Testowanie i walidacja oprogramowania – Raport

Szymon Lepianka

# Testy jednostkowe

## Testy serwera

Przed każdym testem tworzony jest serwer MongoDB w pamięci (MongoMemoryServer) oraz tworzone połączenie z nim. Po każdym teście czyszczona jest baza danych. Po wszystkich testach zamykane jest połączenie do tymczasowej bazy danych.

### Dodawanie nowej listy zakupów

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik nie jest zalogowany. |
| Dane testowe | * --- |
| Procedura | 1. Wykonywany jest request do stworzenia nowej listy zakupów. |
| Warunki końcowe | * *response status code* równa się 401 (*Unauthorized*) * *response text* równa się „*invalid credentials*” |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. |
| Dane testowe | * Poprawna nazwa listy i poprawna data listy, przekazana w request body. |
| Procedura | 1. Wykonywany jest *request* do stworzenia nowej listy zakupów. W *request body* wysyłane są dane nowej listy zakupów. |
| Warunki końcowe | * *response status code* równa się 200 (*OK*) * *response body* równa się:  - *shoppingListId* == dowolny string  - *name* == przekazana nazwa w request body  - *date* == przekazana data w request body  - *completed* == false  - *userId* == id użytkownika wykonującego request |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Użytkownik nie posiada list zakupów. |
| Dane testowe | * Poprawna nazwa listy i pusta data listy, przekazana w request body. |
| Procedura | 1. Wykonywany jest *request* (POST) do stworzenia nowej listy zakupów. W *request body* wysyłane są dane nowej listy zakupów. 2. Wykonywany jest *request* (GET) do odczytu list zakupów użytkownika. |
| Warunki końcowe | * *POST:  - response status code* równa się 400 (*Bad Request*)  - *response text* równa się "*Missing date param*"  - *response body* jest puste * *GET:  - response status code* równa się 200 (*OK*)  - *response body* jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Użytkownik nie posiada list zakupów. |
| Dane testowe | * Pusta nazwa listy i poprawna data listy, przekazana w request body |
| Procedura | 1. Wykonywany jest *request* (POST) do stworzenia nowej listy zakupów. W *request body* wysyłane są dane nowej listy zakupów. 2. Wykonywany jest *request* (GET) do odczytu list zakupów użytkownika. |
| Warunki końcowe | * *POST:  - response status code* równa się 400 (*Bad Request*)  - *response text* równa się "*Missing name param*"  - *response body* jest puste * *GET:  - response status code* równa się 200 (*OK*)  - *response body* jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Użytkownik nie posiada list zakupów. |
| Dane testowe | * Poprawna nazwa listy i niepoprawna data listy, przekazana w request body. |
| Procedura | 1. Wykonywany jest *request* (POST) do stworzenia nowej listy zakupów. W *request body* wysyłane są dane nowej listy zakupów. 2. Wykonywany jest *request* (GET) do odczytu list zakupów użytkownika. |
| Warunki końcowe | * *POST:  - response status code* równa się 400 (*Bad Request*)  - *response text* równa się "*Bad date format (expected YYYY-MM-DD)*"  - *response body* jest puste * *GET:  - response status code* równa się 200 (*OK*)  - *response body* jest puste |

### Odczyt list zakupów użytkownika

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik nie jest zalogowany. |
| Dane testowe | * --- |
| Procedura | 1. Wykonywany jest request do odczytu list zakupów użytkownika. |
| Warunki końcowe | * *response status code* równa się 401 (*Unauthorized*) * *response text* równa się „*invalid credentials*” |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Użytkownik nie posiada list zakupów. |
| Dane testowe | * --- |
| Procedura | 1. Wykonywany jest *request* (GET) do odczytu list zakupów użytkownika. |
| Warunki końcowe | * *response status code* równa się 200 (*OK*) * wielkość *response body* == 0 * *response body* jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Użytkownik posiada listę zakupów. |
| Dane testowe | 1. Istniejące listy zakupów użytkownika, przy braku list innych użytkowników. 2. Istniejące listy zakupów użytkownika, przy posiadaniu list przez innych użytkowników. |
| Procedura | 1. Wykonywany jest *request* (GET) do odczytu list zakupów użytkownika. |
| Warunki końcowe | * *response status code* równa się 200 (*OK*) * wielkość *response body* == wielkości listy list zakupów użytkownika * *response body* == listy zakupów użytkownika |

### Aktualizacja listy zakupów

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik nie jest zalogowany. |
| Dane testowe | * --- |
| Procedura | 1. Wykonywany jest request do aktualizacji listy zakupów. |
| Warunki końcowe | * *response status code* równa się 401 (*Unauthorized*) * *response text* równa się „*invalid credentials*” |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Podany identyfikator listy zakupów nie istnieje. |
| Dane testowe | 1. Niepoprawne identyfikatory listy zakupów    * Identyfikator o poprawnym formacie, ale nieistniejący.    * Identyfikator o niepoprawnym formacie. |
| Procedura | 1. Wykonywany jest *request* (PUT) do aktualizacji listy zakupów użytkownika. |
| Warunki końcowe | * *response status code* równa się 400 (*Bad Request*) |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Podany identyfikator listy zakupów istnieje. * Użytkownik nie jest właścicielem listy zakupów. |
| Dane testowe | 1. Dane istniejącej listy zakupów innego użytkownika. |
| Procedura | 1. Wykonywany jest *request* (PUT) do aktualizacji listy zakupów innego użytkownika. |
| Warunki końcowe | * *response status code* równa się 401 (*Unauthorized*) * *response text* == „*Shopping list doesn't belong to this user*” |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Podany identyfikator listy zakupów istnieje. * Użytkownik jest właścicielem listy zakupów. |
| Dane testowe | * Dane istniejącej listy zakupów innego użytkownika. * Niepoprawne dane aktualizujące listę. |
| Procedura | 1. Wykonywany jest *request* (PUT) do aktualizacji listy zakupów użytkownika. |
| Warunki końcowe | * *response status code* równa się 400 (*Bad Request*) |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Podany identyfikator listy zakupów istnieje. * Użytkownik jest właścicielem listy zakupów. |
| Dane testowe | * Dane istniejącej listy zakupów użytkownika. * Poprawne dane aktualizujące listę. |
| Procedura | * Wykonywany jest *request* (PUT) do aktualizacji listy zakupów użytkownika. |
| Warunki końcowe | * *response status code* równa się 200 (*OK*) * *response body* == dana lista z zaktualizowanymi danymi |

### Usuwanie listy zakupów

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik nie jest zalogowany. |
| Dane testowe | * --- |
| Procedura | 1. Wykonywany jest request (DELETE) do usunięcia listy zakupów. |
| Warunki końcowe | * *response status code* równa się 401 (*Unauthorized*) * *response text* równa się „*invalid credentials*” |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Podany identyfikator listy zakupów nie istnieje. |
| Dane testowe | 1. Niepoprawne identyfikatory listy zakupów    * Identyfikator o poprawnym formacie, ale nieistniejący.    * Identyfikator o niepoprawnym formacie. |
| Procedura | 1. Wykonywany jest *request* (DELETE) do usunięcia listy zakupów użytkownika. |
| Warunki końcowe | * *response status code* równa się 400 (*Bad Request*) |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Podany identyfikator listy zakupów istnieje. * Użytkownik nie jest właścicielem listy zakupów. |
| Dane testowe | 1. Dane istniejącej listy zakupów innego użytkownika. |
| Procedura | 1. Wykonywany jest *request* (DELETE) do usunięcia listy zakupów innego użytkownika. |
| Warunki końcowe | * *response status code* równa się 401 (*Unauthorized*) * *response text* == „*Shopping list doesn't belong to this user*” |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Podany identyfikator listy zakupów istnieje. * Użytkownik jest właścicielem listy zakupów. |
| Dane testowe | 1. Dane istniejącej listy zakupów użytkownika. |
| Procedura | 1. Wykonywany jest *request* (DELETE) do usunięcia listy zakupów użytkownika.  2. Wykonywany jest *request* (GET) pobierający wszystkie listy zakupów użytkownika. |
| Warunki końcowe | * *response status code* (DELETE) równa się 200 (*OK*) * *response body* (DELETE) == dana lista zakupów * *response status code* (GET) równa się 200 (*OK*) * *response body* (GET) == [] |

### Stworzenie przedmiotu w liście zakupów

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik nie jest zalogowany. |
| Dane testowe | * --- |
| Procedura | 1. Wykonywany jest request do stworzenia nowej przedmiotu w liście zakupów. |
| Warunki końcowe | * *response status code* równa się 401 (*Unauthorized*) * *response text* równa się „*invalid credentials*” |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Lista zakupów istnieje. |
| Dane testowe | * Poprawka nazwa, poprawna ilość, poprawna gramatura oraz poprawne id listy zakupów |
| Procedura | 1. Wykonywany jest *request* (POST) dodający nowy przedmiot do listy zakupów. W *request body* wysyłane są dane nowego przedmiotu.  2. Wykonywany jest kontrolny *request* (GET) pobierający wszystkie przedmioty z danej listy zakupów. |
| Warunki końcowe | * *response status code* (POST) równa się 200 (*OK*) * *response body* (POST) równa się:  - *shoppingListId* == przekazane id listy zakupów  - *name* == przekazana nazwa w request body  - *amount* == przekazana ilość w request body  - *grammage* == przekazana gramatura w request body  - *completed* == false * *response status code* (GET) równa się 200 (*OK*) * długość *response body* (GET) == 1 |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Użytkownik posiada listę zakupów. |
| Dane testowe | * Brakująca nazwa, poprawna ilość, poprawna gramatura oraz poprawny id listy zakupów. |
| Procedura | 1. Wykonywany jest *request* (POST) dodający nowy przedmiot do listy zakupów. W *request body* wysyłane są dane nowego przedmiotu.  2. Wykonywany jest kontrolny *request* (GET) pobierający wszystkie przedmioty z danej listy zakupów. |
| Warunki końcowe | * *POST:  - response status code* równa się 400 (*Bad Request*)  - *response text* równa się "*Missing name param*"  - *response body* jest puste * *GET:  - response status code* równa się 200 (*OK*)  - *response body* jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Użytkownik posiada listę zakupów. |
| Dane testowe | * Poprawna nazwa, brakująca ilość, poprawna gramatura oraz poprawny id listy zakupów. |
| Procedura | 1. Wykonywany jest *request* (POST) dodający nowy przedmiot do listy zakupów. W *request body* wysyłane są dane nowego przedmiotu.  2. Wykonywany jest kontrolny *request* (GET) pobierający wszystkie przedmioty z danej listy zakupów. |
| Warunki końcowe | * *POST:  - response status code* równa się 400 (*Bad Request*)  - *response text* równa się "*Missing amount param*"  - *response body* jest puste * *GET:  - response status code* równa się 200 (*OK*)  - *response body* jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Użytkownik posiada listę zakupów. |
| Dane testowe | * Poprawna nazwa, poprawna ilość, brakująca gramatura oraz poprawny id listy zakupów. |
| Procedura | 1. Wykonywany jest *request* (POST) dodający nowy przedmiot do listy zakupów. W *request body* wysyłane są dane nowego przedmiotu.  2. Wykonywany jest kontrolny *request* (GET) pobierający wszystkie przedmioty z danej listy zakupów. |
| Warunki końcowe | * *POST:  - response status code* równa się 400 (*Bad Request*)  - *response text* równa się "*Missing grammage param*"  - *response body* jest puste * *GET:  - response status code* równa się 200 (*OK*)  - *response body* jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Użytkownik posiada listę zakupów. |
| Dane testowe | * Poprawna nazwa, poprawna ilość, poprawna gramatura oraz brakujący id listy zakupów. |
| Procedura | 1. Wykonywany jest *request* (POST) dodający nowy przedmiot do listy zakupów. W *request body* wysyłane są dane nowego przedmiotu.  2. Wykonywany jest kontrolny *request* (GET) pobierający wszystkie przedmioty z danej listy zakupów. |
| Warunki końcowe | * *POST:  - response status code* równa się 400 (*Bad Request*)  - *response text* równa się "*Missing shoppingListId param*"  - *response body* jest puste * *GET:  - response status code* równa się 200 (*OK*)  - *response body* jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Użytkownik nie posiada list zakupów. |
| Dane testowe | * Niepoprawny identyfikator listy zakupów   + Identyfikator o poprawnym formacie, ale nieistniejący.   + Identyfikator o niepoprawnym formacie. * Poprawne dane dodawanego przedmiotu. |
| Procedura | 1. Wykonywany jest *request* (POST) dodający nowy przedmiot do listy zakupów (nieistniejącej). W *request body* wysyłane są dane nowego przedmiotu. |
| Warunki końcowe | * *response status code* równa się 400 (*Bad Request*) * *response body* jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Istnieje lista, której właścicielem jest inny użytkownik. |
| Dane testowe | * Istniejąca lista zakupów innego użytkownika. * Poprawne dane dodawanego przedmiotu. |
| Procedura | 1. Wykonywany jest *request* (POST) dodający nowy przedmiot do listy zakupów (innego użytkownika). W *request body* wysyłane są dane nowego przedmiotu.  2. Wykonywany jest kontrolny *request* (GET) pobierający wszystkie przedmioty z danej listy zakupów (innego użytkownika). |
| Warunki końcowe | * *response status code* (POST) równa się 401 (*Unauthorized*) * *response text* (POST) równa „*Shopping list doesn't belong to this user*” * *response body* (POST) jest puste * *response status code* (GET) równa się 200 (*OK*) * *response body* (GET) zawiera jedynie prawidłowe przedmioty |

