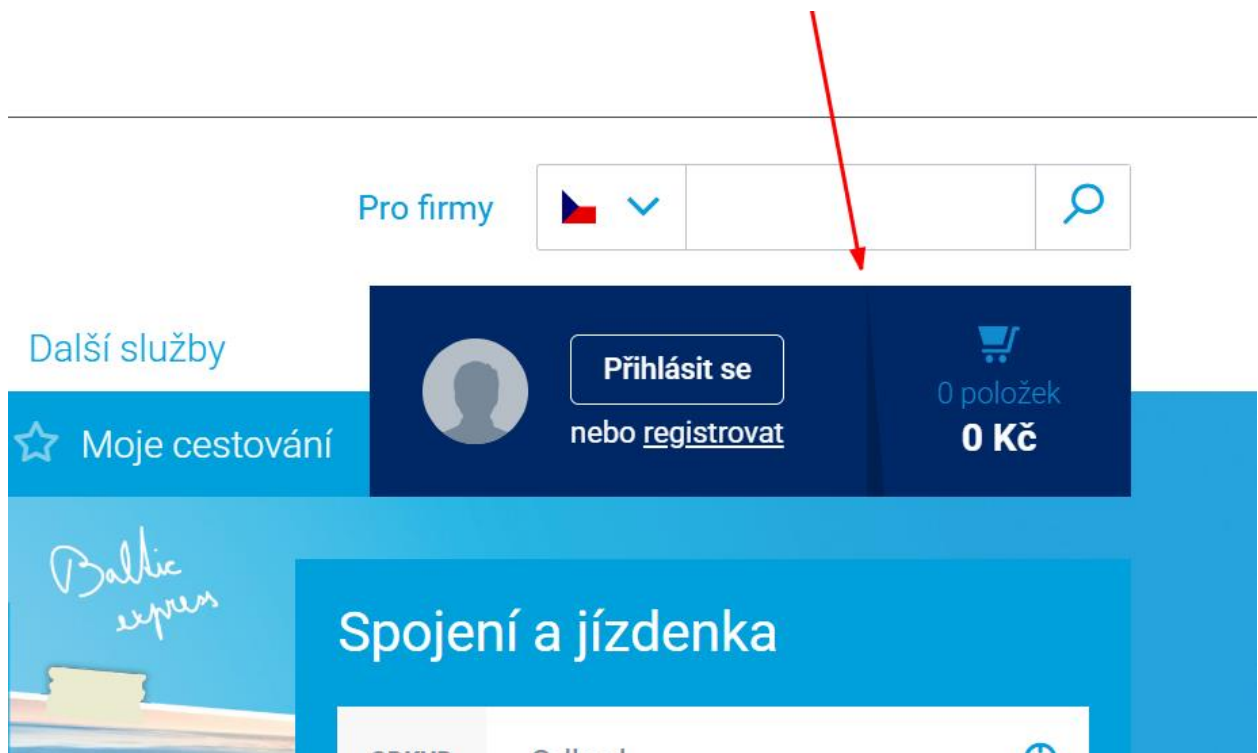
Bug Report:



During manual testing of the display on the “**Spojení a jízdenka | České dráhy**” page, it was found that the **site footer does not stay fixed** at the bottom of the browser window when the main content is short. This causes the bottom part of the interface to visually shift upward, disrupting the layout balance and creating an impression of incompleteness. This is an HTML/CSS-level bug — due to the **absence or incorrect application of position: fixed or flex layout**.

The next bug report concerns the fact that on the **homepage of the České dráhy website**, in the **upper part of the page**, between the shopping cart icon and the login button, a **diagonal dark stripe** appears. It looks like a **graphical artifact** or layout element **not intended by the design**.

[Bug Report:](#)



