

Test Planning and Strategy Execution

The steps involved in planning and executing testing are:

- Use cases/user stories have been clearly defined
- To ensure high quality, we need to test for all the quality aspects.
- Teams should focus on testing and development equally to deliver high-quality software.
- For testing to be successful, it has to be practiced at the micro and macro levels of the application.
- Shift-left testing calls for automation and CI / CD practices to build quality-in.
- Shift-left testing embodies the aphorism “Quality is the team’s responsibility.”
- All team members have to possess testing skills at a certain competency level.
- The two monolith terms, manual and automation testing, mask a vast set of testing skills in them.
- Full stack testing is to apply a set of ten different testing skills at different layers of the application individually and holistically while shifting testing to the left.
- Build test cases and test scripts based on use cases
- Define acceptance criteria for each feature and each process
- Conduct testing
- Review and approve test results
- Migrate changes to production or for the next level of testing (such as user acceptance testing)

Test coverage

Functional and non-functional security and performance testing scenarios were performed with test coverage. In addition to the documentation in the project, a document with the test covered was added. The performance and security tests contain reports with information and analysis of the tested application and have been attached. Defects have been added to the test life cycle in the application.

Automated Regression Testing

The test management and automation tools needed to run the test have been defined and were used in this framework: Cypress Version: 10.1.0 & Cucumber.

The Design Pattern was defined in order to meet the scenarios of each story: Page Object Model with loadable components (POC - reference: <https://martinfowler.com/articles/bigQueryPOC.html>) & AAA (reference: <https://martinfowler.com/articles/practical-test-pyramid.html>).

Performance testing and risks

Non-functional requirements

The system must perform in terms of redirection with millisecond latency. Because of this, the performance test strategy was to meet the expectations for a possible high volume of sales of travel tickets by analyzing the speed and stability under load and stress. The tool to explore the possibilities was the GTmetrix (reference: <https://gtmetrix.com/>) tool to measure page performance based on user-centric metrics and pentest tools (reference: <https://pentest-tools.com/home>) to simulated cyber-attack where professional hackers break into corporate networks to find weaknesses before attackers do.

GTmetrix Tool

The GTmetrix Tool analyzed the application's performance on the home page. The home page showed a med-low performance when the home page downloaded all your page resources such as HTML, CSS, JavaScript, and images.

Top Issues

All	FCP	LCP	TBT	CLS	These audits are identified as the top issues impacting your performance .	
IMPACT		AUDIT				
Med-Low	Serve static assets with an efficient cache policy		Potential savings of 404KB		▼	
Med-Low	Use a Content Delivery Network (CDN)		11 resources found		▼	

Despite this first analysis reporting this med-low impact, the application showed an A concept in a performance analysis on the home page. This helps a lot in the journey of purchasing tickets in the application.

GTmetrix Grade ?

B	Performance ? 80%	Structure ? 87%
---	----------------------	--------------------

Web Vitals ?

LCP ? 400ms	TBT ? 349ms	CLS ? 0.13
----------------	----------------	---------------

Summary

Performance

Structure

Waterfall

Video

History

Performance Metrics

The following metrics are generated using Lighthouse Performance data.

Metric details ☒

First Contentful Paint

How quickly content like text or images are painted onto your page. A good user experience is 0.9s or less. [Learn more.](#)

Good - Nothing to do here

400ms

Time to Interactive

How long it takes for your page to become fully interactive. A good user experience is 2.5s or less. [Learn more.](#)

Good - Nothing to do here

2.4s

Speed Index

How quickly the contents of your page are visibly populated. A good user experience is 1.3s or less. [Learn more.](#)

Good - Nothing to do here

1.2s

Total Blocking Time

How much time is blocked by scripts during your page loading process. A good user experience is 150ms or less. [Learn more.](#)

Longer than recommended

349ms

Largest Contentful Paint

How long it takes for the largest element of content (e.g. a hero image) to be painted on your page. A good user experience is 1.2s or less. [Learn more.](#)

Good - Nothing to do here

400ms

Cumulative Layout Shift

How much your page's layout shifts as it loads. A good user experience is a score of 0.1 or less. [Learn more.](#)

OK, but consider improvement






0.13

To facilitate the visualization of the performance results, a file was created in PDF format and added with the name GTmetrix_report.pdf.

The Pentest Tool

The Pentest Tool performed an analysis of the application's security. The application showed a medium risk level. Since the Secure flag is not set on the cookie, the browser will send it over an unencrypted channel (plain HTTP) if such a request is made. Thus, the risk exists that an attacker will intercept the clear-text communication between the browser and the server and steal the user's cookie. If this is a session cookie, the attacker could gain unauthorized access to the victim's web session.

Vulnerabilities found for server-side software

 CVSS	 CVE	 SUMMARY	 EXPLOIT	 AFFECTED SOFTWARE
4.3	CVE-2019-11358	jQuery before 3.4.0, as used in Drupal, Backdrop CMS, and other products, mishandles jQuery.extend(true, {}, ...) because of Object.prototype pollution. If an unsanitized source object contained an enumerable __proto__ property, it could extend the native Object.prototype.	N/A	jQuery 3.3.1
4.3	CVE-2020-11022	In jQuery versions greater than or equal to 1.2 and before 3.5.0, passing HTML from untrusted sources - even after sanitizing it - to one of jQuery's DOM manipulation methods (i.e. .html(), .append(), and others) may execute untrusted code. This problem is patched in jQuery 3.5.0.	N/A	jQuery 3.3.1
4.3	CVE-2020-11023	In jQuery versions greater than or equal to 1.0.3 and before 3.5.0, passing HTML containing <option> elements from untrusted sources - even after sanitizing it - to one of jQuery's DOM manipulation methods (i.e. .html(), .append(), and others) may execute untrusted code. This problem is patched in jQuery 3.5.0.	N/A	jQuery 3.3.1

Risk description

These vulnerabilities expose the affected applications to the risk of unauthorized access to confidential data and possibly denial of service attacks. An attacker could search for an appropriate exploit (or create one himself) for any of these vulnerabilities and use it to attack the system.

Recommendation

We recommend you upgrade the affected software to the latest version to eliminate the risk of these vulnerabilities.

Priority:

1: The product should not ship without the successful resolution of the work item. The bug should be addressed as soon as possible.

2: The priority of the product should be resolved in the ordinary course of development activities. It can wait until a new build or version is created.

3: Resolution of the work item is optional based on resources, time, and risk.

Improvement opportunities

Improvement points and security recommendations were pointed out in the security scan report. The document with the name PentestTools_scan_report in PDF format was added, which helps with more information.