### Odczyt przedmiotów danej listy zakupów

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik nie jest zalogowany. |
| Dane testowe | * --- |
| Procedura | 1. Wykonywany jest request do odczytu przedmiotów list zakupów użytkownika. |
| Warunki końcowe | * *response status code* równa się 401 (*Unauthorized*) * *response text* równa się „*invalid credentials*” |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Użytkownik nie posiada list zakupów. |
| Dane testowe | * Niepoprawny identyfikator listy zakupów   + Identyfikator o poprawnym formacie, ale nieistniejący.   + Identyfikator o niepoprawnym formacie. |
| Procedura | 1. Wykonywany jest *request* (GET) do odczytu przedmiotów danej listy zakupów użytkownika. |
| Warunki końcowe | * *response status code* równa się 400 (*Bad* *Request*) * *response text* równa się „Shopping list doesn't exist” |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Użytkownik posiada listę zakupów. |
| Dane testowe | * Istniejące przedmioty na liście zakupów. * Istniejąca lista zakupów. |
| Procedura | 1. Wykonywany jest *request* (GET) do odczytu przedmiotów danej listy zakupów użytkownika. |
| Warunki końcowe | * *response status code* równa się 200 (*OK*) * wielkość *response body* == liczba przedmiotów na liście zakupów użytkownika * *response body* == przedmioty dodane do listy zakupów |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Inny użytkownik posiada listę zakupów. |
| Dane testowe | * Istniejąca lista zakupów innego użytkownika. * Istniejący przedmiot na liście zakupów innego użytkownika. |
| Procedura | 1. Wykonywany jest *request* (GET) do odczytu przedmiotów danej listy zakupów użytkownika. |
| Warunki końcowe | * *response status code* równa się 401 (*Unauthorized*) * *response text* == „*Shopping list doesn't belong to this user*” * *response body* jest puste |

### Aktualizacja przedmiotu na liście zakupów

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik nie jest zalogowany. |
| Dane testowe | * --- |
| Procedura | 1. Wykonywany jest request do aktualizacji przedmiotu na liście zakupów. |
| Warunki końcowe | * *response status code* równa się 401 (*Unauthorized*) * *response text* równa się „*invalid credentials*” |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Podany identyfikator przedmiotu nie istnieje. |
| Dane testowe | * Niepoprawny identyfikator listy zakupów   + Identyfikator o poprawnym formacie, ale nieistniejący.   + Identyfikator o niepoprawnym formacie. |
| Procedura | 1. Wykonywany jest *request* (PUT) do aktualizacji przedmiotu (który nie istnieje) na liście zakupów użytkownika. |
| Warunki końcowe | * *response status code* równa się 400 (*Bad Request*) |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Podany identyfikator przedmiotu istnieje. * Użytkownik nie jest właścicielem listy zakupów. |
| Dane testowe | * Istniejąca lista zakupów innego użytkownika. * Istniejący przedmiot na liście zakupów innego użytkownika. |
| Procedura | 1. Wykonywany jest *request* (PUT) do aktualizacji listy zakupów innego użytkownika. |
| Warunki końcowe | * *response status code* równa się 401 (*Unauthorized*) * *response text* == „*Shopping list doesn't belong to this user*” |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Podany identyfikator przedmiotu istnieje. * Użytkownik jest właścicielem listy zakupów. |
| Dane testowe | * Istniejąca lista zakupów użytkownika. * Istniejący przedmiot na liście zakupów użytkownika. * Poprawne dane aktualizujące przedmiot. |
| Procedura | 1. Wykonywany jest *request* (PUT) do aktualizacji listy zakupów użytkownika. |
| Warunki końcowe | * *response status code* równa się 200 (*OK*) * *response body* == dana lista z zaktualizowanymi danymi |

### Usuwanie przedmiotu z listy zakupów

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik nie jest zalogowany. |
| Dane testowe | * --- |
| Procedura | 1. Wykonywany jest request (DELETE) do usunięcia przedmiotu z listy zakupów. |
| Warunki końcowe | * *response status code* równa się 401 (*Unauthorized*) * *response text* równa się „*invalid credentials*” |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Podany identyfikator przedmiotu nie istnieje. |
| Dane testowe | * Niepoprawny identyfikator listy zakupów   + Identyfikator o poprawnym formacie, ale nieistniejący.   + Identyfikator o niepoprawnym formacie. |
| Procedura | 1. Wykonywany jest *request* (DELETE) do usunięcia przedmiotu z listy zakupów użytkownika. |
| Warunki końcowe | * *response status code* równa się 400 (*Bad Request*) |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Podany identyfikator przedmiotu istnieje. * Użytkownik nie jest właścicielem listy zakupów. |
| Dane testowe | * Istniejąca lista zakupów innego użytkownika. * Istniejący przedmiot na liście zakupów innego użytkownika. |
| Procedura | 1. Wykonywany jest *request* (DELETE) do usunięcia przedmiotu z listy zakupów innego użytkownika. |
| Warunki końcowe | * *response status code* równa się 401 (*Unauthorized*) * *response text* == „*Shopping list doesn't belong to this user*” |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik jest zalogowany. * Podany identyfikator przedmiotu istnieje. * Użytkownik jest właścicielem listy zakupów. |
| Dane testowe | * Istniejąca lista zakupów użytkownika. * Przedmiot na liście zakupów użytkownika. * Dane aktualizujące przedmiot. |
| Procedura | 1. Wykonywany jest *request* (DELETE) do usunięcia przedmiotu z listy zakupów użytkownika.  2. Wykonywany jest *request* (GET) pobierający wszystkie przedmioty z listy zakupów użytkownika. |
| Warunki końcowe | * *response status code* (DELETE) równa się 200 (*OK*) * *response body* (DELETE) == dany przedmiot * *response status code* (GET) równa się 200 (*OK*) * *response body* (GET) == [] |

### Logowanie użytkownika

|  |  |
| --- | --- |
| Warunki początkowe | * Istnieje użytkownik o danych testowych. |
| Dane testowe | * Pusta nazwa i poprawne hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* równa się 400 (*Bad Request*) * *response text* == „*Empty username*” * *response body* jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | * Istnieje użytkownik o danych testowych. |
| Dane testowe | * Brakująca nazwa oraz poprawne hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* równa się 400 (*Bad Request*) * *response text* == „*Empty username*” * *response body* jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | * Istnieje użytkownik o danych testowych. |
| Dane testowe | * Brakująca nazwa i brakujące hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* równa się 400 (*Bad Request*) * *response body* jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | * Istnieje użytkownik o danych testowych. |
| Dane testowe | * Poprawna nazwa i puste hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* równa się 400 (*Bad Request*) * *response text* == „*Empty password*” * *response body* jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | * Istnieje użytkownik o danych testowych. |
| Dane testowe | * Pusta nazwa i puste hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* równa się 400 (*Bad Request*) * *response body* jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | * Istnieje użytkownik o danych testowych. |
| Dane testowe | * Poprawna nazwa i brakujące hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* równa się 400 (*Bad Request*) * *response text* == „*Empty password*” * *response body* jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | * Istnieje użytkownik o danych testowych. |
| Dane testowe | * Poprawna nazwa i poprawne hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* równa się 200 (*OK*) * przekazany zostaje *access token* (*string*) * weryfikacja tokenu jest poprawna |

|  |  |
| --- | --- |
| Warunki początkowe | * Nazwa użytkownika jest poprawna. * Hasło jest niepoprawne. |
| Dane testowe | * Poprawna nazwa i niepoprawne hasło   + Puste hasło   + Za krótkie hasło.   + Niepoprawne hasło.   + Hasło zawierające niepoprawne znaki.   + Za długie hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* nie równa się 200 (*OK*) |

|  |  |
| --- | --- |
| Warunki początkowe | * Nazwa użytkownika jest niepoprawna. * Hasło jest poprawne. |
| Dane testowe | * Niepoprawna nazwa i poprawne hasło   + Pusta nazwa.   + Za krótka nazwa.   + Niepoprawna nazwa.   + Nazwa zawierająca niepoprawne znaki.   + Za długa nazwa. |
| Procedura | 1. Wykonywany jest *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* nie równa się 200 (*OK*) |

|  |  |
| --- | --- |
| Warunki początkowe | * Nazwa użytkownika jest niepoprawna. * Hasło jest niepoprawne. |
| Dane testowe | * Niepoprawna nazwa i niepoprawne hasło   + Pusta nazwa i hasło.   + Za krótka nazwa i hasło.   + Niepoprawna nazwa i hasło.   + Nazwa i hasło zawierające niepoprawne znaki.   + Za długa nazwa i hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* nie równa się 200 (*OK*) |

### Rejestracja użytkownika

|  |  |
| --- | --- |
| Warunki początkowe | Brak. |
| Dane testowe | * Pusta nazwa i poprawne hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do rejestrowania użytkownika.  2. Wykonywany jest kontrolny *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* (REGISTER) równa się 400 (*Bad Request*) * *response text* (REGISTER) równa się „*Empty username*” * *response status code* (LOGIN) równa się 400 (*Bad Request*) * *response text* (LOGIN) równa się „*Empty username*” * *response body* (LOGIN) jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | Brak. |
| Dane testowe | * Brakująca nazwa i poprawne hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do rejestrowania użytkownika.  2. Wykonywany jest kontrolny *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* (REGISTER) równa się 400 (*Bad Request*) * *response text* (REGISTER) równa się „*Empty username*” * *response status code* (LOGIN) równa się 400 (*Bad Request*) * *response text* (LOGIN) równa się „*Empty username*” * *response body* (LOGIN) jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | Brak. |
| Dane testowe | * Brakująca nazwa i brakujące hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do rejestrowania użytkownika.  2. Wykonywany jest kontrolny *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* (REGISTER) równa się 400 (*Bad Request*) * *response text* (REGISTER) równa się „*Empty username*” * *response status code* (LOGIN) równa się 400 (*Bad Request*) * *response text* (LOGIN) równa się „*Empty username*” * *response body* (LOGIN) jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | Brak. |
| Dane testowe | * Poprawna nazwa i puste hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do rejestrowania użytkownika.  2. Wykonywany jest kontrolny *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* (REGISTER) równa się 400 (*Bad Request*) * *response text* (REGISTER) równa się „*Empty password*” * *response status code* (LOGIN) równa się 400 (*Bad Request*) * *response text* (LOGIN) równa się „*Empty password*” * *response body* (LOGIN) jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | Brak. |
| Dane testowe | * Pusta nazwa i puste hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do rejestrowania użytkownika.  2. Wykonywany jest kontrolny *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* (REGISTER) równa się 400 (*Bad Request*) * *response text* (REGISTER) równa się „*Empty username*” * *response status code* (LOGIN) równa się 400 (*Bad Request*) * *response text* (LOGIN) równa się „*Empty username*” * *response body* (LOGIN) jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | Brak. |
| Dane testowe | * Poprawna nazwa i brakujące hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do rejestrowania użytkownika.  2. Wykonywany jest kontrolny *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* (REGISTER) równa się 400 (*Bad Request*) * *response text* (REGISTER) równa się „*Empty password*” * *response status code* (LOGIN) równa się 400 (*Bad Request*) * *response text* (LOGIN) równa się „*Empty password*” * *response body* (LOGIN) jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | Brak. |
| Dane testowe | * Poprawna nazwa i poprawne hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do rejestrowania użytkownika.  2. Wykonywany jest kontrolny *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* (REGISTER) równa się 201 (*Created*) * *response text* (REGISTER) równa się „*Created!*” * *response status code* (LOGIN) równa się 200 (*OK*) * przekazany zostaje *access token* (*string*) * weryfikacja tokenu jest poprawna |

|  |  |
| --- | --- |
| Warunki początkowe | Brak. |
| Dane testowe | * Poprawna nazwa i niepoprawne hasło.   + Puste hasło.   + Za krótkie hasło.   + Hasło z nielegalnymi znakami.   + Za długie hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do rejestrowania użytkownika.  2. Wykonywany jest kontrolny *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* (REGISTER) nie równa się 201 (*Created*) * *response status code* (LOGIN) nie równa się 200 (*OK*) * *response body* (LOGIN) jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | Brak. |
| Dane testowe | * Niepoprawna nazwa i poprawne hasło.   + Pusta nazwa.   + Za krótka nazwa.   + Nazwa z nielegalnymi znakami.   + Za długa nazwa. |
| Procedura | 1. Wykonywany jest *request* (POST) do rejestrowania użytkownika.  2. Wykonywany jest kontrolny *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* (REGISTER) nie równa się 201 (*Created*) * *response status code* (LOGIN) nie równa się 200 (*OK*) * *response body* (LOGIN) jest puste |

|  |  |
| --- | --- |
| Warunki początkowe | Brak. |
| Dane testowe | * Niepoprawna nazwa i niepoprawne hasło.   + Pusta nazwa i hasło.   + Za krótka nazwa i hasło.   + Nazwa z nielegalnymi znakami i hasło.   + Za długa nazwa i hasło. |
| Procedura | 1. Wykonywany jest *request* (POST) do rejestrowania użytkownika.  2. Wykonywany jest kontrolny *request* (POST) do zalogowania użytkownika. |
| Warunki końcowe | * *response status code* (REGISTER) nie równa się 201 (*Created*) * *response status code* (LOGIN) nie równa się 200 (*OK*) * *response body* (LOGIN) jest puste |

