1 INTRODUCTION

This test plan is developed for testing the web application “epc-guys”, written in Java using the frameworks spring boot, spring security, spring jpa, spring web and MySQL database, as well as the bootstrap and JQuery frameworks.

The purpose of this testing is to check the operation of the application, its main functions and capabilities, user interface, as well as to ensure that this application meets the established requirements and ensure its high quality and reliability. The test plan includes a detailed description of the object of testing, risks, aspects, approaches to testing, development of test scenarios and evaluation of testing.

This testing will check the application for the following quality attributes:

- Functionality: the application must correctly perform all required functions and meet the customer’s specifications.
- Usability: the application should have a simple and intuitive interface that facilitates user interaction with the application and understanding of its work.
- Reliability: the application should be stable and not cause crashes, errors or data loss during normal and extreme use.
- Security: the application must protect the confidentiality and integrity of user data from unauthorized access, modification or deletion.

2 OBJECT OF TESTING

The developed web application is necessary for publishing various articles on the main topics: sports, IT, travel and science. The main functionality includes adding and deleting articles, searching for articles in the database. After adding articles, they are displayed in the main lists presented on the web application page.

This web application has the ability to register and authorize users. All information about the user, the article is stored in the database.

3 RISKS

To launch this web application on the server, you need the minimum system requirements:

- Java version not lower than 18;
- Operating system - Windows 10, 11;
- RAM - 2GB;
- ROM - 20MB.

For correct operation, access to the database is required. It is assumed that this application can be launched on various operating systems: Windows, Linux, MacOS, however, testing was carried out only on the Windows platform.

4 ASPECTS OF TESTING

The following application functionality will be tested:

- Attempt to log in with an incorrect login or password;

- Successful login attempt;
- Adding a new article;
- Deleting an existing article;
- Displaying all articles;
- Searching for an article by title.

5 APPROACHES TO TESTING

Testing will be performed only with the back-end of the application. Testing will take place in real time (that is, analyzing the data received by the application and comparing them with the expected results). Postman will be used for testing.

6 TESTING RESULTS

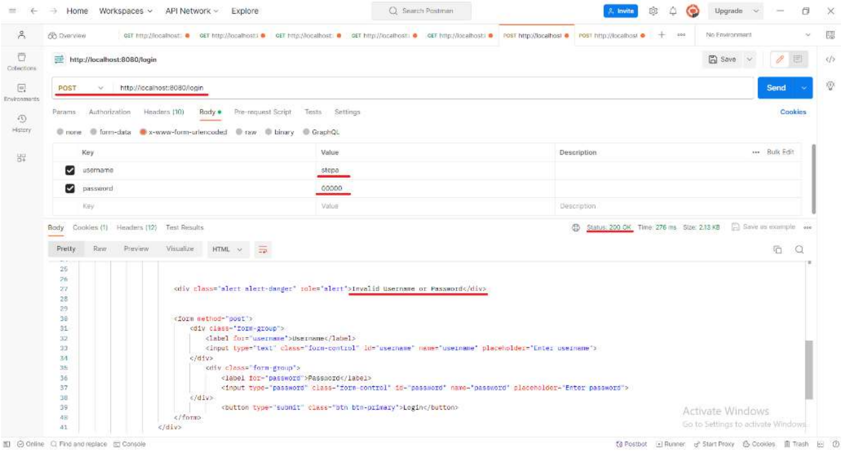
6.1 Attempt to log in with an incorrect login or password

Purpose: to check the security of the application.

Scenario: enter incorrect data in the “Username” and “Password” fields, press the “Send” button.

Expected result: authentication does not occur, an html code of the username and password entry page is issued with the message “Invalid Username or Password”.

Actual result: authentication does not occur, a form for entering a username and password appears with a message about unsuccessful authentication.



http://localhost:8080/login

POST http://localhost:8080/login

○ none ○ form-data ● x-www-form-urlencoded ○ raw ○ binary ○ GraphQL

	Key	Value	Description
☑	username	stepa	
☑	password	00000	
	Key	Value	Description

Body

Status: 200 OK Time: 276 ms Size: 2.13 KB

```html
        <div class="alert alert-danger" role="alert">Invalid Username or Password</div>


        <form method="post">
            <div class="form-group">
                <label for="username">Username</label>
                <input type="text" class="form-control" id="username" name="username" placeholder="Enter username">
            </div>
                <div class="form-group">
                    <label for="password">Password</label>
                    <input type="password" class="form-control" id="password" name="password" placeholder="Enter password">
            </div>
                    <button type="submit" class="btn btn-primary">Login</button>
        </form>
    </div>
```

Testing Evaluation: Successful.