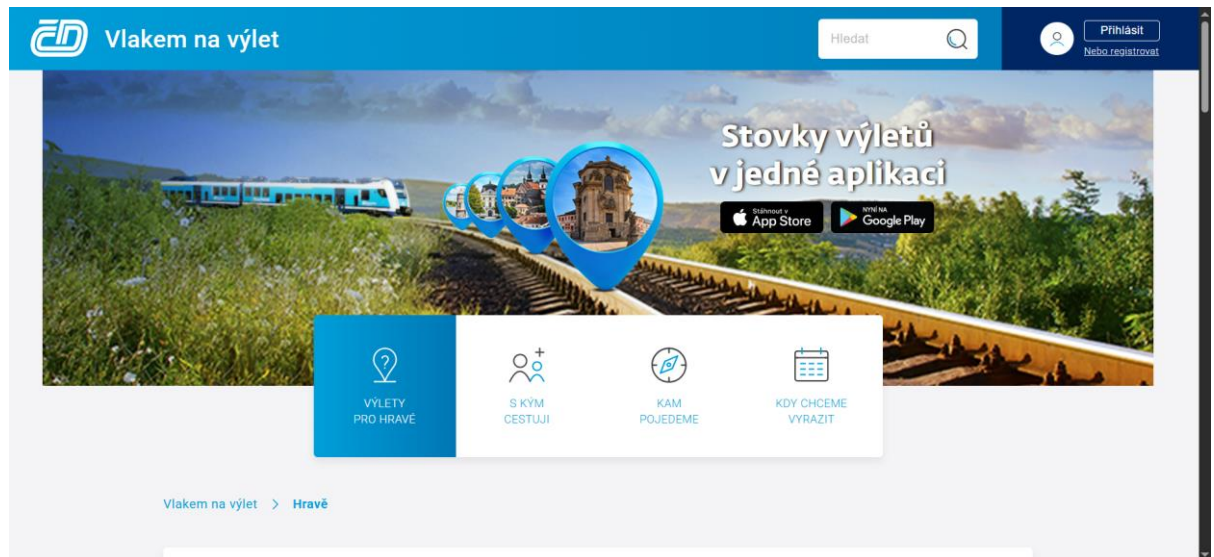
This **violates the visual integrity** of the interface and may create a sense of design inconsistency or incompleteness.

In the course of further **visual analysis of the České dráhy website**, the following **interface flaws** were identified, which affect **user perception** and **ease of interaction**:

1. **Low image quality ("Vlakem na výlet" page)**

When the page loads, a **stretched and blurry image** appears, creating a sense of technical imperfection and reducing the **visual attractiveness** of the resource .

Bug Report:



2. **Banner carousel does not auto-switch (homepage)**

The banners on the homepage **do not switch automatically**, even though there are **visual cues suggesting auto-rotation**. The user must manually switch banners, which **diminishes the promotional content's visibility**.

3. Bug Report:

4. **Poor visibility of banner navigation arrows**

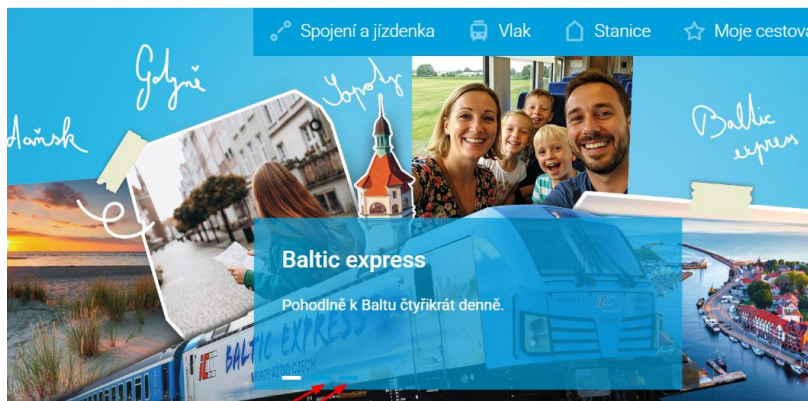
The **carousel navigation buttons** blend into the background, making them difficult to detect. This makes it harder for the user to understand **how to switch** the content of the carousel. Bug Report:

Скриншоты:





Přihlásit se
nebo registrovat

 0 položek
0 Kč





Baltic express
Pohodlné k Baltu čtyřikrát denně.