## Testy klienta

### Strona logowania

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik nie jest zalogowany. * Sprawdzanie renderowania strony. |
| Dane testowe | --- |
| Procedura | 1. Renderowana jest strona logowania (*LoginPage*). |
| Warunki końcowe | * Na ekranie znajduje się:   + Etykieta z tytułem strony   + Etykieta przy inputcie nazwy użytkownika   + Etykieta przy inputcie hasła   + Input nazwy użytkownika   + Input hasła   + Przycisk do zatwierdzenia logowania * Na ekranie nie znajduje się:   + Etykieta z informacją o błędzie |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik nie jest zalogowany. * Sprawdzanie zachowania formatki logowania w przypadku niepoprawnych danych logowania. |
| Dane testowe | * Nie poprawne dane logowania (nazwa użytkownika i hasło):   + Pusta nazwa i puste hasło.   + Poprawna nazwa i puste hasło.   + Pusta nazwa i poprawne hasło.   + Za krótka nazwa.   + Za krótkie hasło.   + Nazwa z nielegalnymi znakami.   + Hasło z nielegalnymi znakami.   + Nazwa i hasło z nielegalnymi znakami.   + Za długa nazwa.   + Za długie hasło.   + Za długa nazwa i hasło. |
| Procedura | 1. Renderowana jest strona logowania (*LoginPage*).  2. Uzupełniane są dane logowania (nazwa użytkownika i hasło).  3. Klikany jest przycisk zatwierdzający logowanie. |
| Warunki końcowe | * Przed kliknięciem na ekranie nie znajduje się etykieta z wiadomością błędu. * Po kliknięciu na ekranie znajduje się etykieta z wiadomością błędu. * Treść błędu jest odpowiednia (zgodna z przekazaną). |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik nie jest zalogowany. * Sprawdzanie braku błędu po zatwierdzeniu formatki logowania (przy wpisaniu danych). |
| Dane testowe | 1. Poprawne dane logowania (nazwa użytkownika i hasło). |
| Procedura | 1. Renderowana jest strona logowania (*LoginPage*).  2. Uzupełniane są dane logowania (nazwa użytkownika i hasło).  3. Klikany jest przycisk zatwierdzający logowanie. |
| Warunki końcowe | * Przed kliknięciem na ekranie nie znajduje się etykieta z wiadomością błędu. * Po kliknięciu na ekranie nie znajduje się tekst „*Username and password required!”.* * Po kliknięciu na ekranie nie znajduje się etykieta z wiadomością błędu. |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik nie jest zalogowany. * Sprawdzanie poprawnego zalogowania i przekierowania w przypadku poprawnego zalogowania. |
| Dane testowe | 1. Poprawne dane logowania (nazwa użytkownika i hasło). |
| Procedura | 1. Renderowana jest strona logowania (*LoginPage*).  2. Uzupełniane są dane logowania (nazwa użytkownika i hasło).  3. Klikany jest przycisk zatwierdzający logowanie. |
| Warunki końcowe | * Przekierowanie zostało wykonane raz. * Przekierowanie zostało wykonane na stronę „/”. |

### Strona rejestracji

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik nie jest zarejestrowany. * Sprawdzanie renderowania strony. |
| Dane testowe | --- |
| Procedura | 1. Renderowana jest strona rejestracji (*RegisterPage*). |
| Warunki końcowe | * Na ekranie znajduje się:   + Etykieta z tytułem strony   + Etykieta przy inputcie nazwy użytkownika   + Etykieta przy inputcie hasła   + Input nazwy użytkownika   + Input hasła   + Przycisk do zatwierdzenia rejestracji * Na ekranie nie znajduje się:   + Etykieta z informacją o błędzie |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik nie jest zarejestrowany. * Sprawdzanie zachowania formatki rejestracji w przypadku niepoprawnych danych rejestracji. |
| Dane testowe | * Nie poprawne dane rejestracji (nazwa użytkownika i hasło):   + Pusta nazwa i puste hasło.   + Poprawna nazwa i puste hasło.   + Pusta nazwa i poprawne hasło.   + Za krótka nazwa.   + Za krótkie hasło.   + Nazwa z nielegalnymi znakami.   + Hasło z nielegalnymi znakami.   + Nazwa i hasło z nielegalnymi znakami.   + Za długa nazwa.   + Za długie hasło.   + Za długa nazwa i hasło. |
| Procedura | 1. Renderowana jest strona logowania (*RegisterPage*).  2. Uzupełniane są dane rejestracji (nazwa użytkownika i hasło).  3. Klikany jest przycisk zatwierdzający rejestrację. |
| Warunki końcowe | * Przed kliknięciem na ekranie nie znajduje się etykieta z wiadomością błędu. * Po kliknięciu na ekranie znajduje się etykieta z wiadomością błędu. * Treść błędu jest odpowiednia (zgodna z przekazaną). |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik nie jest zarejestrowany. * Sprawdzanie braku błędu po zatwierdzeniu formatki rejestracji (przy wpisaniu danych). |
| Dane testowe | 1. Poprawne dane rejestracji (nazwa użytkownika i hasło). |
| Procedura | 1. Renderowana jest strona rejestracji (*RegisterPage*).  2. Uzupełniane są dane rejestracji (nazwa użytkownika i hasło).  3. Klikany jest przycisk zatwierdzający rejestrację. |
| Warunki końcowe | * Przed kliknięciem na ekranie nie znajduje się etykieta z wiadomością błędu. * Po kliknięciu na ekranie nie znajduje się tekst „*Username and password required!”.* * Po kliknięciu na ekranie nie znajduje się etykieta z wiadomością błędu. |

|  |  |
| --- | --- |
| Warunki początkowe | * Użytkownik nie jest zarejestrowany. * Sprawdzanie poprawnej rejestracji i przekierowania w przypadku poprawnej rejestracji. |
| Dane testowe | 1. Poprawne dane rejestracji (nazwa użytkownika i hasło). |
| Procedura | 1. Renderowana jest strona rejestracji (*RegisterPage*).  2. Uzupełniane są dane rejestracji (nazwa użytkownika i hasło).  3. Klikany jest przycisk zatwierdzający rejestrację. |
| Warunki końcowe | * Przekierowanie zostało wykonane raz. * Przekierowanie zostało wykonane na stronę „*/login*”. |

### Komponent CreateShoppingListForm

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie renderowania komponentu. |
| Dane testowe | --- |
| Procedura | 1. Renderowany jest komponent do tworzenia listy zakupów (*CreateShoppingListForm*). |
| Warunki końcowe | * Na ekranie znajduje się:   + Etykieta przy nazwie listy zakupów.   + Etykieta przy dacie listy zakupów.   + Input nazwy listy zakupów.   + Input daty listy zakupów.   + Przycisk do zatwierdzenia tworzenia listy zakupów. |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie zachowania formatki tworzenia listy zakupów w przypadku niepoprawnych danych. |
| Dane testowe | * Dane listy zakupów (nazwa i data):   + Pusta nazwa i pusta data.   + Poprawna nazwa i pusta data.   + Pusta nazwa i poprawna data.   + Za krótka nazwa.   + Nazwa z nielegalnymi znakami.   + Data z nielegalnymi znakami.   + Nazwa i data z nielegalnymi znakami.   + Za długa nazwa.   + Za długie data.   + Za długa nazwa i data. |
| Procedura | 1. Renderowany jest komponent do tworzenia listy zakupów (*CreateShoppingListForm*).  2. Uzupełniane są dane listy zakupów (nazwa i data).  3. Klikany jest przycisk zatwierdzający tworzenie listy zakupów. |
| Warunki końcowe | * Request na serwer nie został wykonany (dane są niepoprawne). |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie zachowania formatki tworzenia listy zakupów w przypadku poprawnych danych. |
| Dane testowe | 1. Poprawne dane listy zakupów (nazwa i data). |
| Procedura | 1. Renderowany jest komponent do tworzenia listy zakupów (*CreateShoppingListForm*).  2. Uzupełniane są dane listy zakupów (nazwa i data).  3. Klikany jest przycisk zatwierdzający tworzenie listy zakupów. |
| Warunki końcowe | * Został wykonany jeden request (*axios*). * Request został wykonany na odpowiedni *endpoint* – lista zakupów została stworzona. |

### Komponent CreateTodoForm

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie renderowania komponentu. |
| Dane testowe | --- |
| Procedura | 1. Renderowany jest komponent do tworzenia przedmiotu na liście zakupów (*CreateTodoForm*). |
| Warunki końcowe | * Na ekranie znajduje się:   + Etykieta przy nazwie przedmiotu.   + Etykieta przy ilości przedmiotu.   + Etykieta przy gramaturze przedmiotu.   + Input nazwy przedmiotu.   + Input ilości przedmiotu.   + Input gramatury przedmiotu.   + Przycisk do zatwierdzenia tworzenia przedmiotu. |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie zachowania formatki tworzenia przedmiotu w przypadku niepoprawnych danych. |
| Dane testowe | * Dane przedmiotu (nazwa, ilość, gramatura):   + Pusta nazwa, pusta ilość, pusta gramatura.   + Pusta nazwa, pusta ilość.   + Pusta ilość, pusta gramatura.   + Pusta nazwa, pusta gramatura.   + Pusta nazwa.   + Pusta ilość.   + Pusta gramatura.   + Poprawna nazwa i pusta data.   + Pusta nazwa i poprawna data.   + Za krótka nazwa.   + Nazwa z nielegalnymi znakami.   + Ilość niebędąca liczbą.   + Niepoprawna gramatura.   + Za długa nazwa.   + Za długa ilość.   + Za długa gramatura. |
| Procedura | 1. Renderowany jest komponent do tworzenia listy zakupów (*CreateTodoForm*).  2. Uzupełniane są dane przedmiotu (nazwa, ilość i gramatura).  3. Klikany jest przycisk zatwierdzający tworzenie przedmiotu. |
| Warunki końcowe | * Request na serwer nie został wykonany (dane są niepoprawne). |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie zachowania formatki tworzenia przedmiotu w przypadku poprawnych danych. |
| Dane testowe | 1. Poprawne dane przedmiotu (nazwa, ilość i gramatura). |
| Procedura | 1. Renderowany jest komponent do tworzenia przedmiotu (*CreateTodoForm*).  2. Uzupełniane są dane przedmiotu (nazwa, ilość i gramatura).  3. Klikany jest przycisk zatwierdzający tworzenie przedmiotu. |
| Warunki końcowe | * Został wykonany jeden request (*axios*). * Request został wykonany na odpowiedni *endpoint* – przedmiot został stworzony. |

### Komponent ShoppingList

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie renderowania komponentu. |
| Dane testowe | --- |
| Procedura | 1. Renderowany jest komponent do wyświetlania i edycji danych listy zakupów (*ShoppingList*). |
| Warunki końcowe | * Na ekranie znajduje się:   + Etykieta przy nazwie listy zakupów.   + Etykieta przy dacie listy zakupów.   + Etykieta daty listy zakupów.   + Input nazwy listy zakupów.   + Input wykonania listy zakupów.   + Przycisk do usunięcia listy zakupów.   + Przycisk do przekierowania do szczegółów listy zakupów. |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie zachowania zaznaczenia checkboxu o wykonaniu danej listy zakupów. |
| Dane testowe | 1. Poprawne dane listy zakupów. |
| Procedura | 1. Renderowany jest komponent do wyświetlania i edycji danych listy zakupów (*ShoppingList*).  2. Stan zaznaczenia checkboxa jest prawidłowy.  3. Klikany jest checkbox (wykonanie listy zakupów). |
| Warunki końcowe | * Został wykonany jeden request (*axios*). * Request został wykonany na odpowiedni *endpoint* – lista zakupów została ukończona. * Zwrócona została informacja o zmienieniu stanu listy zakupów. |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie zmiany nazwy listy zakupów przy wprowadzeniu niepoprawnych danych. |
| Dane testowe | * Niepoprawna nazwa listy zakupów   + Pusta nazwa.   + Za krótka nazwa.   + Nazwa z nielegalnymi znakami.   + Za długa nazwa. |
| Procedura | 1. Renderowany jest komponent do wyświetlania i edycji danych listy zakupów (*ShoppingList*).  2. Input z nazwą listy zakupów jest uzupełniany niepoprawnymi danymi. |
| Warunki końcowe | * Nie został wykonany request (*axios*) na serwer. |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie zmiany nazwy listy zakupów przy wprowadzeniu poprawnych danych. |
| Dane testowe | 1. Poprawna nazwa listy zakupów. |
| Procedura | 1. Renderowany jest komponent do wyświetlania i edycji danych listy zakupów (*ShoppingList*).  2. Input z nazwą listy zakupów jest uzupełniany poprawnymi danymi. |
| Warunki końcowe | * Został wykonany jeden request (*axios*). * Request został wykonany na odpowiedni *endpoint* – nazwa listy zakupów została zaktualizowana. |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie poprawności usunięcia listy zakupów. |
| Dane testowe | 1. Dane istniejącej lista zakupów. |
| Procedura | 1. Renderowany jest komponent do wyświetlania i edycji danych listy zakupów (*ShoppingList*).  2. Klikany jest przycisk do usunięcia listy zakupów. |
| Warunki końcowe | * Został wykonany jeden request (*axios*). * Request został wykonany na odpowiedni *endpoint* –listy zakupów została usunięta. |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie poprawności przekierowania na stronę ze szczegółami listy zakupów po kliknięciu przycisku. |
| Dane testowe | 1. Dane istniejącej lista zakupów. |
| Procedura | 1. Renderowany jest komponent do wyświetlania i edycji danych listy zakupów (*ShoppingList*).  2. Klikany jest przycisk do przekierowania na stronę ze szczegółami listy zakupów. |
| Warunki końcowe | * Przekierowanie zostało wykonane raz. * Przekierowanie zostało wykonane na stronę „/*details*”. |

### Komponent *TodoItem*

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie renderowania komponentu oraz statusu ukończenia przedmiotu. |
| Dane testowe | 1. Dane istniejącego przedmiotu. |
| Procedura | 1. Renderowany jest komponent do wyświetlania i edycji danych przedmiotu (*TodoItem*). |
| Warunki końcowe | * Na ekranie znajduje się:   + Nieprzekreślona etykieta nazwy przedmiotu.   + Etykieta ilości przedmiotu.   + Etykieta gramatury przedmiotu.   + Input (checkbox) wykonania przedmiotu (niewykonany).   + Przycisk do usunięcia przedmiotu. |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie renderowania komponentu przedmiotu. * Sprawdzanie przekreślenia nazwy przedmiotu przy zaznaczonym checkboxie. |
| Dane testowe | 1. Dane istniejącego przedmiotu. |
| Procedura | 1. Renderowany jest komponent do wyświetlania i edycji danych przedmiotu (*TodoItem*).  2. Stan zaznaczenia checkboxa jest prawidłowy. |
| Warunki końcowe | * Na ekranie znajduje się przekreślona etykieta nazwy przedmiotu. |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie aktualizacji ukończenia przedmiotu po kliknięciu checkboxa. |
| Dane testowe | 1. Dane istniejącego przedmiotu. |
| Procedura | 1. Renderowany jest komponent do wyświetlania i edycji danych przedmiotu (*TodoItem*).  2. Stan zaznaczenia checkboxa jest prawidłowy.  3. Klikany jest checkbox. |
| Warunki końcowe | * Został wykonany jeden request (*axios*). * Request został wykonany na odpowiedni *endpoint* – przedmiot został zaktualizowany. |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie poprawności usunięcia przedmiotu. |
| Dane testowe | 1. Dane istniejącego przedmiotu. |
| Procedura | 1. Renderowany jest komponent do wyświetlania i edycji danych przedmiotu (*TodoItem*).  2. Klikany jest przycisk do usunięcia przedmiotu. |
| Warunki końcowe | * Został wykonany jeden request (*axios*). * Request został wykonany na odpowiedni *endpoint* – przedmiot został usunięty. |