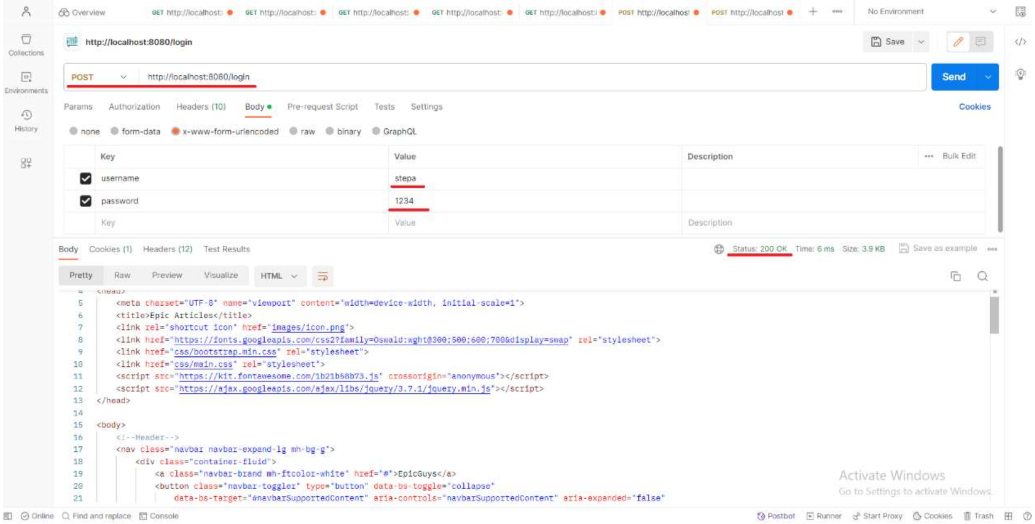
6.2 Successful Login Attempt

Purpose: To check the security of the application.

Scenario: Enter incorrect data in the “Username” and “Password” fields, press the “Send” button.

Expected result: Successful authentication, redirect to the guest page.

Actual result: Successful authentication, received html code of the guest page of the site.



POST http://localhost:8080/login

Body: x-www-form-urlencoded

	Key	Value	Description
☑	username	stepa	
☑	password	1234	

Status: 200 OK Time: 6 ms Size: 3.9 KB

```html
<head>
    <meta charset="UTF-8" name="viewport" content="width=device-width, initial-scale=1">
    <title>Epic Articles</title>
    <link rel="shortcut icon" href="images/icon.png">
    <link href="https://fonts.googleapis.com/css2?family=Oswald:wght@300;500;600;700&display=swap" rel="stylesheet">
    <link href="css/bootstrap.min.css" rel="stylesheet">
    <link href="css/main.css" rel="stylesheet">
    <script src="https://kit.fontawesome.com/1b21b58b73.js" crossorigin="anonymous"></script>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
</head>

<body>
    <!--Header-->
    <nav class="navbar navbar-expand-lg mh-bg-g">
        <div class="container-fluid">
            <a class="navbar-brand mh-ftcolor-white" href="#">EpicGuys</a>
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
                data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false"
```

Testing Evaluation: Successful.