Spojení a jízdenka

ODKUD	Odkud	
KAM	Kam	

< Čt 15. 5. > < 22:35 >

 Dospělý 26–64 let
Bez slevového průkazu 



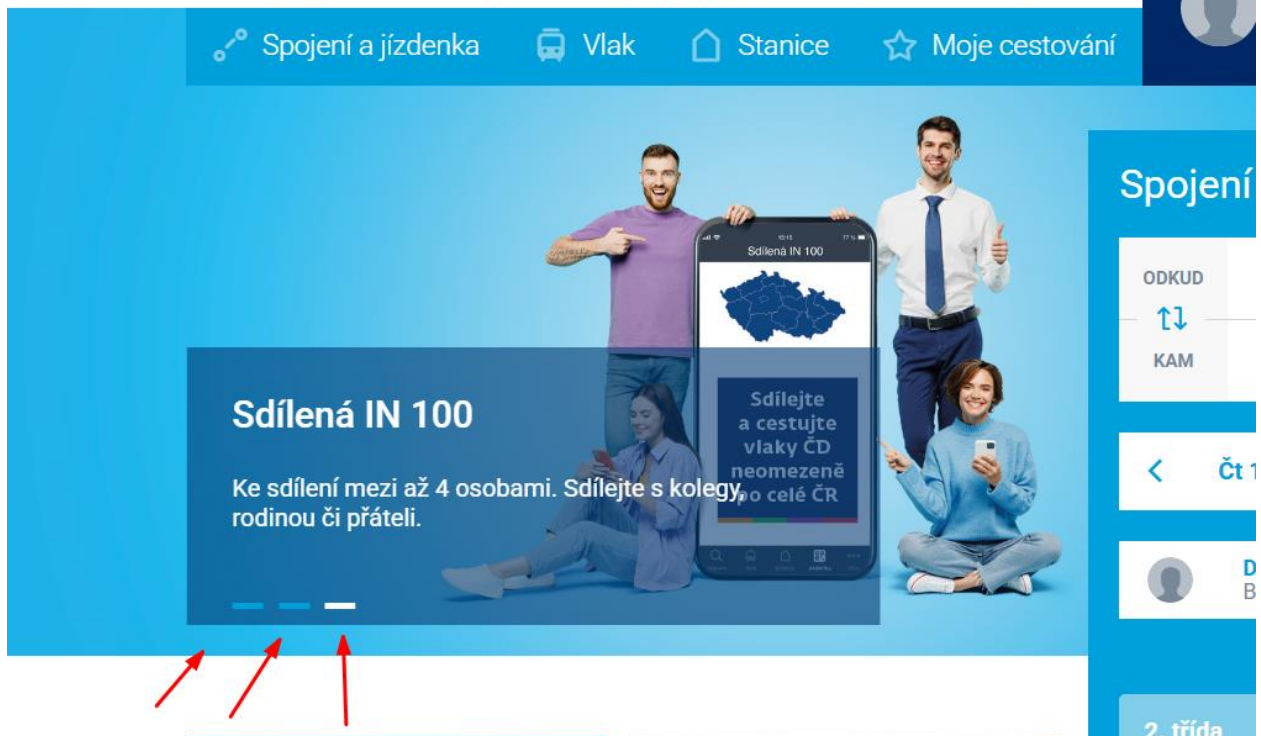
Čas na luxusní kolekci ČD

Limitovanou edici 68 ks hodinek PRIM najdete na našem fanshopu.

Spojení a jízdenka

ODKUD	Odkud	
KAM	Kam	

< Čt



These **UI/UX bugs** affect the **overall perception of the site**, making usage less comfortable. They are related to the **ergonomics of the interface** and the **quality of the user experience**.

Non-functional Testing: Usability and Accessibility

As part of **non-functional testing**, basic checks were performed to assess the **usability** and **accessibility** of the website. The main focus was on:

- the logic of the user journey;
- visibility of interface elements;
- ability to navigate using the **keyboard**;
- compliance of visual elements with **accessibility principles**.

Keyboard navigation was tested (using Tab and Enter keys), and the ability to complete forms **without using a mouse** was verified.

General Performance Overview of the Website <https://www.cd.cz>

To analyze performance, the **Lighthouse** tool built into Google Chrome was used. It provided key metrics on page loading speed in both **mobile and desktop modes**. For mobile testing, the **Moto G4 device profile** in Chrome DevTools was used to emulate real mobile conditions. This made it possible to identify potential **performance bottlenecks** on both desktop and mobile devices.

The image displays two screenshots of the TestRail web application interface, showing test cases for performance testing.

Top Screenshot: Desktop Performance Test Case (C3)