### Strona ShoppingListPage

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie renderowania strony. * Sprawdzanie renderowania tytułu, spinnera oraz formularza podczas ładowania danych. |
| Dane testowe | --- |
| Procedura | 1. Renderowana jest strona z listami zakupów (*ShoppingListPage*). |
| Warunki końcowe | * Na ekranie znajduje się:   + Etykieta z tytułem strony o poprawnej treści.   + Ładujący się spinner.   + Formularz do tworzenia nowej listy zakupów. * Na ekranie nie znajduje się:   + Dane o listach zakupów. |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie renderowania strony. * Sprawdzanie renderowania tytułu, list zakupów oraz formularza, gdy dane nie są ładowane. |
| Dane testowe | 1. Dane istniejącej listy zakupów. |
| Procedura | 1. Renderowana jest strona z listami zakupów (*ShoppingListPage*). |
| Warunki końcowe | * Na ekranie znajduje się:   + Etykieta z tytułem strony o poprawnej treści.   + Rekordy z listami zakupów (odpowiednia ich liczba).   + Formularz do tworzenia nowej listy zakupów. * Na ekranie nie znajduje się:   + Ładujący się spinner. |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie renderowania strony. * Sprawdzanie czy request pobierający dane o zakupach jest wykonywany z poprawnym tokenem. |
| Dane testowe | * Dane istniejącej listy zakupów. * Dane tokenu autoryzacyjnego. |
| Procedura | 1. Renderowana jest strona z listami zakupów (*ShoppingListPage*). |
| Warunki końcowe | * Został wykonany jeden request (*axios*). * Request został wykonany na odpowiedni *endpoint* – z poprawnymi danymi autoryzującymi. |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie renderowania strony. * Sprawdzanie czy renderowana jest poprawna liczba rekordów z listami zakupów. |
| Dane testowe | 1. Dane istniejących list zakupów. |
| Procedura | 1. Renderowana jest strona z listami zakupów (*ShoppingListPage*). |
| Warunki końcowe | * Na ekranie znajdują się rekordy z listami zakupów (odpowiednia ich liczba). |

### Strona TodoPage

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie renderowania strony. * Sprawdzanie renderowania tytułu, spinnera oraz formularza podczas ładowania danych. |
| Dane testowe | --- |
| Procedura | 1. Renderowana jest strona z przedmiotami (*TodoPage*). |
| Warunki końcowe | * Na ekranie znajduje się:   + Etykieta z tytułem strony o poprawnej treści.   + Ładujący się spinner.   + Formularz do tworzenia nowego przedmiotu. * Na ekranie nie znajduje się:   + Dane o przedmiotach. |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie renderowania strony. * Sprawdzanie renderowania tytułu, list zakupów oraz formularza, gdy dane nie są ładowane. |
| Dane testowe | 1. Dane istniejących przedmiotów. |
| Procedura | 1. Renderowana jest strona z przedmiotami (*TodoPage*). |
| Warunki końcowe | * Na ekranie znajduje się:   + Etykieta z tytułem strony o poprawnej treści.   + Rekordy z przedmiotami (odpowiednia ich liczba).   + Formularz do tworzenia nowego przedmiotu. * Na ekranie nie znajduje się:   + Ładujący się spinner. |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie renderowania strony. * Sprawdzanie czy request pobierający dane o przedmiotach jest wykonywany z poprawnym tokenem. |
| Dane testowe | 1. Dane istniejących przedmiotów.  2. Token autoryzacyjny. |
| Procedura | 1. Renderowana jest strona z przedmiotami (*TodoPage*). |
| Warunki końcowe | * Został wykonany jeden request (*axios*). * Request został wykonany na odpowiedni *endpoint* – z poprawnymi danymi autoryzującymi. |

|  |  |
| --- | --- |
| Warunki początkowe | * Sprawdzanie renderowania strony. * Sprawdzanie czy renderowana jest poprawna liczba rekordów z przedmiotami. |
| Dane testowe | 1. Dane przedmiotów na liście zakupów. |
| Procedura | 1. Renderowana jest strona z przedmiotami (*TodoPage*). |
| Warunki końcowe | * Na ekranie znajdują się rekordy z przedmiotami (odpowiednia ich liczba). |

## Metoda uruchamiania testów jednostkowych i integracyjnych

### Wymagania dla środowiska

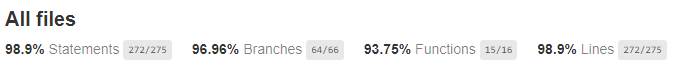
* Posiadanie instalacji node.js + npm kompatybilnej z bibliotekami z package.json
* Posiadanie lokalnej kopii kodu.

### Procedura testowania

* Uruchomienie komendy *npm install* osobno w katalogu w serwerem i klientem.
* Uruchomienie komendy *npm run test* osobno w katalogu w serwerem i klientem.

## Pokrycie kodu testami

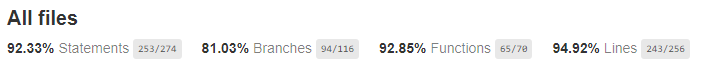
### Raport z pokrycia testami serwera



Cały raport znajduje się w katalogu *server/coverage/lcov-report/index.html*.

Kod został pokryty w 98.9%.

### Raport z pokrycia testami klienta



Cały raport znajduje się w katalogu *client/coverage/lcov-report/index.html*.

Kod został pokryty w 92.33%.

# Testy systemowe

## Terminologia i założenia

W poniższych przypadkach będę używał następujących założeń:

* Strona główna jest serwowana w domenie <http://localhost:3000>
* W serwisie istnieje konto testowego użytkownika.

## Przypadki testowe

* TC1: Wyświetlanie strony logowania.
* TC2: Logowanie użytkownika.
* TC3: Odrzucanie logowań nieznanych użytkowników.
* TC4: Wyświetlanie strony rejestracji.
* TC5: Rejestracja użytkownika.
* TC6: Rejestracja użytkownika z niepoprawnymi danymi.
* TC7: Wyświetlanie strony głównej.
* TC8: Stworzenie nowej listy zakupów.
* TC9: Odrzucenie stworzenia nowej listy zakupów z niepoprawnymi danymi.
* TC10: Usuwanie listy zakupów.
* TC11: Aktualizacja nazwy listy zakupów.
* TC12: Aktualizacja wykonania listy zakupów.
* TC13: Wyświetlanie strony z przedmiotami na liście zakupów.
* TC14: Dodanie nowego przedmiotu do listy zakupów.
* TC15: Odrzucenie stworzenia nowego przedmiotu z niepoprawnymi danymi.
* TC16: Zaznaczenie realizacji przedmiotu.
* TC17: Usuwanie przedmiotu.

## Opisy przypadków testowych

### TC1

|  |  |
| --- | --- |
| Nazwa | TC1: Wyświetlanie strony logowania. |
| Opis | Sprawdzenie możliwości wyświetlenia strony logowania do aplikacji. |
| Warunki wstępne | * Użytkownik nie jest zalogowany. |
| Dane testowe | * --- |
| Typ testu | Pozytywny |

|  |  |  |  |
| --- | --- | --- | --- |
| L.P. | Opis kroku | Oczekiwany rezultat | Rzeczywisty rezultat |
|  | Wejdź na stronę „<http://localhost:3000/>”. | Zostaje wyświetlona strona logowania (http://localhost:3000/login). | Strona logowania wyświetliła się. |
|  | Sprawdź, czy na stronie wyświetla się:  - tytuł „Login”  - formatka logowania,  - etykieta „Username: ”,  - input nazwy użytkownika,  - etykieta „Password: ”,  - input hasła. | Na stronie wyświetla się:  - tytuł „Login”  - formatka logowania,  - etykieta „Username: ”,  - input nazwy użytkownika,  - etykieta „Password: ”,  - input hasła. | Na stronie wyświetliło się:  - tytuł „Login”  - formatka logowania,  - etykieta „Username: ”,  - input nazwy użytkownika,  - etykieta „Password: ”,  - input hasła. |
|  | Sprawdź, czy strona nie zawiera komunikatu o błędzie. | Strona nie zawiera komunikatu o błędzie. | Strona nie zawierała komunikatu o błędzie. |

### TC2

|  |  |
| --- | --- |
| Nazwa | TC2: Logowanie użytkownika. |
| Opis | Sprawdzenie możliwości zalogowania użytkownika do systemu. |
| Warunki wstępne | * Użytkownik nie jest zalogowany. * Użytkownik posiada konto w systemie. |
| Dane testowe | * Poprawne dane użytkownika testowego (nazwa i hasło). |
| Typ testu | Pozytywny |

|  |  |  |  |
| --- | --- | --- | --- |
| L.P. | Opis kroku | Oczekiwany rezultat | Rzeczywisty rezultat |
|  | Wejdź na stronę logowania http://localhost:3000/login | Zostaje wyświetlona strona logowania (http://localhost:3000/login) | Strona logowania wyświetliła się |
|  | Wypełnij pola *username* oraz *password* danymi testowymi | Pola *username* oraz *password* zostaną wypełnione danymi testowymi. | Pola *username* oraz *password* zostały wypełnione danymi testowymi. |
|  | Kliknij przycisk *Submit*. | Użytkownik zostanie przekierowany na stronę http://localhost:3000/. | Użytkownik został przekierowany na stronę http://localhost:3000/. |
|  | Sprawdź, czy na stronie wyświetla się tytuł „Shopping List App”. | Na stronie zostanie wyświetlony tytuł „Shopping List App”. | Na stronie został wyświetlony tytuł „Shopping List App”. |

### TC3

|  |  |
| --- | --- |
| Nazwa | TC3: Odrzucanie logowań nieznanych użytkowników. |
| Opis | Sprawdzenie zabezpieczenia zalogowania się z dowolnej nazwy użytkownika i dowolnego hasła. |
| Warunki wstępne | * Użytkownik nie jest zalogowany. * Nie istnieje użytkownik o podanych danych logowania (nazwie użytkownika i haśle) |
| Dane testowe | * Niepoprawne dane logowania (nazwa oraz hasło) oraz odpowiadająca im treść komunikatu o błędzie.   + Pusta nazwa i puste hasło.   + Poprawna nazwa i puste hasło.   + Pusta nazwa i poprawne hasło.   + Za krótka nazwa i poprawne hasło.   + Poprawna nazwa i za krótkie hasło.   + Za krótka nazwa i za krótkie hasło.   + Nazwa i hasło składające się z nielegalnego znaku.   + Poprawna nazwa i hasło z nielegalnymi znakami.   + Nazwa i hasło z nielegalnymi znakami.   + Nazwa z nielegalnymi znakami i poprawne hasło.   + Za długa nazwa i za długie hasło.   + Poprawna nazwa i za długie hasło.   + Za długa nazwa i poprawne hasło.   + Niepoprawna nazwa i niepoprawne hasło.   + Poprawna nazwa i niepoprawne hasło.   + Niepoprawna nazwa i poprawne hasło. |
| Typ testu | Negatywny |

|  |  |  |  |
| --- | --- | --- | --- |
| L.P. | Opis kroku | Oczekiwany rezultat | Rzeczywisty rezultat |
|  | Wejdź na stronę logowania http://localhost:3000/login. | Zostaje wyświetlona strona logowania (http://localhost:3000/login). | Strona logowania wyświetliła się. |
|  | Wypełnij pola *username* oraz *password* danymi testowymi | Pola *username* oraz *password* zostaną wypełnione danymi testowymi. | Pola *username* oraz *password* zostały wypełnione danymi testowymi. |
|  | Kliknij przycisk *Submit*. | Użytkownik nie zostanie zalogowany. Użytkownik nie zostanie przekierowany na stronę http://localhost:3000/. | Użytkownik nie został zalogowany. Użytkownik nie został przekierowany na stronę http://localhost:3000/. |
|  | Sprawdź, czy został wyświetlony komunikat o niepowodzeniu, o odpowiedniej treści. | Na stronie zostanie wyświetlony komunikat o niepowodzeniu, o odpowiedniej treści. | Na stronie został wyświetlony komunikat o niepowodzeniu, o odpowiedniej treści. |

### TC4

|  |  |
| --- | --- |
| Nazwa | TC4: Wyświetlanie strony rejestracji. |
| Opis | Sprawdzenie możliwości wyświetlenia strony rejestracji do aplikacji. |
| Warunki wstępne | * Użytkownik nie jest zalogowany. |
| Dane testowe | * --- |
| Typ testu | Pozytywny |

|  |  |  |  |
| --- | --- | --- | --- |
| L.P. | Opis kroku | Oczekiwany rezultat | Rzeczywisty rezultat |
|  | Wejdź na stronę „<http://localhost:3000/>register”. | Zostaje wyświetlona strona rejestracji (http://localhost:3000/register). | Strona rejestracji wyświetliła się. |
|  | Sprawdź, czy na stronie wyświetla się:  - tytuł „Register”,  - formatka rejestracji,  - etykieta „Username: ”,  - input nazwy użytkownika,  - etykieta „Password: ”,  - input hasła. | Na stronie wyświetla się:  - tytuł „Register”,  - formatka rejestracji,  - etykieta „Username: ”,  - input nazwy użytkownika,  - etykieta „Password: ”,  - input hasła. | Na stronie wyświetliło się:  - tytuł „Register”,  - formatka rejestracji,  - etykieta „Username: ”,  - input nazwy użytkownika,  - etykieta „Password: ”,  - input hasła. |
|  | Sprawdź, czy strona nie zawiera komunikatu o błędzie. | Strona nie zawiera komunikatu o błędzie. | Strona nie zawierała komunikatu o błędzie. |