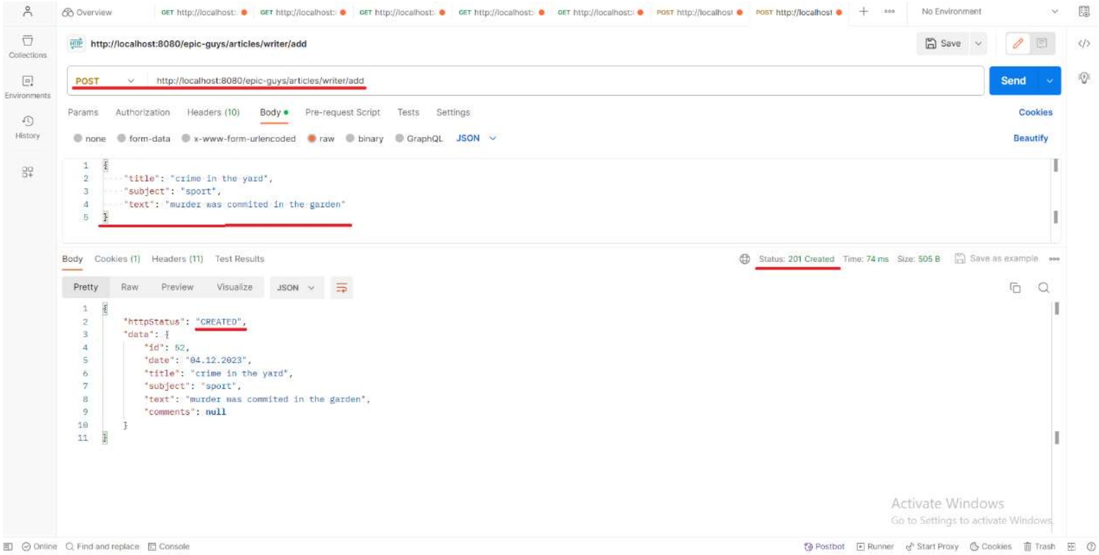
6.3 Adding a New Article

Purpose: To check the functionality of the web service.

Scenario: Perform a post request, passing in the body JSON information about the added article: title, topic, text of the article.

Expected result: Receiving JSON with status code 201, as well as an article object written to the database.

Actual result: The article is added to the database and the returned code is equal to 201.



POST http://localhost:8080/epic-guys/articles/writer/add

```json
{
    "title": "crime in the yard",
    "subject": "sport",
    "text": "murder was commited in the garden"
}
```

Status: 201 Created Time: 74 ms Size: 505 B

```json
{
    "httpStatus": "CREATED",
    "data": {
        "id": 52,
        "date": "04.12.2023",
        "title": "crime in the yard",
        "subject": "sport",
        "text": "murder was commited in the garden",
        "comments": null
    }
}
```

Testing Evaluation: Successful.

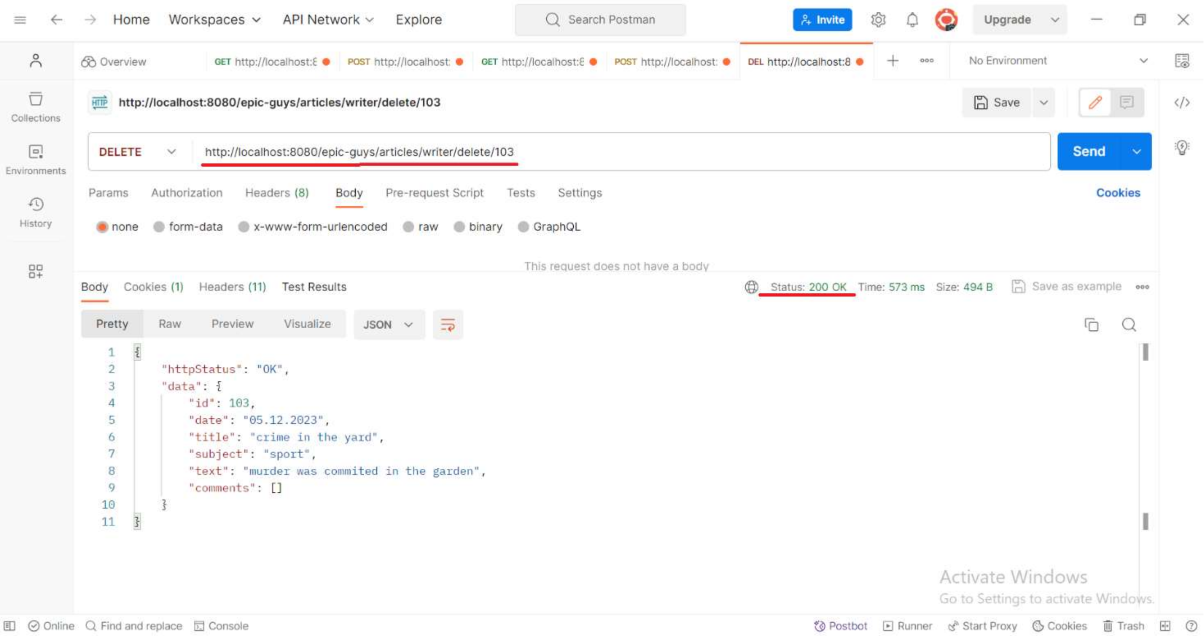
6.4 Deleting an Existing Article Purpose:

To check the functionality of the web service.

Scenario: Perform a delete request, passing as a parameter the id of the article to be deleted.

Expected result: Receiving JSON with status code 200, as well as the object that was deleted.

Actual result: Status code of operation 200 and returned object of deleted article.



http://localhost:8080/epic-guys/articles/writer/delete/103

DELETE http://localhost:8080/epic-guys/articles/writer/delete/103

Body: ● none ○ form-data ○ x-www-form-urlencoded ○ raw ○ binary ○ GraphQL

This request does not have a body

Status: 200 OK Time: 573 ms Size: 494 B

JSON

```
{
    "httpStatus": "OK",
    "data": {
        "id": 103,
        "date": "05.12.2023",
        "title": "crime in the yard",
        "subject": "sport",
        "text": "murder was commited in the garden",
        "comments": []
    }
}
```

Testing Evaluation: Successful.

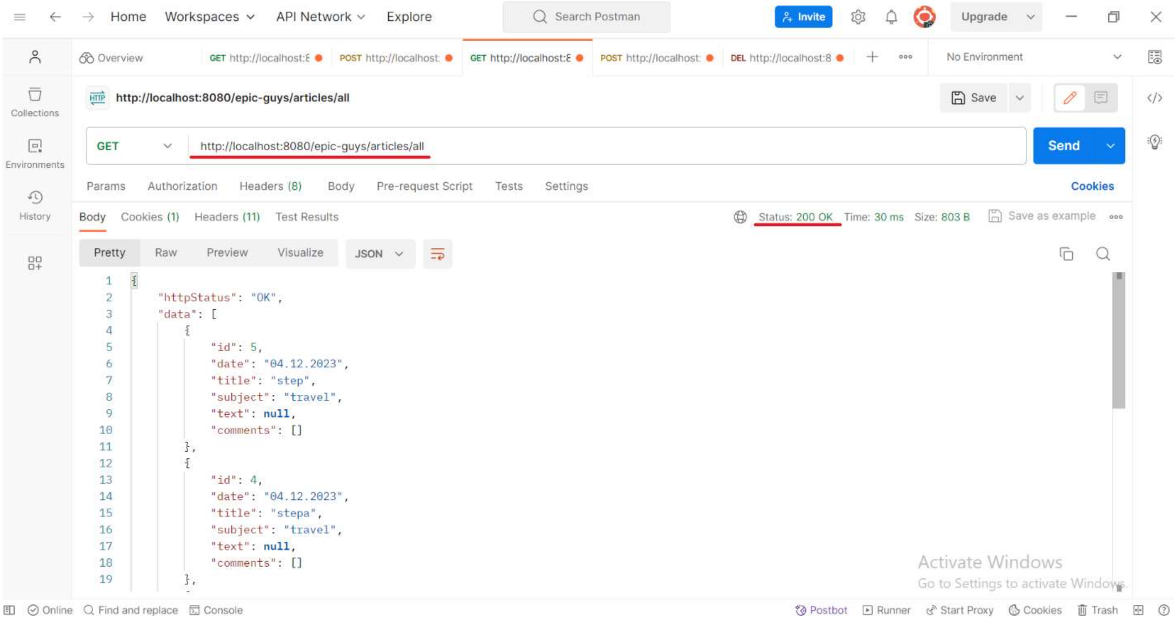
6.5 Displaying All Articles

Purpose: To check functionality.

Scenario: Perform a get request with a path equal to http://localhost:8080/epic-guys/articles/all

Expected result: Return JSON with a collection of all created articles from the database, request status code 200.