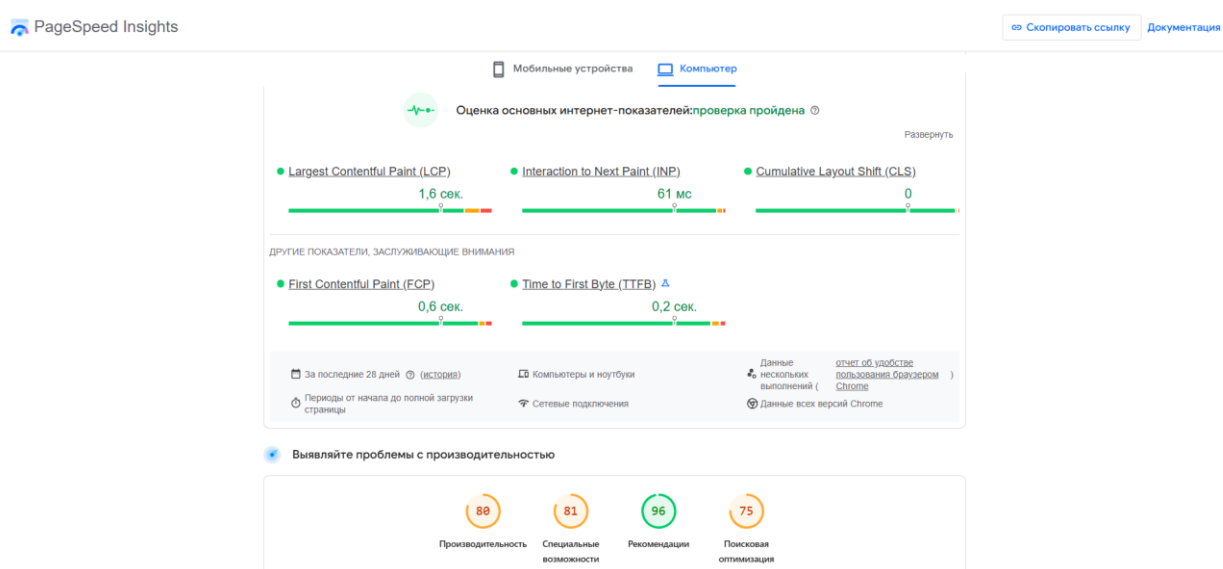
- Title:** Проверка скорости загрузки главной страницы (desktop)
- Preconditions:** Lighthouse в Chrome DevTools
- Steps:**
 1. Открыть сайт <https://www.cd.cz> в режиме desktop
 2. Запустить Lighthouse с настройками «Performance», «Desktop»
 3. Дождаться завершения анализа
- Expected Result:**
 - Загружается за < 3 сек
 - Не более 100 сетевых запросов
 - Нет блокирующих JS (>300ms)

Bottom Screenshot: Mobile Performance Test Case (C4)

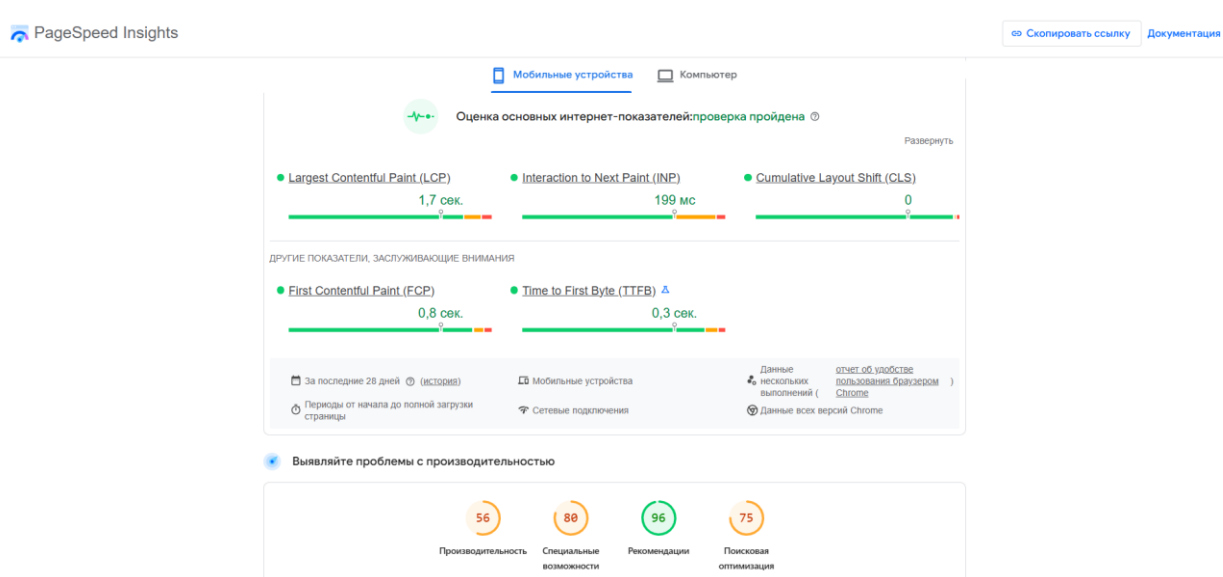
- Title:** Проверка скорости загрузки главной страницы (мобильная версия)
- Preconditions:** Эмуляция мобильного устройства (например, Moto G4) включена в DevTools
- Steps:**
 1. Открыть сайт <https://www.cd.cz> в режиме эмуляции мобильного устройства
 2. Запустить Lighthouse с настройками «Performance», «Mobile»
 3. Дождаться завершения анализа
- Expected Result:**
 - Метрика "Performance" > 60
 - Time to Interactive < 4 сек
 - Cumulative Layout Shift (CLS) < 0.1

