



Algorium

Motywacja i cel

Algorium jest platforma edukacyjną, którym celem jest ułatwienie nauki algorytmów i struktur danych. Usprawnienie nauki będzie polegać na rozwiązaniu jednego z głównych problemów, jakie mogą wystąpić podczas uczenia się algorytmów, między innymi trudne zrozumienie, jak dany algorytm zachowuje się w konkretnych operacjach.

Projekt *Algorium* uprości ten proces nauki za pomocą rozmaitych wizualizacji i animacji algorytmów oraz struktur danych. Użytkownik będzie mógł wylosować lub podać swoje własne dane wejściowe do algorytmu, które będą w zależności od danego algorytmu odpowiednio zwizualizowane na stronie. Dodatkowo użytkownik będzie miał pełną kontrolę nad operacjami danego algorytmu w postaci zatrzymywania w dowolnym kroku jaki wykonuje algorytm oraz podejrzenie jaka aktualnie linia kodu jest wykonywana. Taki proces nauki zdecydowanie ułatwi i na pewno uprzyjemni czas spędzony na nauce.

Opis projektu

Platforma edukacyjna będzie podzielona na moduły, czyli dany rodzaj algorytmów lub struktur danych. Każdy taki moduł będzie się składał z konkretnych algorytmów i struktur danych, które będą zawierać opis i swoją własną interaktywną wizualizację.

Lista modułów

Algorytmy sortujące

- Sortowanie bąbelkowe (Bubble sort)
- Sortowanie przez wstawianie (Insertion sort)
- Sortowanie szybkie (Quick sort)

Algorytmy grafowe

- Przeszukiwanie wszerz (Breadth-first search, BFS)
- Przeszukiwanie w głąb (Depth-first search, DFS)
- Dijkstra

Podstawowe Struktury danych

- Stos (Stack)
- Kolejka (Queue)
- Kopiec (Heap)

Listy

- Lista jednokierunkowa (Linked list)
- Lista dwukierunkowa (Doubly Linked list)

Quiz

Użytkownik będzie miał możliwość przechodzenia do kolejnego modułu po pozytywnym zaliczeniu quizu. Suma punktów będzie widoczna na górze strony oraz na liście modułów, gdzie będą one odpowiednio oznaczone (w trakcie nauki/zaliczone).