Actual result: The actual result matches the expected one.



http://localhost:8080/epic-guys/articles/all

GET http://localhost:8080/epic-guys/articles/all

Status: 200 OK Time: 30 ms Size: 803 B

```
{
    "httpStatus": "OK",
    "data": [
        {
            "id": 5,
            "date": "04.12.2023",
            "title": "step",
            "subject": "travel",
            "text": null,
            "comments": []
        },
        {
            "id": 4,
            "date": "04.12.2023",
            "title": "stepa",
            "subject": "travel",
            "text": null,
            "comments": []
        },
```

Testing Evaluation: Successful.

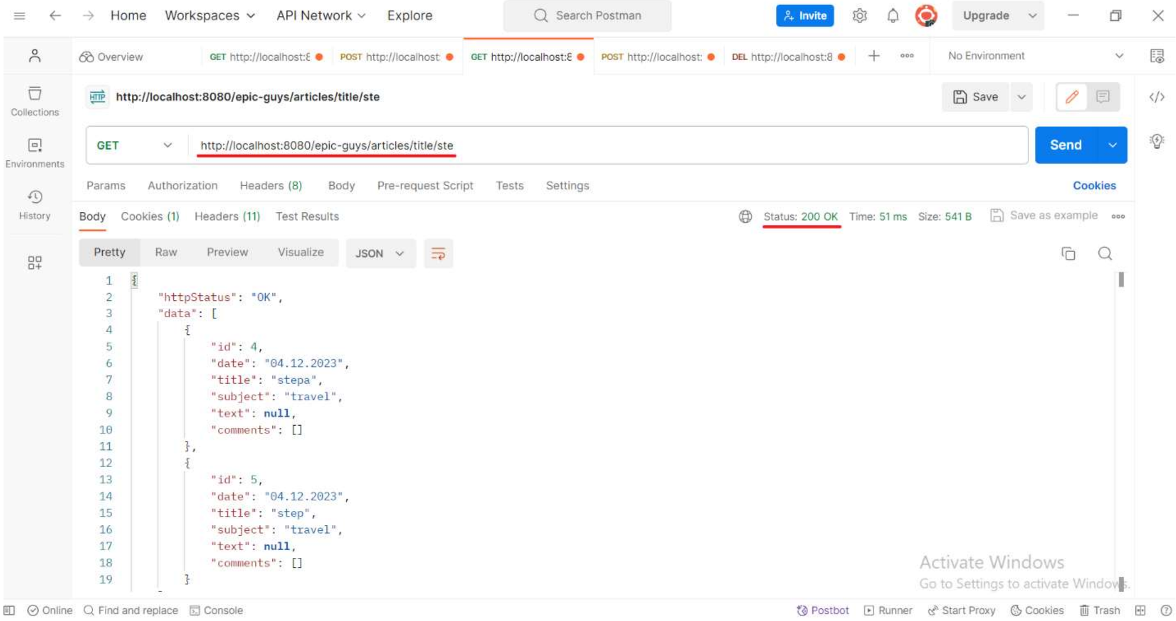
6.6 Searching for an Article by

Title Purpose: To check functionality.

Scenario: Perform a get request with a path equal to http://localhost:8080/epic-guys/articles/title/travel

Expected result: Return JSON with a collection of all articles matching the title search, request status code 200.

Actual result: The actual result matches the expected one.



http://localhost:8080/epic-guys/articles/title/ste

GET http://localhost:8080/epic-guys/articles/title/ste

Status: 200 OK Time: 51 ms Size: 541 B

```
{
    "httpStatus": "OK",
    "data": [
        {
            "id": 4,
            "date": "04.12.2023",
            "title": "stepa",
            "subject": "travel",
            "text": null,
            "comments": []
        },
        {
            "id": 5,
            "date": "04.12.2023",
            "title": "step",
            "subject": "travel",
            "text": null,
            "comments": []
        }
```

Testing Evaluation: Successful.

7 CONCLUSION

The developed web service has successfully passed all the presented tests, these tests have confirmed the quality attributes of the program: functionality, ease of use, reliability, security. Based on this, it can be considered that it meets all necessary requirements.