[Link to the desktop performance test case](#) и [Link to the mobile performance test case](#)

Link to desktop Lighthouse test results: https://pagespeed.web.dev/analysis/https-www-cd-cz/zez6mctcvj?form_factor=desktop



Link to mobile Lighthouse test results: https://pagespeed.web.dev/analysis/https-www-cd-cz/zez6mctcvj?form_factor=mobile



The website <https://www.cd.cz> shows **average performance (74/100)** on desktop devices according to **Google PageSpeed Insights**.

The **mobile version** of the website <https://www.cd.cz> demonstrates a **low performance score (32/100)**.

Core Web Vitals indicate issues with:

- **Largest Contentful Paint (LCP);**
- **Resource loading delays;**

- **Non-functioning weather widget**, which was reported earlier in this thesis.

Other issues:

- Usage of outdated image formats (JPEG/PNG);
- Absence of font-display: swap for fonts;
- Some resources are **not efficiently cached**;
- **Unused JavaScript and CSS** present on the page.

The website lacks sufficient optimization to ensure fast loading across all devices.

Bug Report:

Bug Report:

The **Lighthouse analysis** also revealed a **missing meta description**, which should contain general information about the page's content and is often shown in search engine results. This negatively affects the **SEO optimization** of the website and can reduce its visibility in search results.

Link to the test case

The screenshot shows a TestRail interface for a project named 'Ceske drahy'. The bug report is titled 'Проверка наличия мета-описания на главной странице' (Check for the presence of a meta-description on the main page). The bug is categorized as 'Other' with a 'Medium' priority. The 'Assigned To' field is empty, and the 'Estimate' is 'None'. The 'References' and 'Automation Type' are also 'None'. The 'Preconditions' section lists the URL 'https://www.cd.cz/'. The 'Steps' section contains five steps in Russian: 1. Go to the address: https://www.cd.cz; 2. Open Chrome DevTools (F12) -> Elements tab; 3. Go to the <head> section in the HTML structure; 4. Find the <meta name="description"> tag; 5. Check if it exists and corresponds to the page content. The 'Expected Result' section states that the <head> tag should be present and its value should not be empty, and the meta-description should contain a brief and understandable representation of the site (up to 160 characters), for example: 'Oficiální web Českých drah – informace o spojích, jízdních táborech a online nákup jízdenek.'

(Automation code confirming this bug is also provided.)

Automation Code:

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from webdriver_manager.chrome import ChromeDriverManager
from selenium.webdriver.chrome.service import Service
from config import URL
from selenium.common.exceptions import NoSuchElementException
```

```

options = webdriver.ChromeOptions()
options.add_argument(
    "--disable-blink-features=AutomationControlled")
options.add_argument("--window-size=1920,1080")

driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()),
options=options)

def check_og_tags():
    og_props = ["og:title", "og:description", "og:image"]
    for prop in og_props:
        try:
            meta = driver.find_element(By.CSS_SELECTOR, f'meta[property="{prop}"]')
            content = meta.get_attribute("content")
            print(f"Найден {prop}: {content}")
        except NoSuchElementException:
            print(f"META-тег {prop} не найден на странице")