### TC5

|  |  |
| --- | --- |
| Nazwa | TC5: Rejestracja użytkownika. |
| Opis | Sprawdzenie możliwości rejestrowania użytkownika do systemu. |
| Warunki wstępne | * Użytkownik nie jest zalogowany. * Użytkownik nie posiada konta w systemie. |
| Dane testowe | * Poprawna nazwa oraz poprawne hasło użytkownika do rejestracji. |
| Typ testu | Pozytywny |

|  |  |  |  |
| --- | --- | --- | --- |
| L.P. | Opis kroku | Oczekiwany rezultat | Rzeczywisty rezultat |
|  | Wejdź na stronę rejestracji http://localhost:3000/register | Zostaje wyświetlona strona rejestracji (http://localhost:3000/register) | Strona rejestracji wyświetliła się. |
|  | Wypełnij pola *username* oraz *password* danymi testowymi | Pola *username* oraz *password* zostaną wypełnione danymi testowymi. | Pola *username* oraz *password* zostały wypełnione danymi testowymi. |
|  | Kliknij przycisk *Submit Register*. | Użytkownik zostanie przekierowany na stronę logowania: http://localhost:3000/login. | Użytkownik został przekierowany na stronę logowania: http://localhost:3000/login. |
|  | Sprawdź, czy na stronie wyświetla się tytuł „Login”. | Na stronie zostanie wyświetlony tytuł „Login”. | Na stronie został wyświetlony tytuł „Login”. |

### TC6

|  |  |
| --- | --- |
| Nazwa | TC6: Rejestracja użytkownika z niepoprawnymi danymi. |
| Opis | Sprawdzenie odrzucenia możliwości zarejestrowania użytkownika w systemie z niepoprawnymi danymi. |
| Warunki wstępne | * Użytkownik nie jest zalogowany. * Użytkownik nie posiada konta w systemie. |
| Dane testowe | * Niepoprawne dane rejestracji (nazwa oraz hasło).   + Pusta nazwa i puste hasło.   + Poprawna nazwa i puste hasło.   + Pusta nazwa i poprawne hasło.   + Za krótka nazwa i poprawne hasło.   + Poprawna nazwa i za krótkie hasło.   + Za krótka nazwa i za krótkie hasło.   + Nazwa i hasło składające się z nielegalnego znaku.   + Poprawna nazwa i hasło z nielegalnymi znakami.   + Nazwa i hasło z nielegalnymi znakami.   + Nazwa z nielegalnymi znakami i poprawne hasło.   + Za długa nazwa i za długie hasło.   + Poprawna nazwa i za długie hasło.   + Za długa nazwa i poprawne hasło. |
| Typ testu | Negatywny |

|  |  |  |  |
| --- | --- | --- | --- |
| L.P. | Opis kroku | Oczekiwany rezultat | Rzeczywisty rezultat |
|  | Wejdź na stronę rejestracji http://localhost:3000/register. | Zostaje wyświetlona strona rejestracji (http://localhost:3000/register). | Strona rejestracji wyświetliła się. |
|  | Wypełnij pola *username* oraz *password* danymi testowymi | Pola *username* oraz *password* zostaną wypełnione danymi testowymi. | Pola *username* oraz *password* zostały wypełnione danymi testowymi. |
|  | Kliknij przycisk *Submit Register*. | Użytkownik nie zostanie zarejestrowany. Użytkownik nie zostanie przekierowany na stronę http://localhost:3000/login. | Użytkownik nie został zarejestrowany. Użytkownik nie został przekierowany na stronę http://localhost:3000/login. |
|  | Sprawdź, czy został wyświetlony komunikat o niepowodzeniu, o odpowiedniej treści. | Na stronie zostanie wyświetlony komunikat o niepowodzeniu, o odpowiedniej treści. | Na stronie został wyświetlony komunikat o niepowodzeniu, o odpowiedniej treści. |

### TC7

|  |  |
| --- | --- |
| Nazwa | TC7: Wyświetlanie strony głównej. |
| Opis | Sprawdzenie możliwości wyświetlenia głównej strony aplikacji. |
| Warunki wstępne | * Użytkownik jest zalogowany. |
| Dane testowe | * Dane testowego użytkownika, istniejącego w systemie. |
| Typ testu | Pozytywny |

|  |  |  |  |
| --- | --- | --- | --- |
| L.P. | Opis kroku | Oczekiwany rezultat | Rzeczywisty rezultat |
|  | Zaloguj się i wejdź na stronę „<http://localhost:3000/>”. | Zostaje wyświetlona strona główna (http://localhost:3000/). | Strona główna wyświetliła się. |
|  | Sprawdź, czy na stronie wyświetla się:  - tytuł „Shopping List App”  - rekordy z listami zakupów,  - formatka do tworzenia listy zakupów,  - etykieta „Nazwa: ”,  - input nazwy listy zakupów,  - etykieta „Sugerowana data wykonania: ”,  - input daty,  - przycisk do tworzenia listy zakupów. | Na stronie wyświetla się:  - tytuł „Shopping List App”  - rekordy z listami zakupów,  - formatka do tworzenia listy zakupów,  - etykieta „Nazwa: ”,  - input nazwy listy zakupów,  - etykieta „Sugerowana data wykonania: ”,  - input daty,  - przycisk do tworzenia listy zakupów. | Na stronie wyświetliło się:  - tytuł „Shopping List App”  - rekordy z listami zakupów,  - formatka do tworzenia listy zakupów,  - etykieta „Nazwa: ”,  - input nazwy listy zakupów,  - etykieta „Sugerowana data wykonania: ”,  - input daty,  - przycisk do tworzenia listy zakupów. |

### TC8

|  |  |
| --- | --- |
| Nazwa | TC8: Stworzenie nowej listy zakupów. |
| Opis | Sprawdzenie możliwości stworzenia nowej listy zakupów. |
| Warunki wstępne | * Użytkownik jest zalogowany. |
| Dane testowe | * Poprawna nazwa oraz poprawna data testowej listy zakupów. |
| Typ testu | Pozytywny |

|  |  |  |  |
| --- | --- | --- | --- |
| L.P. | Opis kroku | Oczekiwany rezultat | Rzeczywisty rezultat |
|  | Zaloguj się i wejdź na stronę „<http://localhost:3000/>”. | Zostaje wyświetlona strona główna (http://localhost:3000/). | Strona główna wyświetliła się. |
|  | Wypełnij pola *name* oraz *date* danymi testowymi. | Pola *name* oraz *date* zostaną wypełnione danymi testowymi. | Pola *name* oraz *date* zostały wypełnione danymi testowymi. |
|  | Kliknij przycisk *Create SL*. | Nowa lista zostanie wyświetlona jako nowy element strony. Liczba wyświetlanych list zakupów zwiększy się o 1. | Nowa lista została wyświetlona jako nowy element strony. Liczba wyświetlanych list zakupów zwiększyła się o 1. |
|  | Sprawdź, czy nowo dodana lista jest widoczna. | Na stronie zostanie wyświetlona nowo dodana lista. | Na stronie została wyświetlona nowo dodana lista. |

### TC9

|  |  |
| --- | --- |
| Nazwa | TC9: Odrzucenie stworzenia nowej listy zakupów z niepoprawnymi danymi. |
| Opis | Sprawdzenie odrzucenia możliwości stworzenia nowej listy zakupów. |
| Warunki wstępne | * Użytkownik jest zalogowany. |
| Dane testowe | * Niepoprawne dane dodawanej listy zakupów (nazwa oraz data):   + Za długa nazwa i poprawna data.   + Za krótka nazwa i poprawna data.   + Pusta nazwa i poprawna data.   + Poprawna nazwa i pusta data.   + Nazwa z nielegalnymi znakami i poprawna data. |
| Typ testu | Negatywny |

|  |  |  |  |
| --- | --- | --- | --- |
| L.P. | Opis kroku | Oczekiwany rezultat | Rzeczywisty rezultat |
|  | Zaloguj się i wejdź na stronę „<http://localhost:3000/>”. | Zostaje wyświetlona strona główna (http://localhost:3000/). | Strona główna wyświetliła się. |
|  | Wypełnij pola *name* oraz *date* danymi testowymi. | Pola *name* oraz *date* zostaną wypełnione danymi testowymi. | Pola *name* oraz *date* zostały wypełnione danymi testowymi. |
|  | Kliknij przycisk *Create SL*. | Nowa lista nie zostanie wyświetlona jako nowy element strony. Liczba wyświetlanych list zakupów nie zmieni się. | Nowa lista nie została wyświetlona jako nowy element strony. Liczba wyświetlanych list zakupów nie zmieniła się. |
|  | Sprawdź, czy nowo dodana lista nie jest widoczna. | Na stronie nie zostanie wyświetlona nowo dodana lista. | Na stronie nie została wyświetlona nowo dodana lista. |

### TC10

|  |  |
| --- | --- |
| Nazwa | TC10: Usuwanie listy zakupów. |
| Opis | Sprawdzenie możliwości usunięcia istniejącej listy zakupów. |
| Warunki wstępne | * Użytkownik jest zalogowany. * Lista zakupów istnieje. |
| Dane testowe | * Nazwa oraz data testowej listy zakupów, istniejącej w systemie. |
| Typ testu | Pozytywny |

|  |  |  |  |
| --- | --- | --- | --- |
| L.P. | Opis kroku | Oczekiwany rezultat | Rzeczywisty rezultat |
|  | Zaloguj się i wejdź na stronę „<http://localhost:3000/>”. | Zostaje wyświetlona strona główna (http://localhost:3000/). | Strona główna wyświetliła się. |
|  | Dodaj nową listę zakupów. Wypełnij pola *name* oraz *date* danymi testowymi. Następnie kliknij przycisk *Create SL*. | Nowa lista zostanie dodana i będzie widoczna. | Nowa lista została dodana i jest widoczna. |
|  | Kliknij przycisk *Delete* przy danej liście zakupów. | Nowa lista przestanie być wyświetlana na stronie. Liczba wyświetlanych list zakupów zmniejszy się o 1. | Nowa lista przestała być wyświetlana na stronie. Liczba wyświetlanych list zakupów zmniejszyła się o 1. |
|  | Sprawdź, czy lista nie jest widoczna. | Lista nie będzie widoczna na stronie. | Lista nie jest widoczna na stronie. |

### TC11

|  |  |
| --- | --- |
| Nazwa | TC10: Aktualizacja nazwy listy zakupów. |
| Opis | Sprawdzenie możliwości zaktualizowania nazwy listy zakupów. |
| Warunki wstępne | * Użytkownik jest zalogowany. * Lista zakupów istnieje. |
| Dane testowe | * Nazwa oraz data testowej listy zakupów, istniejącej w systemie. * Nowa nazwa listy zakupów (poprawna; za długa; niepoprawna):   + Poprawna nowa nazwa.   + Za długa nowa nazwa.   + Nazwa składająca się z nielegalnych znaków. |
| Typ testu | Pozytywny |

|  |  |  |  |
| --- | --- | --- | --- |
| L.P. | Opis kroku | Oczekiwany rezultat | Rzeczywisty rezultat |
|  | Zaloguj się i wejdź na stronę „<http://localhost:3000/>”. | Zostaje wyświetlona strona główna (http://localhost:3000/). | Strona główna wyświetliła się. |
|  | Dodaj nową listę zakupów. Wypełnij pola *name* oraz *date* danymi testowymi. Następnie kliknij przycisk *Create SL*. | Nowa lista zostanie dodana i będzie widoczna. | Nowa lista została dodana i jest widoczna. |
|  | W polu z nazwą listy dopisz nową nazwę. | Lista będzie posiadać nową nazwę. Liczba wyświetlanych list nie zmieni się. | Lista posiada nową nazwę. Liczba wyświetlanych list nie zmieniła się. |
|  | Sprawdź, czy lista z nową nazwą jest widoczna. | Lista z nową nazwą będzie widoczna na stronie. | Lista z nową nazwą jest widoczna na stronie. |
|  | W polu z nazwą listy klikaj backspace, o liczbie długości nazwy listy. | Lista będzie posiadać nową nazwę, o długości 2 (minimalna). Liczba wyświetlanych list nie zmieniła się. | Lista posiada nową nazwę, o długości 2. Liczba wyświetlanych list nie zmieniła się. |
|  | Sprawdź, czy lista z nową nazwą jest widoczna. | Lista z nową nazwą będzie widoczna na stronie. | Lista z nową nazwą jest widoczna na stronie. |
|  | W polu z nazwą listy dopisz nielegalną nazwę. | Lista będzie posiadać niezmienioną nazwę (o długości 2). Liczba wyświetlanych list nie zmieniła się. | Lista posiada niezmienioną nazwę (o długości 2). Liczba wyświetlanych list nie zmieniła się. |
|  | Sprawdź, czy lista z nową nazwą jest widoczna. | Lista z nową nazwą będzie widoczna na stronie. | Lista z nową nazwą jest widoczna na stronie. |
|  | W polu z nazwą listy dopisz za długą nazwę. | Lista będzie posiadać nową nazwę, o długości 50 (maksymalna). Liczba wyświetlanych list nie zmieniła się. | Lista posiada nową nazwę, o długości 50. Liczba wyświetlanych list nie zmieniła się. |
|  | Sprawdź, czy lista z nową nazwą jest widoczna. | Lista z nową nazwą będzie widoczna na stronie. | Lista z nową nazwą jest widoczna na stronie. |