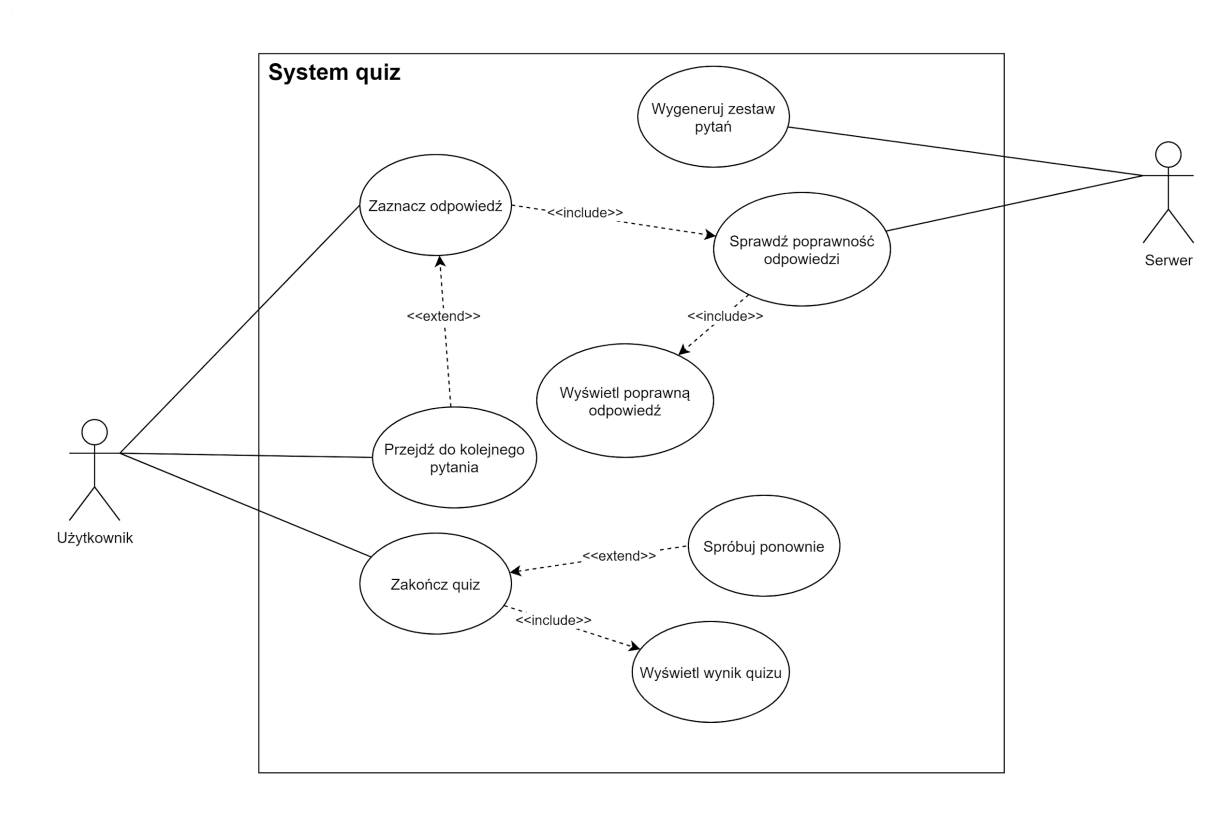


Diagram przypadku użycia systemu quiz

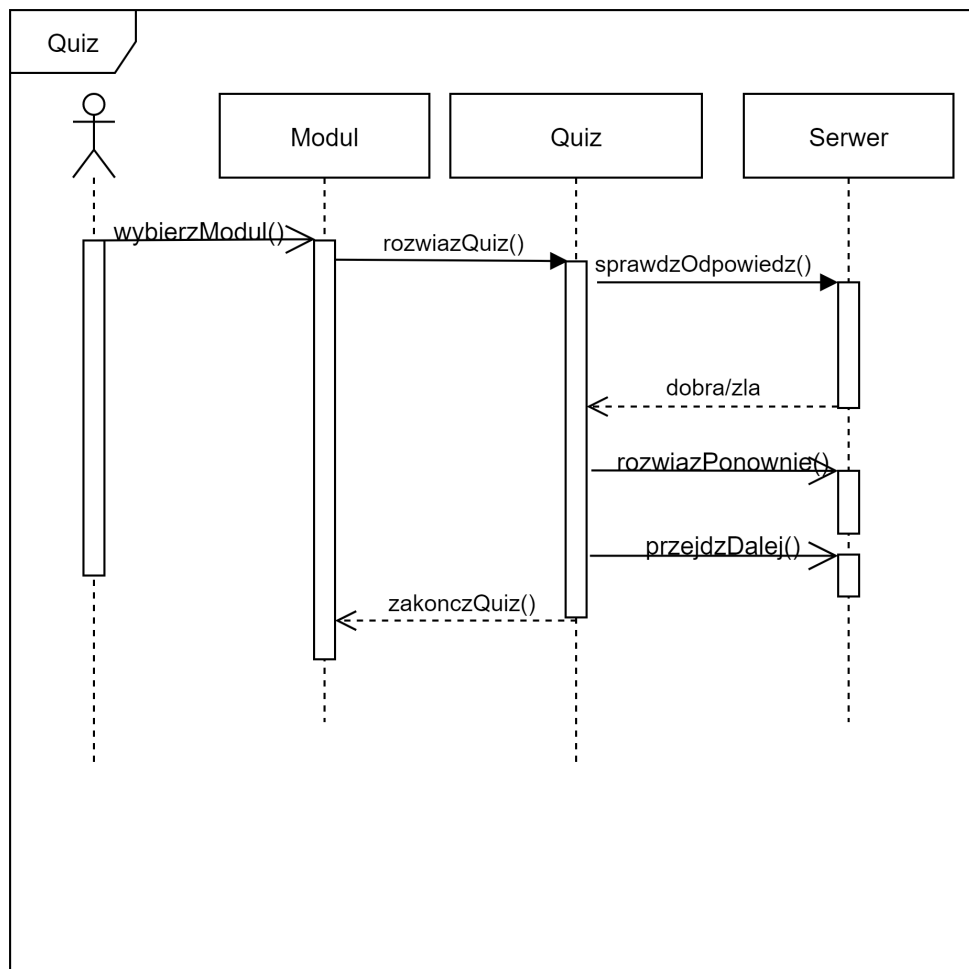


Diagram sekwencji systemu quiz

Rejestracja i logowanie

Użytkownik będzie mógł korzystać z platformy po uprzedniej rejestracji (nazwa użytkownika, email, hasło).