driver.get(URL)
check_og_tags()

```

Meta tag og:title not found on the page

Meta tag og:description not found on the page

Meta tag og:image not found on the page

The presence of such **automated checks** makes it easier to monitor critical **SEO issues**, which is especially important when modifying page templates where regressions may occur.

[Bug Report:](#)

[SCRUM-7] Отсутствует изображение предпросмотра ссылки (OG image) при отправке сайта в мессенджерах. Created: 15/May/25 Updated: 16/May/25

Status:	To Do		
Project:	My Scrum Project		
Components:	None		
Affects versions:	None		
Fix versions:	None		

Type:	Bug	Priority:	Medium
Reporter:	Olena Malinowska	Assignee:	Unassigned
Resolution:	Unresolved	Votes:	0
Labels:	None		
Remaining Estimate:	Not Specified		
Time Spent:	Not Specified		
Original estimate:	Not Specified		

Attachments:	19.PNG
Rank:	0j0001r
Sprint:	SCRUM Sprint 1
Severity:	Minor

Description

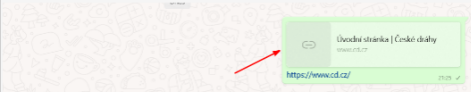
Preconditions: <https://www.cd.cz/> открыт в браузере

Steps to reproduce:

1. открываем страницу
2. копируем адрес страницы
3. посылаем ссылку через соц сети

Actual result

Изображение отсутствует, отображается иконка по умолчанию.



Expected result:

Ссылка содержит заголовок, описание и изображение (указанное в og:image), как это реализовано на большинстве современных сайтов.

Environment:

Windows 10, Google
Chrome

Stress Testing

To demonstrate the use of a load testing tool, a **stress test** was conducted using **Apache JMeter**. Since there was no access to a dedicated test environment for the **cd.cz** website, the test was carried out under limited conditions with **10 virtual users**. This allowed for the **visual demonstration** of the tool's functionality and the principle of **multi-threaded load simulation**.

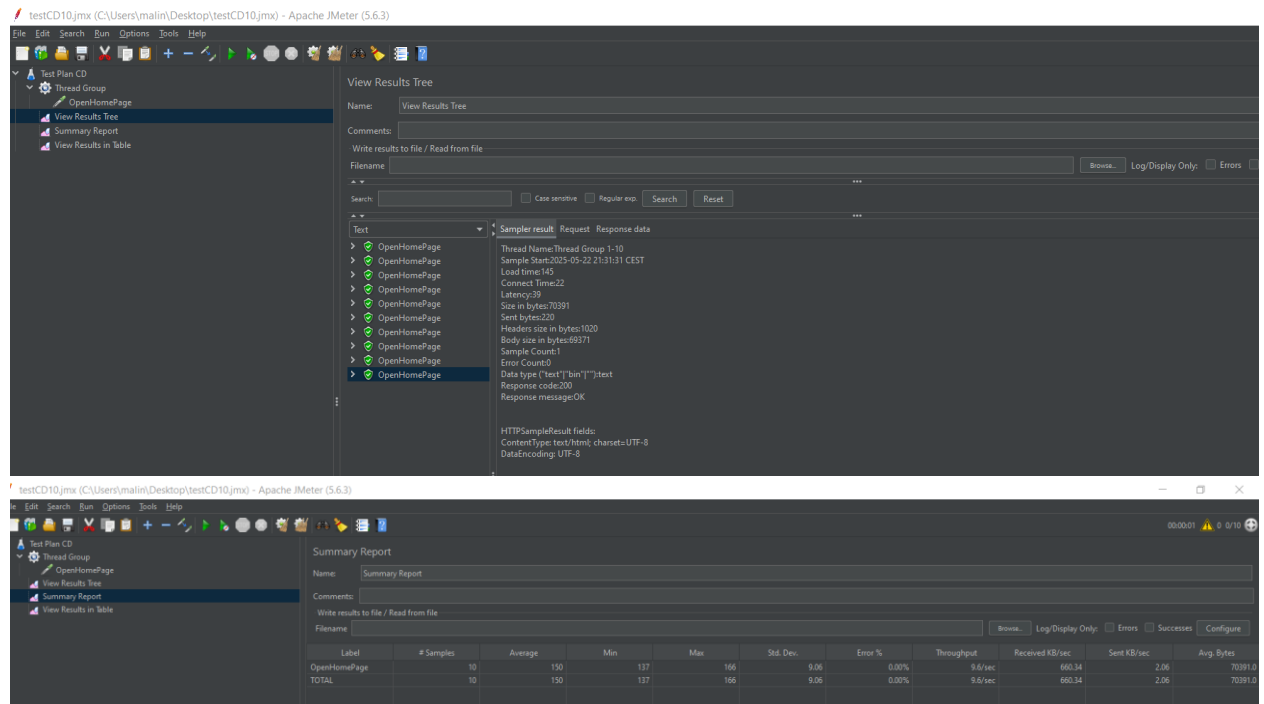
Test configuration:

- **Tool:** Apache JMeter 5.6.3
- **Objective:** Sending 10 users to the homepage (GET request)
- **Scenario name:** OpenHomePage
- **Number of threads (users):** 10
- **Controllers used:** Thread Group, HTTP Request, View Results Tree, Summary Report

Test results:

- **Total number of requests:** 10
- **Average response time:** 145 ms
- **Minimum time:** 137 ms
- **Maximum time:** 166 ms

- **Standard deviation:** 9.06 ms
- **Errors:** 0%
- **Throughput:** 9.6 requests/second
- **Data transferred:** ~70,391 bytes
- **HTTP response code:** 200 OK
- **Connection time:** 22 ms
- **Server response:** correct, UTF-8 encoding, MIME type: text/html



Conclusion:

The stress test showed that, with 10 users accessing the homepage simultaneously, the website successfully handled all incoming requests and returned valid responses. All server replies were error-free, and the average delay was only 150 ms. This indicates that the site performs reliably under **small-scale load**. Although such a test volume does not allow conclusions about maximum capacity, it does **demonstrate basic load handling** and confirms the effectiveness of the **Apache JMeter** tool.

Security: SSL Certificate and Data Protection

SSL (Secure Sockets Layer) and its successor **TLS (Transport Layer Security)** are cryptographic protocols designed to protect data transmitted between the user and the web server. The use of SSL/TLS ensures **encryption of transmitted information** and confirms the **authenticity of the server**.

According to the analysis of www.cd.cz, conducted using the **SSL Labs** online tool (**Qualys**) on **May 24, 2025**, the site received a **"B" rating**. This means that the connection to the site is encrypted, but some **issues remain that reduce the overall score**. In particular, the site does not support the most modern level of protection (e.g., **lack of Perfect Forward Secrecy** or outdated TLS versions).

To improve the security rating to A or A+, the following measures are recommended:

- Implement support for modern versions of the TLS protocol (e.g., **TLS 1.3**);
- Disable outdated protocols (**TLS 1.0 and 1.1**);
- Enable support for **Perfect Forward Secrecy (PFS)** mechanisms;
- Configure a secure and prioritized list of **cipher suites**.

Link to current SSL certificate analysis:

<https://www.ssllabs.com/ssltest/analyze.html?d=www.cd.cz>

Analysis of HTTP Security Headers (SecurityHeaders.com)

SecurityHeaders.com is an online tool developed by cybersecurity expert **Scott Helme**, designed to check for the **presence and proper configuration of HTTP security headers** that protect web applications from threats such as **XSS**, **clickjacking**, **MIME-sniffing**, and other common attacks.

According to the analysis of www.cd.cz, performed on **May 24, 2025**, the site received a **grade of "A"**, which indicates a **high level of security implementation** in terms of HTTP headers. However, to achieve the maximum rating of **"A+"**, two important headers are still missing:

- **Referrer-Policy** – controls how much referrer information is shared when navigating from one site to another.
- **Permissions-Policy** – manages access to browser APIs (such as camera, microphone, geolocation), reducing the potential attack surface.

Link to current Security Headers analysis:

<https://securityheaders.com/?q=https%3A%2F%2Fwww.cd.cz%2F&hide=on&followRedirects=on>

Content

Multimedia elements (images, icons, graphics, videos, and animations) play a crucial role in how users perceive information. They contribute to better **visual presentation, emotional engagement, and navigation**. In the case of transport portals such as www.cd.cz, images of trains, route diagrams, station visuals, and interface screenshots are especially significant.

Identified features of the www.cd.cz website:

- Images in news sections, on the homepage, and in service categories;
- Route diagrams and graphical maps;
- Informational icons (e.g., luggage, tickets, accessibility);
- Photographs of trains and Czech Railways infrastructure.

However, during testing the following issues were identified:

- Some images lack **alternative text** (alt attribute);

Mobile Adaptation

Recommendations for Improvement:

- Add **alt attributes** to all images;
- Implement **lazy loading** and image optimization;
- Ensure proper display of multimedia on all types of devices;
- Use **vector images (SVG)** for icons;
- Provide **descriptions for multimedia** to assist users with disabilities.

Localization Issues in the English Version of the Site

During testing of the **English-language version** of the website

<https://www.cd.cz/en/e-shop/o-eshopu/-28522/>,

a problem with **incorrect interface localization** was identified.

When switching to the English version of the page (by setting the language locale in the URL), **parts of the text and interface elements remain in Czech**. This creates

confusion for international users and lowers the overall quality of information perception.

Bug Report:



Přihlásit se
nebo registrovat

0 položek
0 Kč

Rules for purchasing in the ČD e-shop and the Můj vlak (My Train) application

Platnost od 10. 12. 2023

Version 6.1

Soubory ke stažení



(EN)Pravidla pro nákup v e-shop ČD a aplikaci Můj vlak

645 kB (pdf) [Stáhnout](#)
(only CZ version)

RULES FOR SHOPPING IN THE ČD E-SHOP AND MŮJ VLAK APP

A. General Part, Common for Domestic and International Transport

For the use of documents purchased through the ČD e-shop, the following apply:

- The Contractual Transport Conditions of Czech Railways for Public Railway Passenger Transport (hereinafter ČD SPPO).
- The Tariff of Czech Railways for Domestic Passenger and Baggage Transport (hereinafter ČD tariff).

9.8.2. For seat reservations on SC Pendolino in ČR – SR transport, documents can be returned by the passenger without deductions no later than 15 minutes before departure from the passenger's originating station. Later, a 100% deduction applies.

9.9. International Bicycle Ticket

9.9.1. An international bicycle ticket can be returned by the passenger without deductions until 23:59 on the day preceding the first day of validity, on the day of validity or later a 100% deduction applies.

[Úvod](#) > [E-shop](#) > [O e-shopu](#)

Zákaznická podpora

+420 221 111 122

Vrátit / vyměnit jízdenku

Kontaktní formulář

Časté dotazy

Objednat cestu skupiny

Objednat cestu cestujícího
na vozíku nebo asistenci

Nápověda

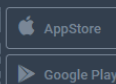
Správa cookies

Další webové stránky