### TC12

|  |  |
| --- | --- |
| Nazwa | TC10: Aktualizacja wykonania listy zakupów. |
| Opis | Sprawdzenie możliwości zaktualizowania wykonania listy zakupów. |
| Warunki wstępne | * Użytkownik jest zalogowany. * Lista zakupów istnieje. |
| Dane testowe | * Nazwa oraz data testowej listy zakupów, istniejącej w systemie. |
| Typ testu | Pozytywny |

|  |  |  |  |
| --- | --- | --- | --- |
| L.P. | Opis kroku | Oczekiwany rezultat | Rzeczywisty rezultat |
|  | Zaloguj się i wejdź na stronę „<http://localhost:3000/>”. | Zostaje wyświetlona strona główna (http://localhost:3000/). | Strona główna wyświetliła się. |
|  | Dodaj nową listę zakupów. Wypełnij pola *name* oraz *date* danymi testowymi. Następnie kliknij przycisk *Create SL*. | Nowa lista zostanie dodana i będzie widoczna. | Nowa lista została dodana i jest widoczna. |
|  | Zaznacz pole *checkbox* danej listy. | Pole *checkbox* danej listy będzie zaznaczone. Liczba wyświetlanych list nie zmieni się. | Pole *checkbox* danej listy jest zaznaczone. Liczba wyświetlanych list nie zmieniła się. |
|  | Sprawdź, czy lista z wykonaniem jest widoczna. | Po przelogowaniu, lista z wykonaniem będzie widoczna na stronie. | Po przelogowaniu, lista z wykonaniem jest widoczna na stronie. |
|  | Odznacz pole *checkbox* danej listy. | Pole *checkbox* danej listy będzie odznaczone. Liczba wyświetlanych list nie zmieni się. | Pole *checkbox* danej listy jest odznaczone. Liczba wyświetlanych list nie zmieniła się. |
|  | Sprawdź, czy lista (niewykonana) jest widoczna. | Lista (niewykonana) będzie widoczna na stronie. | Lista (niewykonana) jest widoczna na stronie. |

### TC13

|  |  |
| --- | --- |
| Nazwa | TC13: Wyświetlanie strony z przedmiotami na liście zakupów. |
| Opis | Sprawdzenie możliwości wyświetlenia strony z przedmiotami znajdującymi się na liście zakupów. |
| Warunki wstępne | * Użytkownik jest zalogowany. * Lista zakupów istnieje. |
| Dane testowe | * Dane testowego użytkownika, istniejącego w systemie. * Dane testowej listy zakupów, istniejącej w systemie. |
| Typ testu | Pozytywny |

|  |  |  |  |
| --- | --- | --- | --- |
| L.P. | Opis kroku | Oczekiwany rezultat | Rzeczywisty rezultat |
|  | Zaloguj się i wejdź na stronę „<http://localhost:3000/>”. | Zostaje wyświetlona strona główna (http://localhost:3000/). | Strona główna wyświetliła się. |
|  | Stwórz listę testową i przejdź do szczegółów tej listy (klikając *Details*). | Testowa lista zostanie stworzona. Zostanie wykonane przekierowanie na stronę ze szczegółami listy zakupów. | Testowa lista została stworzona. Zostało wykonane przekierowanie na stronę ze szczegółami listy zakupów. |
|  | Sprawdź, czy na stronie wyświetla się:  - tytuł „Todo App”  - rekordy z przedmiotami,  - formatka do dodawania przedmiotów,  - etykieta „Nazwa: ”,  - input nazwy listy zakupów,  - etykieta „Ilość: ”,  - input ilości,  - etykieta „Gramatura: ”,  - input gramatury,  - przycisk do tworzenia przedmiotu. | Na stronie wyświetla się:  - tytuł „Todo App”  - rekordy z przedmiotami,  - formatka do dodawania przedmiotów,  - etykieta „Nazwa: ”,  - input nazwy listy zakupów,  - etykieta „Ilość: ”,  - input ilości,  - etykieta „Gramatura: ”,  - input gramatury,  - przycisk do tworzenia przedmiotu. | Na stronie wyświetliło się:  - tytuł „Todo App”  - rekordy z przedmiotami,  - formatka do dodawania przedmiotów,  - etykieta „Nazwa: ”,  - input nazwy listy zakupów,  - etykieta „Ilość: ”,  - input ilości,  - etykieta „Gramatura: ”,  - input gramatury,  - przycisk do tworzenia przedmiotu. |

### TC14

|  |  |
| --- | --- |
| Nazwa | TC14: Dodanie nowego przedmiotu do listy zakupów. |
| Opis | Sprawdzenie możliwości dodania nowego przedmiotu do listy zakupów. |
| Warunki wstępne | * Użytkownik jest zalogowany. * Lista zakupów istnieje. |
| Dane testowe | * Dane testowego użytkownika, istniejącego w systemie. * Dane testowej listy zakupów, istniejącej w systemie. * Poprawne dane dodawanego przedmiotu.   + Nazwa będąca wśród listy wyboru, poprawna liczba, gramatura z listy wyboru.   + Własna nazwa przedmiotu, poprawna liczba, gramatura z listy wyboru. |
| Typ testu | Pozytywny |

|  |  |  |  |
| --- | --- | --- | --- |
| L.P. | Opis kroku | Oczekiwany rezultat | Rzeczywisty rezultat |
|  | Zaloguj się i wejdź na stronę „<http://localhost:3000/>”. | Zostaje wyświetlona strona główna (http://localhost:3000/). | Strona główna wyświetliła się. |
|  | Przejdź do szczegółów listy testowej (klikając *Details*). | Zostanie wykonane przekierowanie na stronę ze szczegółami listy zakupów. Zostaje wyświetlona strona ze szczegółami (http://localhost:3000/details). | Zostało wykonane przekierowanie na stronę ze szczegółami listy zakupów. Została wyświetlona strona ze szczegółami. |
|  | Wypełnij pola *name, amount* oraz *grammage* danymi testowymi. | Pola *name, amount* oraz *grammage* zostaną wypełnione danymi testowymi. | Pola *name, amount* oraz *grammage* zostały wypełnione danymi testowymi. |
|  | Kliknij przycisk *Create TODO*. | Nowy przedmiot zostanie wyświetlony jako nowy element strony. Liczba wyświetlanych przedmiotów zwiększy się o 1. | Nowy przedmiot został wyświetlony jako nowy element strony. Liczba wyświetlanych przedmiotów zwiększyła się o 1. |
|  | Sprawdź, czy nowo dodany przedmiot jest widoczny. | Na stronie zostanie wyświetlony nowo dodany przedmiot. | Na stronie został wyświetlony nowo dodany przedmiot. |

### TC15

|  |  |
| --- | --- |
| Nazwa | TC15: Odrzucenie stworzenia nowego przedmiotu z niepoprawnymi danymi. |
| Opis | Sprawdzenie odrzucenia możliwości stworzenia nowego przedmiotu. |
| Warunki wstępne | * Użytkownik jest zalogowany. * Lista zakupów istnieje. |
| Dane testowe | * Dane testowego użytkownika, istniejącego w systemie. * Dane testowej listy zakupów, istniejącej w systemie. * Niepoprawne dane dodawanego przedmiotu:   + Pusta nazwa, poprawna liczba, poprawna gramatura.   + Poprawna nazwa, pusta liczba, poprawna gramatura.   + Poprawna nazwa, poprawna liczba, pusta gramatura.   + Poprawna nazwa, pusta liczba, pusta gramatura.   + Pusta nazwa, poprawna liczba, pusta gramatura.   + Pusta nazwa, pusta liczba, poprawna gramatura.   + Pusta nazwa, pusta liczba, pusta gramatura.   + Poprawna nazwa, 0, poprawna gramatura.   + Za długa nazwa, poprawna liczba, poprawna gramatura.   + Nazwa z nielegalnym znakiem/znakami, poprawna liczba, poprawna gramatura. |
| Typ testu | Negatywny |

|  |  |  |  |
| --- | --- | --- | --- |
| L.P. | Opis kroku | Oczekiwany rezultat | Rzeczywisty rezultat |
|  | Zaloguj się i wejdź na stronę „<http://localhost:3000/>”. | Zostaje wyświetlona strona główna (http://localhost:3000/). | Strona główna wyświetliła się. |
|  | Przejdź do szczegółów listy testowej (klikając *Details*). | Zostanie wykonane przekierowanie na stronę ze szczegółami listy zakupów. Zostaje wyświetlona strona ze szczegółami (http://localhost:3000/details). | Zostało wykonane przekierowanie na stronę ze szczegółami listy zakupów. Została wyświetlona strona ze szczegółami. |
|  | Wypełnij pola *name, amount* oraz *grammage* danymi testowymi. | Pola *name, amount* oraz *grammage* zostaną wypełnione danymi testowymi. | Pola *name, amount* oraz *grammage* zostały wypełnione danymi testowymi. |
|  | Kliknij przycisk *Create TODO*. | Nowy przedmiot nie zostanie wyświetlony jako nowy element strony. Liczba wyświetlanych przedmiotów nie zmieni się. | Nowy przedmiot nie został wyświetlony. Liczba wyświetlanych przedmiotów nie zmieniła się. |
|  | Sprawdź, czy nowo dodany przedmiot jest nie widoczny. | Na stronie nie zostanie wyświetlony nowo dodany przedmiot. | Na stronie nie został wyświetlony nowo dodany przedmiot. |

### TC16

|  |  |
| --- | --- |
| Nazwa | TC16: Zaznaczenie realizacji przedmiotu. |
| Opis | Sprawdzenie możliwości zaktualizowania wykonania przedmiotu. |
| Warunki wstępne | * Użytkownik jest zalogowany. * Lista zakupów istnieje. |
| Dane testowe | * Dane testowego użytkownika, istniejącego w systemie. * Dane testowej listy zakupów, istniejącej w systemie. * Dane testowego przedmiotu, istniejącego w systemie. |
| Typ testu | Pozytywny |

|  |  |  |  |
| --- | --- | --- | --- |
| L.P. | Opis kroku | Oczekiwany rezultat | Rzeczywisty rezultat |
|  | Zaloguj się i wejdź na stronę „<http://localhost:3000/>”. | Zostaje wyświetlona strona główna (http://localhost:3000/). | Strona główna wyświetliła się. |
|  | Przejdź do szczegółów listy testowej (klikając *Details*). | Zostanie wykonane przekierowanie na stronę ze szczegółami listy zakupów. Zostaje wyświetlona strona ze szczegółami (http://localhost:3000/details). | Zostało wykonane przekierowanie na stronę ze szczegółami listy zakupów. Została wyświetlona strona ze szczegółami. |
|  | Zaznacz pole *checkbox* danego przedmiotu. | Pole *checkbox* danego przedmiotu będzie zaznaczone. Liczba wyświetlanych przedmiotów nie zmieni się. Nazwa będzie przekreślona. | Pole *checkbox* danego przedmiotu jest zaznaczone. Liczba wyświetlanych przedmiotów nie zmieniła się. Nazwa jest przekreślona. |
|  | Sprawdź, czy przedmiot z wykonaniem jest widoczny. | Przed i po przelogowaniu, przedmiot z wykonaniem będzie widoczny na stronie. Nazwa będzie przekreślona. | Po przelogowaniu, lista z wykonaniem jest widoczna na stronie. Nazwa jest przekreślona. |
|  | Odznacz pole *checkbox* danego przedmiotu. | Pole *checkbox* danego przedmiotu będzie odznaczone. Liczba wyświetlanych list nie zmieni się. Nazwa nie będzie przekreślona. | Pole *checkbox* danego przedmiotu jest odznaczone. Liczba wyświetlanych przedmiotów nie zmieniła się. Nazwa nie jest przekreślona. |
|  | Sprawdź, czy przedmiot (niewykonana) jest widoczna. | Przedmiot (niewykonany) będzie widoczna na stronie. | Przedmiot (niewykonany) jest widoczny na stronie. |

### TC17

|  |  |
| --- | --- |
| Nazwa | TC10: Usuwanie przedmiotu. |
| Opis | Sprawdzenie możliwości usunięcia istniejącego przedmiotu. |
| Warunki wstępne | * Użytkownik jest zalogowany. * Lista zakupów istnieje. * Przedmiot istnieje. |
| Dane testowe | * Dane testowego użytkownika, istniejącego w systemie. * Dane testowej listy zakupów, istniejącej w systemie. * Dane testowego przedmiotu, istniejącego w systemie. |
| Typ testu | Pozytywny |

|  |  |  |  |
| --- | --- | --- | --- |
| L.P. | Opis kroku | Oczekiwany rezultat | Rzeczywisty rezultat |
|  | Zaloguj się i wejdź na stronę „<http://localhost:3000/>”. | Zostaje wyświetlona strona główna (http://localhost:3000/). | Strona główna wyświetliła się. |
|  | Przejdź do szczegółów listy testowej (klikając *Details*). | Zostanie wykonane przekierowanie na stronę ze szczegółami listy zakupów. Zostaje wyświetlona strona ze szczegółami (http://localhost:3000/details). | Zostało wykonane przekierowanie na stronę ze szczegółami listy zakupów. Została wyświetlona strona ze szczegółami. |
|  | Kliknij przycisk *Delete* przy danym przedmiocie. | Przedmiot przestanie być wyświetlany na stronie. Liczba wyświetlanych przedmiotów zmniejszy się o 1. | Przedmiot przestał być wyświetlany na stronie. Liczba wyświetlanych przedmiotów zmniejszyła się o 1. |
|  | Sprawdź, czy przedmiot nie jest widoczna. | Przedmiot nie będzie widoczny na stronie. | Przedmiot nie jest widoczny na stronie. |

## Metoda uruchamiania testów systemowych

### Wymagania dla środowiska

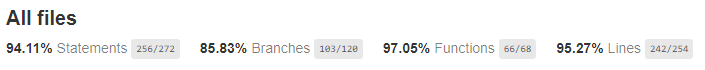
* Posiadanie instalacji node.js + npm kompatybilnej z bibliotekami z package.json.
* Posiadanie lokalnej kopii kodu.