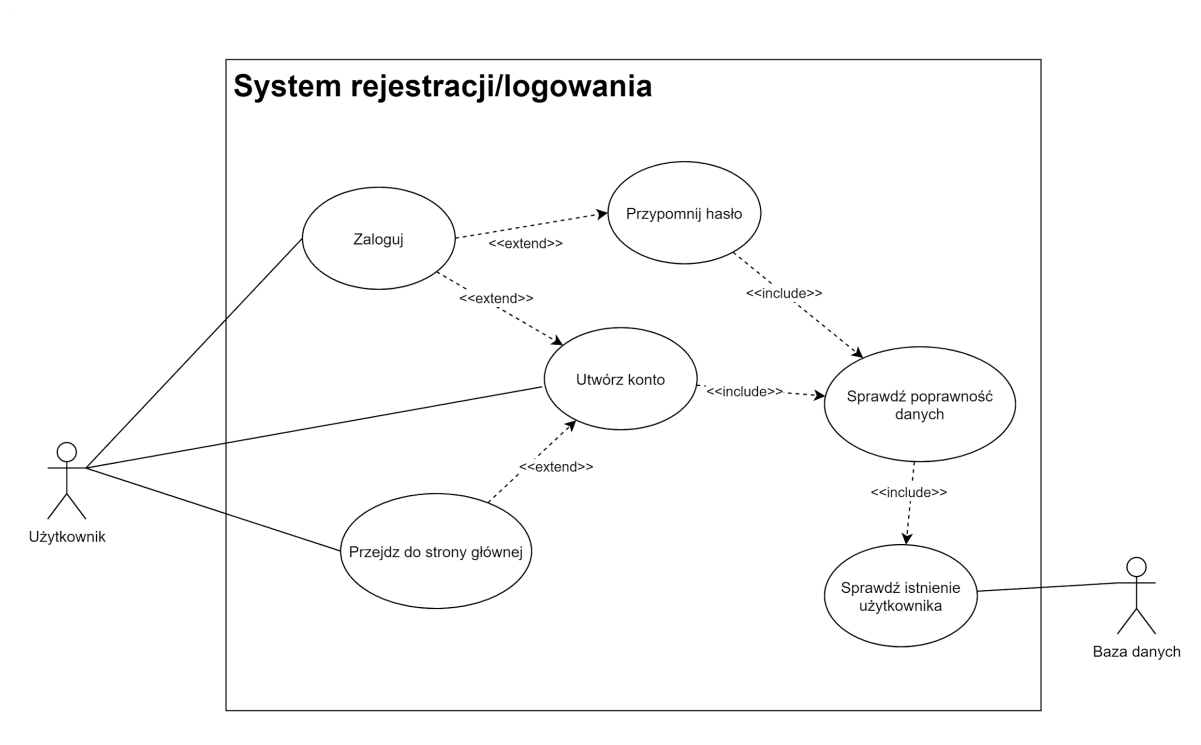


Diagram przypadku użycia systemu rejestracji i logowania