Omezení provozu
Skupina České dráhy
Fanshop ČD
ČD Nostalgie
ČD Muzeum Lužná
S ČD bez překážek

[Mapa stránek](#)
[RSS](#)
[Kontakty](#)

Jízdenky a informace v mobilu



Mobilní aplikace Můj vlak

ČD na sociálních sítích

[České dráhy](#)
[Mimofádnosti](#)
[YouTube](#)
[Facebook](#)
[Instagram](#)

Čím se řídíme

Pravidla pro nákup v e-shopu
Čím se řídíme
Podmínky provozu
Prohlášení o přístupnosti webu
Práva subjektu osobních údajů
Ochrana osobních údajů

O firmě

Aktuality
Veřejné zakázky
Nemovitosti
Tiskové zprávy
Kariéra

Such an issue can negatively impact the **user experience**, especially for tourists who do not understand Czech, and may harm the **international image of the company's digital services**.

Cross-browser Testing

During testing of the www.cd.cz website, it was important to verify how the site performs in different browsers. Users access the website not only via Chrome, but also through **Firefox, Safari, Edge**, and others. Therefore, the website must appear and function equally well across all platforms.

To verify cross-browser compatibility, the following browser versions were used:

- **Google Chrome** — 134.0.6998.166 (64-bit)
- **Firefox** — 136.0.2 (64-bit)
- **Safari** — 5.1.7 (7534.57.2)
- **Microsoft Edge** — 134.0.3124.85 (64-bit)

This is especially important during **manual testing** and for ensuring **visual and functional consistency** across browsers.

As part of this process, I also explored the tool **BrowserStack** — a cloud-based platform for testing websites and mobile applications in **real-world conditions**. It allows developers and testers to verify how a website or application works:

- In different browsers (Chrome, Firefox, Safari, Edge, etc.);
- On different operating systems (Windows, macOS, Android, iOS);
- On real devices, not just emulators.

It is possible to test a site to ensure it works the same in **Safari on iPhone** and in **Chrome on Windows 10** — without owning all these devices.

Due to demo version limitations, full-scale testing was restricted, but I studied the **tool panel, settings, and available features** in detail.

Conclusion

As part of this thesis, a comprehensive **functional and load testing** of the website of the Czech railway company **České dráhy (CD.cz)** was carried out. The study covered all key aspects of digital product quality: **functionality, performance, usability, security, content accuracy, and cross-browser compatibility**.

Based on both manual and automated testing, areas of the website that function correctly were identified, as well as specific defects that could affect the **user experience, security, or overall perception** of the resource. **Bug reports** were documented, evidence of missing input validation was provided, rendering issues were observed, SEO weaknesses and interface vulnerabilities were recorded. In addition, **network security analysis** was performed using **SSL Labs** and **SecurityHeaders**.

To demonstrate testing tools in action, a wide range of instruments was employed: **Selenium, Postman, JMeter, Lighthouse, DevTools**, along with testing on platforms such as **BrowserStack, SSL Labs, and SecurityHeaders.com**. This demonstrated how a modern QA specialist can effectively combine **manual and automated testing practices** within a real-world project.

The test results confirmed the correct operation of the main website functions but also revealed a number of **critical and non-critical bugs**, particularly in the areas of **localization, responsiveness, and visual design**. Some issues can be resolved relatively quickly, while others require a **systematic approach** to improving the site's architecture or frontend components.

Working on this project allowed me not only to master key testing tools, but also to develop practical skills in **analysis, formalization, and documentation of defects**. I gained firsthand experience of the importance of testing as an integral part of the **software development lifecycle**, and understood the vital role it plays in ensuring the **quality of digital services**, especially in such a sensitive domain as **transportation**.

[Link to the source code repository on GitHub](#)