### Procedura testowania

* Uruchomienie komendy *npm install* w katalogu z projektem klienta.
* Uruchomienie komendy *npm run cy:coverage* w katalogu z projektem klienta. Wtedy dostępny będzie raport pokrycia testami systemowymi.
  + Aby uruchomić testy w trybie interaktywnym należy uruchomić komendę *npm run cypress*.

## Pokrycie kodu testami systemowymi

Raport z pokrycia znajduje się w folderze: *client/coverage/*.



Kod został pokryty w 94.11%.

# Testy wydajnościowe

## Użyte narzędzia i technologie

Do testów użyto dwóch rodzajów narzędzi:

* ***BlazeMeter*** jako narzędzie do “nagrywania” kroków testów. Rozszerzenie Chrome rejestruje wszystkie żądania HTTP/S i rzeczywiste interakcje użytkowników wysyłane przez przeglądarkę. Tworzy skrypt *JMeter* (pliku o formacie .jmx) zawierający zapytania użyte przy nagrywaniu testu w przeglądarce.
* ***JMeter*** jako narzędzie do przeprowadzania testów.

Następnie, przy użyciu wygenerowanego pliku .jmx skonfigurowano w GUI JMeter krok, mający na celu output wyników testu do pliku. Skonfigurowano również parametr mający na celu użycie 200 użytkowników na raz, każdy po 20 iteracji scenariusza.

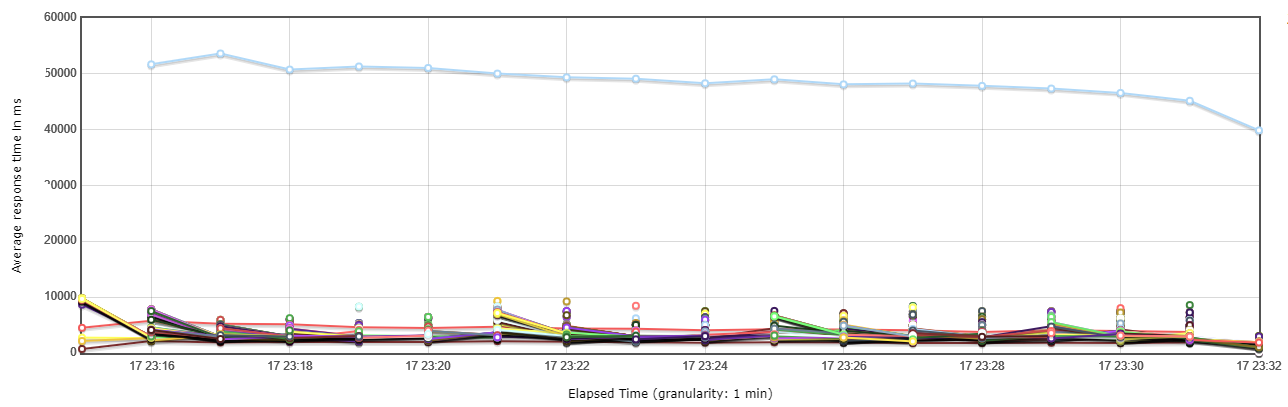
## Scenariusz testowy

* Logowanie użytkownika.
* Stworzenie nowej listy zakupów.
* Przejście do szczegółów listy zakupów.
* Dodanie przedmiotów do listy zakupów.
* Zaznaczenie przedmiotów jako wykonanych.
* Wrócenie na stronę główną.
* Usunięcie listy zakupów.

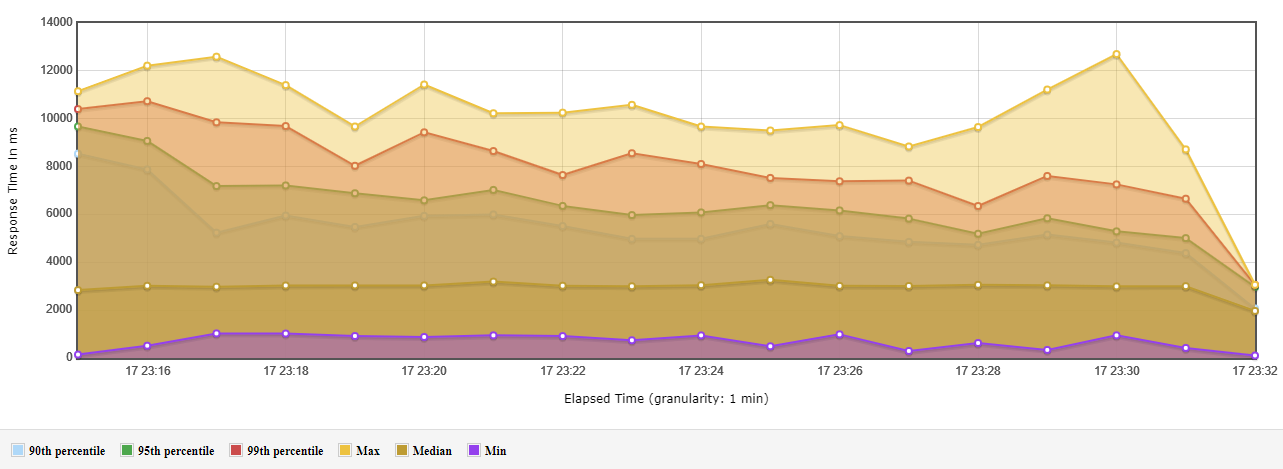
## Wyniki testów

Poniżej znajdują się niektóre wykresy z raportu HTML. Cały raport znajduje się w folderze *performance-tests/report*.

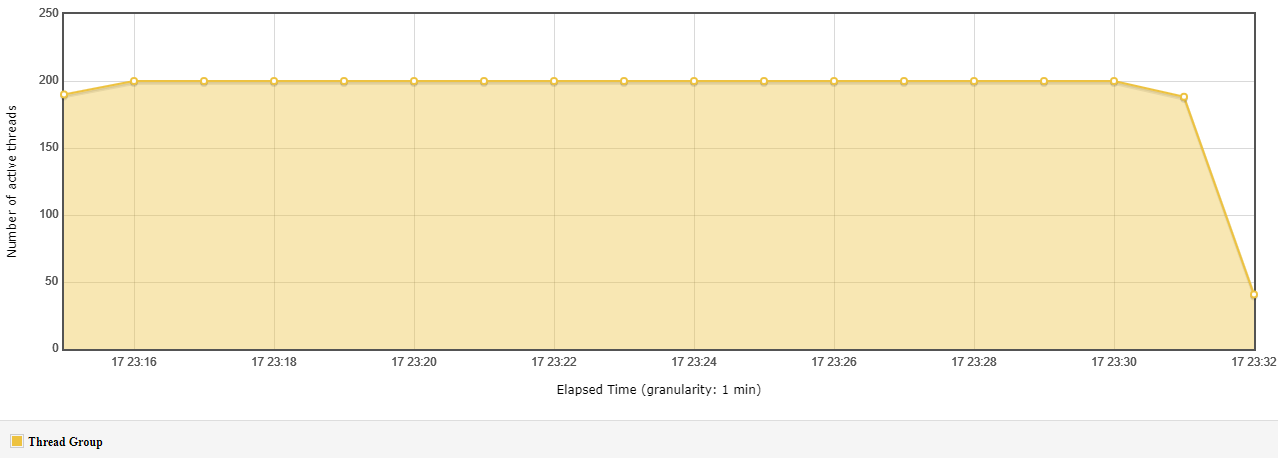
### Czasy odpowiedzi



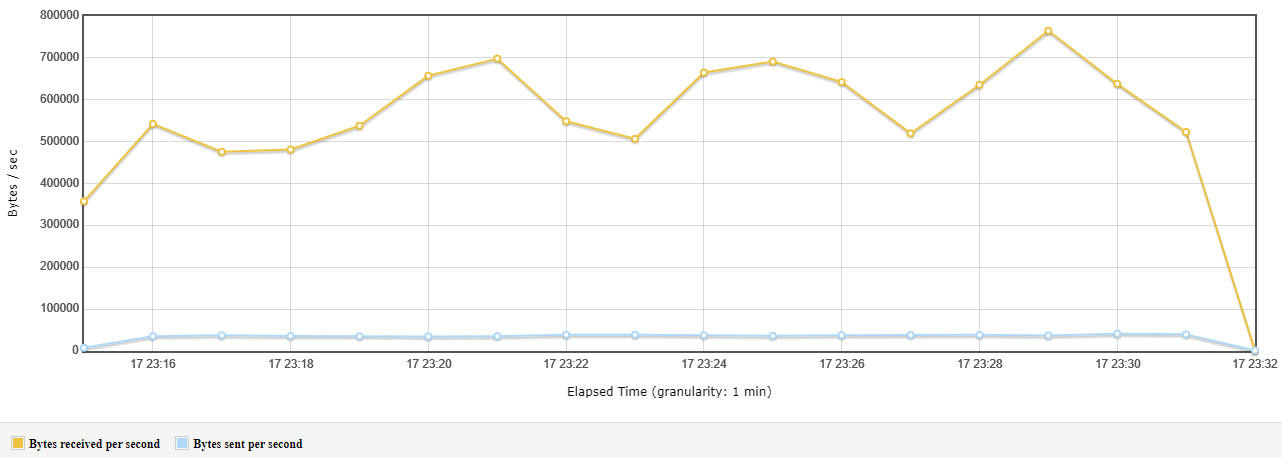
### Percentyle czasu odpowiedzi w czasie (udane odpowiedzi)



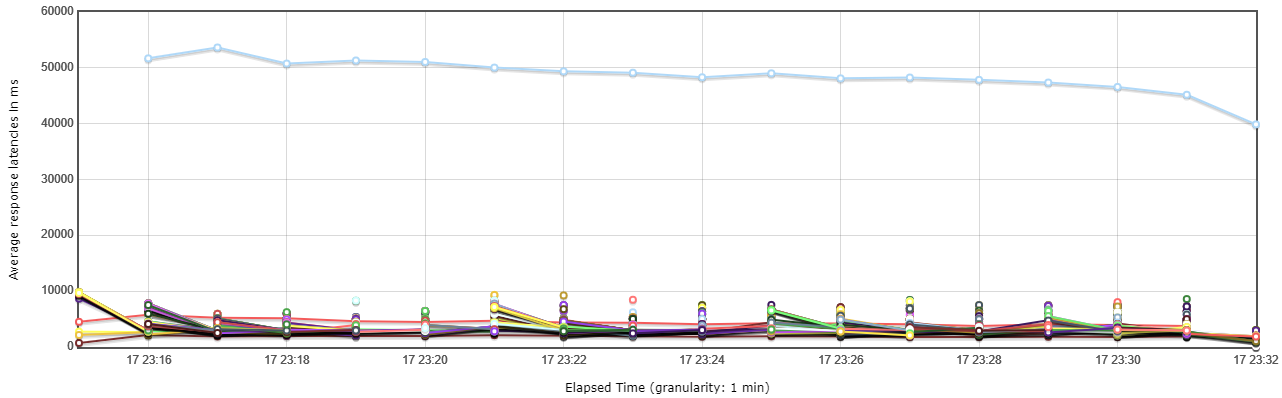
### Aktywne wątki w czasie



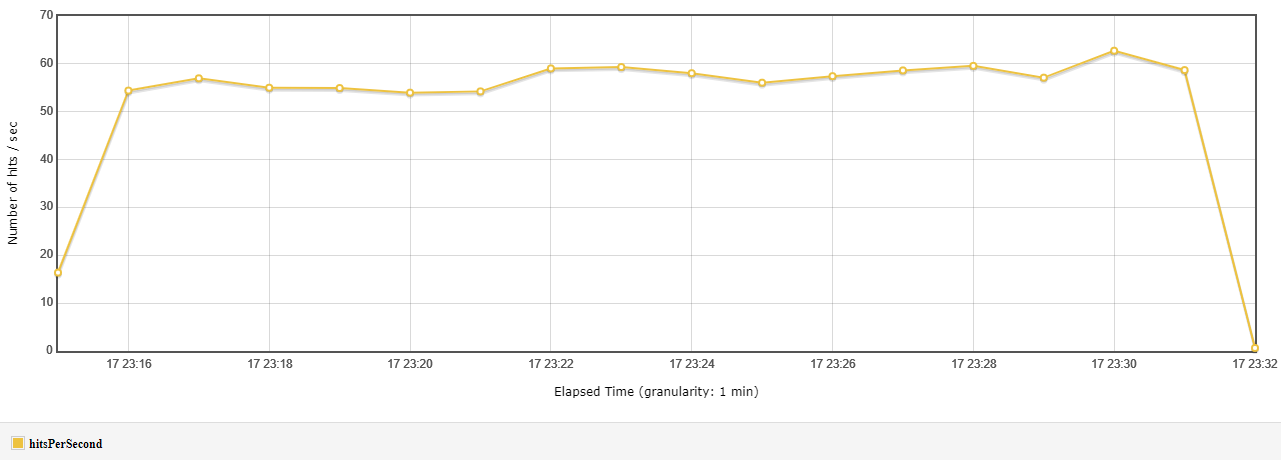
### Przepustowość bajtów w czasie



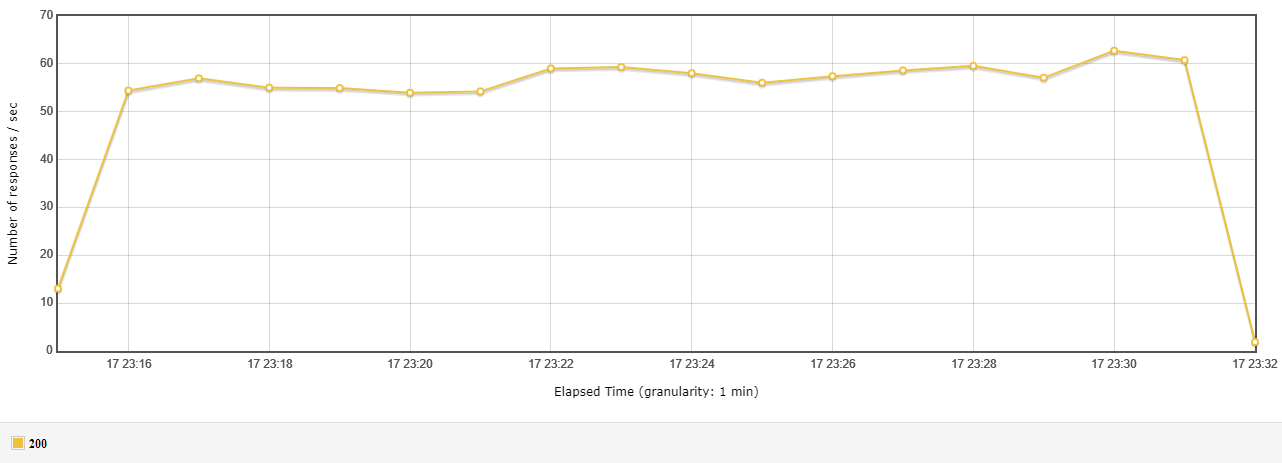
### Opóźnienia w czasie



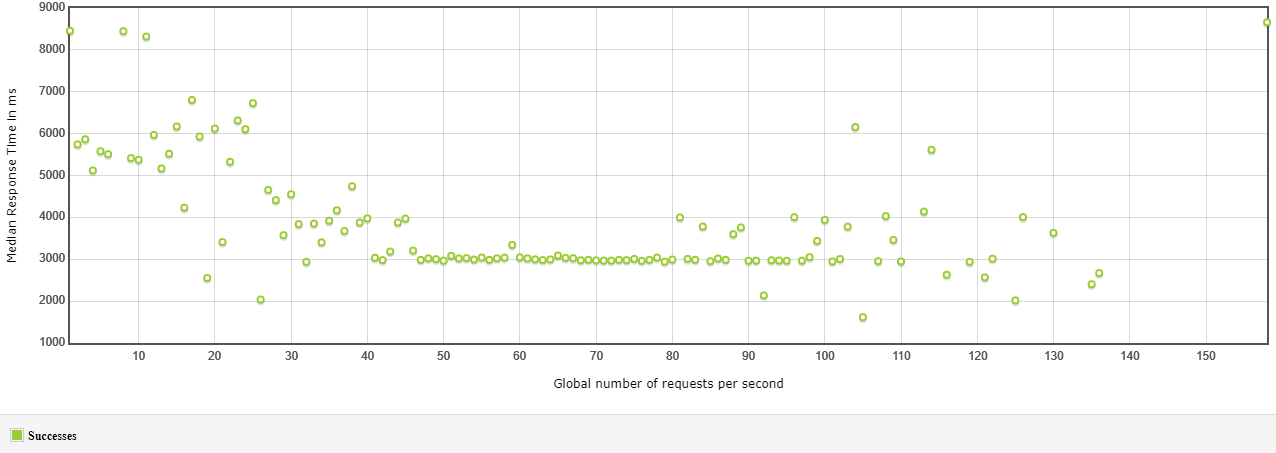
### Liczba trafień na sekundę



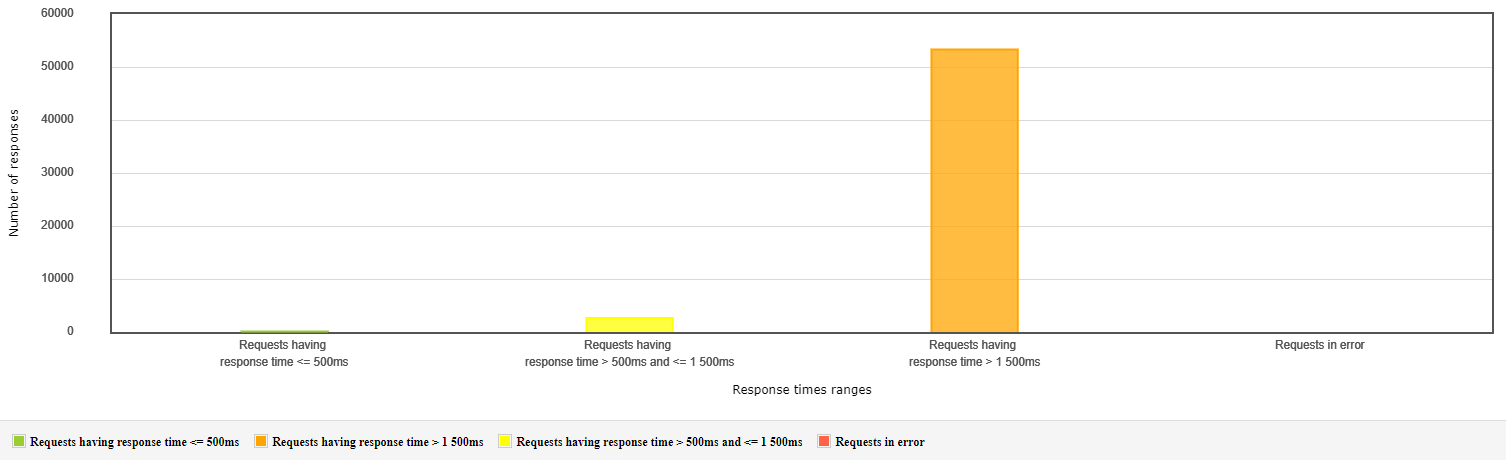
### Liczba odpowiedzi na sekundę



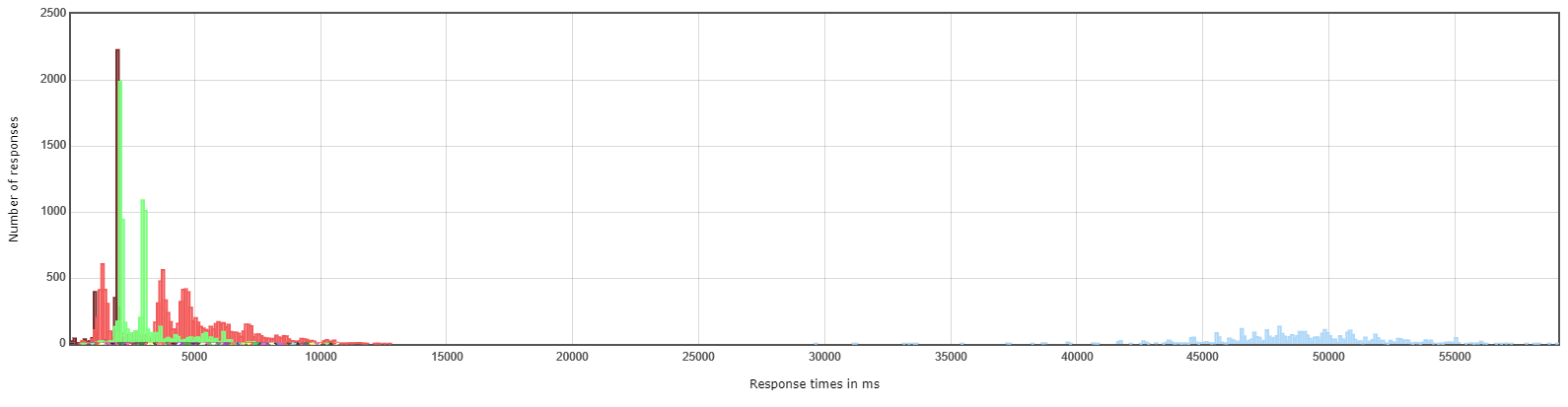
### Czas odpowiedzi vs żądania



### Przegląd czasu odpowiedzi



### Dystrybucja czasów odpowiedzi



## Wnioski z testów

Wydajność aplikacji utrzymywała się na stosunkowo stałym poziomie. Średni czas odpowiedzi dla zapytań pozostał na dobrym poziomie, co oznacza, że API jest wydajne i odpowiada na zapytania szybko. Testy nie wykazały błędów, zwracane kody zawsze były pozytywne, co oznacza, że API jest stabilne i nie ma problemów z jego działaniem. Testy wykazały, że API jest w stanie obsłużyć dużą liczbę jednoczesnych połączeń, co oznacza, że jest ono przystosowane do dużego ruchu. Testy wykazały, że API jest zgodne z wymaganiami dotyczącymi skalowalności i jest w stanie obsługiwać rosnące zapotrzebowanie na usługi.

## Artefakty i reprodukowanie wyników

Artefaktami przeprowadzonych testów wydajnościowych są 3 pliki:

* **tiwo123456.jmx** - to plik konfiguracyjny, który zawiera opis testu wydajnościowego. Ten plik opisuje cały plan testów, włącznie z informacjami o wątkach, żądaniach, listenerach oraz innych elementach testowych. Plik JMX jest formatem natywnym dla JMeter i jest używany do zapisywania i otwierania planów testów. Można go użyć do ponownego uruchomienia testów lub do dalszej edycji. Aby otworzyć plik w programie JMeter należy uruchomić program, a następnie Plik -> Otwórz.
* **tiwo123456\_results.jtl** - to plik, który zawiera dane z testów wydajnościowych. Jest on używany do przechowywania informacji o każdym żądaniu, odpowiedzi i ewentualnych błędach podczas testów. Plik JTL może być używany do generowania raportów i analizowania wyników testów. Na podstawie tego pliku można wygenerować raport HTML.
* **report/index.html** – plik z raportem z wynikami testów. Nie został dołączony do oddawanego projektu ze względu na swoją wielkość (278MB). Aby wygenerować raport należy: Tools -> Generate HTML report -> Wskazać
  + plik .jtl z danymi,
  + plik z preferencjami użytkownika (znajduje się w katalogu z programem JMeter - bin),
  + folder do którego zostanie wygenerowany raport.

Znajdują się one w folderze *performance-tests* w katalogu z projektem. W celu reprodukcji wyników, należy mieć zainstalowane narzędzie JMeter. Testy zostały przeprowadzone z użyciem programu w wersji 5.5.

# Testy bezpieczeństwa

## Użyte narzędzia

Do wykonania testów bezpieczeństwa zostało użyte narzędzie OWASP ZAP (Zed Attack Proxy). Po wykonaniu testu został wygenerowany raport HTML zawierający potencjalne zagrożenia.

## Procedura testowania

Zostało uruchomione narzędzie OWASP ZAP, przy uruchomionej aplikacji na localhost. Wykonane zostało automatyczne skanowanie, podając adres URL aplikacji. Podczas testu zostały włączone spidery tradycyjne i ajax.

## Uzyskane wyniki

Został wygenerowany raport HTML, który zawiera wszystkie wykryte odstępstwa od standardów bezpieczeństwa.

### Podsumowanie raportu

Obraz zawierający stół

Opis wygenerowany automatycznie

## Interpretacja wyników

Wystąpiły 4 błędy o poziomie ryzyka średni:

* **CSP: Wildcard Directive** - Ten błąd jest związany z Content Security Policy (CSP), która jest funkcją bezpieczeństwa, która pomaga wykrywać i zapobiegać pewnym typom ataków, takim jak ataki typu Cross Site Scripting (XSS) i wstrzykiwanie danych. Ten błąd informuje, że serwer WWW, serwer aplikacji lub moduł równoważenia obciążenia nie jest prawidłowo skonfigurowany do ustawiania nagłówka Content-Security-Policy. Rozwiązaniem jest odpowiednie skonfigurowanie serwera do ustawienia nagłówka.
* **Content Security Policy (CSP) Header Not Set** - Ten komunikat o błędzie wskazuje, że nagłówek Content Security Policy (CSP) nie jest ustawiony w witrynie Należy skonfigurować aplikację, aby ustawaić nagłówek CSP w celu zadeklarowania, które źródła treści mogą być ładowane w witrynie.
* **Cross-Domain Misconfiguration** - Ten komunikat o błędzie wskazuje, że na serwerze internetowym występuje błędna konfiguracja Cross-Origin Resource Sharing (CORS). CORS to funkcja bezpieczeństwa, która kontroluje sposób, w jaki przeglądarki internetowe obsługują żądania z innego źródła niż to, które obsłużyło stronę internetową. Serwer jest skonfigurowany tak, aby umożliwić dowolnemu pochodzeniu dostęp do zasobów na serwerze, o czym świadczy nagłówek „Access-Control-Allow-Origin: \*”. Ta konfiguracja może stanowić zagrożenie dla bezpieczeństwa, ponieważ może umożliwić ładowanie danych z serwera przez nieautoryzowane źródła. Rozwiązaniem tego błędu jest upewnienie się, że wrażliwe dane nie są dostępne w sposób nieuwierzytelniony, na przykład za pomocą białej listy adresów IP. Nagłówek HTTP „Access-Control-Allow-Origin” powinien być skonfigurowany do bardziej restrykcyjnego zestawu domen lub całkowicie usunąć wszystkie nagłówki CORS, aby umożliwić przeglądarce internetowej egzekwowanie zasad Same Origin Policy (SOP) w bardziej restrykcyjny sposób.
* **Missing Anti-clickjacking Header** - Ten komunikat o błędzie wskazuje, że odpowiedź z serwera WWW nie zawiera ani dyrektywy Content-Security-Policy with „frame-ancestors”, ani nagłówka X-Frame-Options chroniącego przed atakami ClickJacking. Clickjacking to rodzaj ataku, w którym osoba atakująca nakłania użytkownika do kliknięcia przycisku lub łącza na stronie, z którą nie zamierzał wchodzić w interakcje, za pomocą przezroczystej warstwy na górze strony. Aby rozwiązać ten problem, konieczne jest ustawienie jednego z tych nagłówków na wszystkich stronach internetowych zwracanych przez aplikację.

## Artefakty Testów

Szczegółowe wyniki z testów są dostępne w folderze *security-tests* w folderze projektu. Raport z plikiem raportu HTML znajduje się pod nazwą: **2023-01-17-ZAP-Report-localhost.html**.

## Wymagania środowiska testowego

* Instalacja OWASP ZAP. Podczas testów używana była wersja 2.12.0.