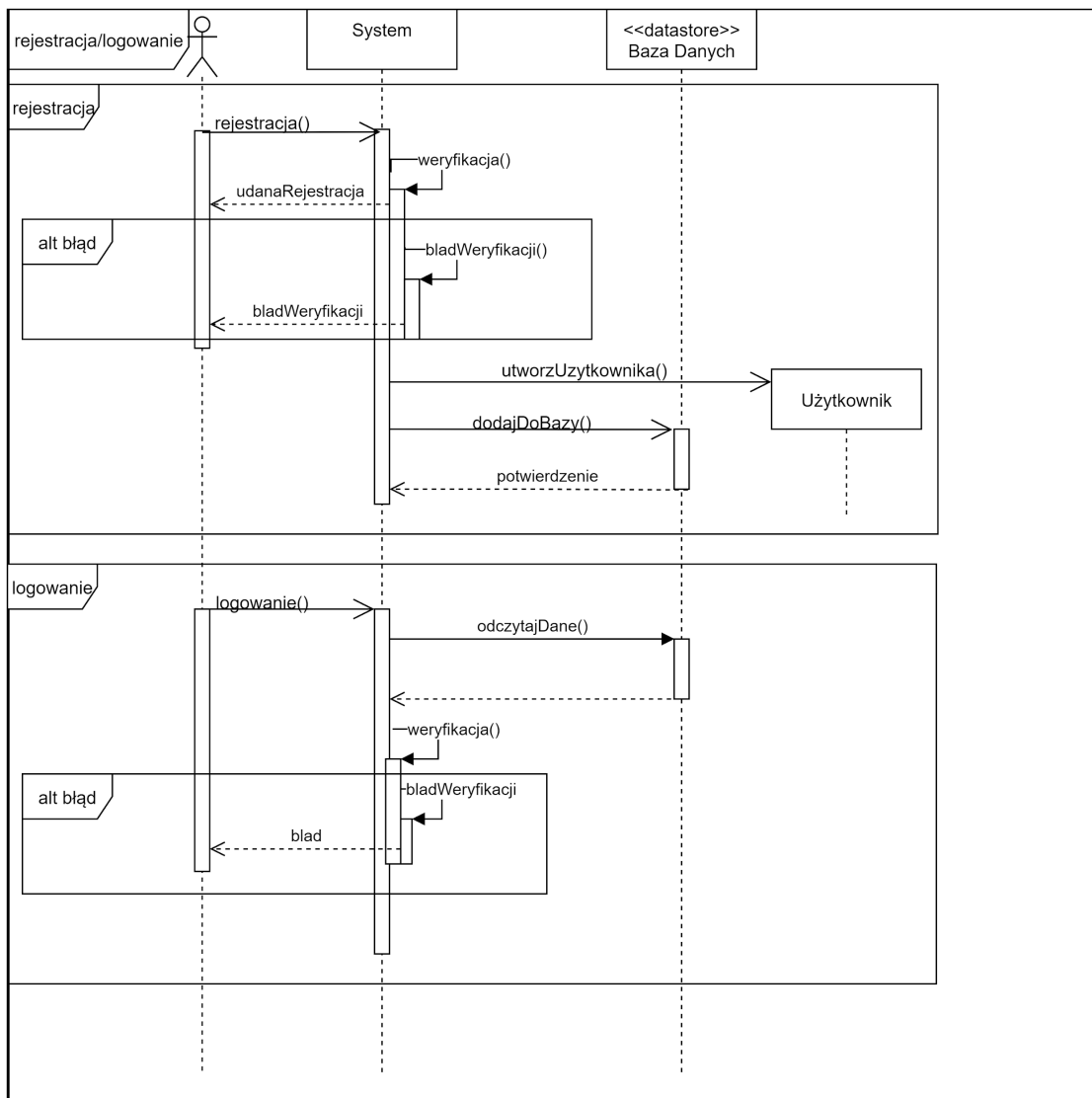


Diagram sekwencji systemu rejestracji i logowania

Wizualizacja algorytmów

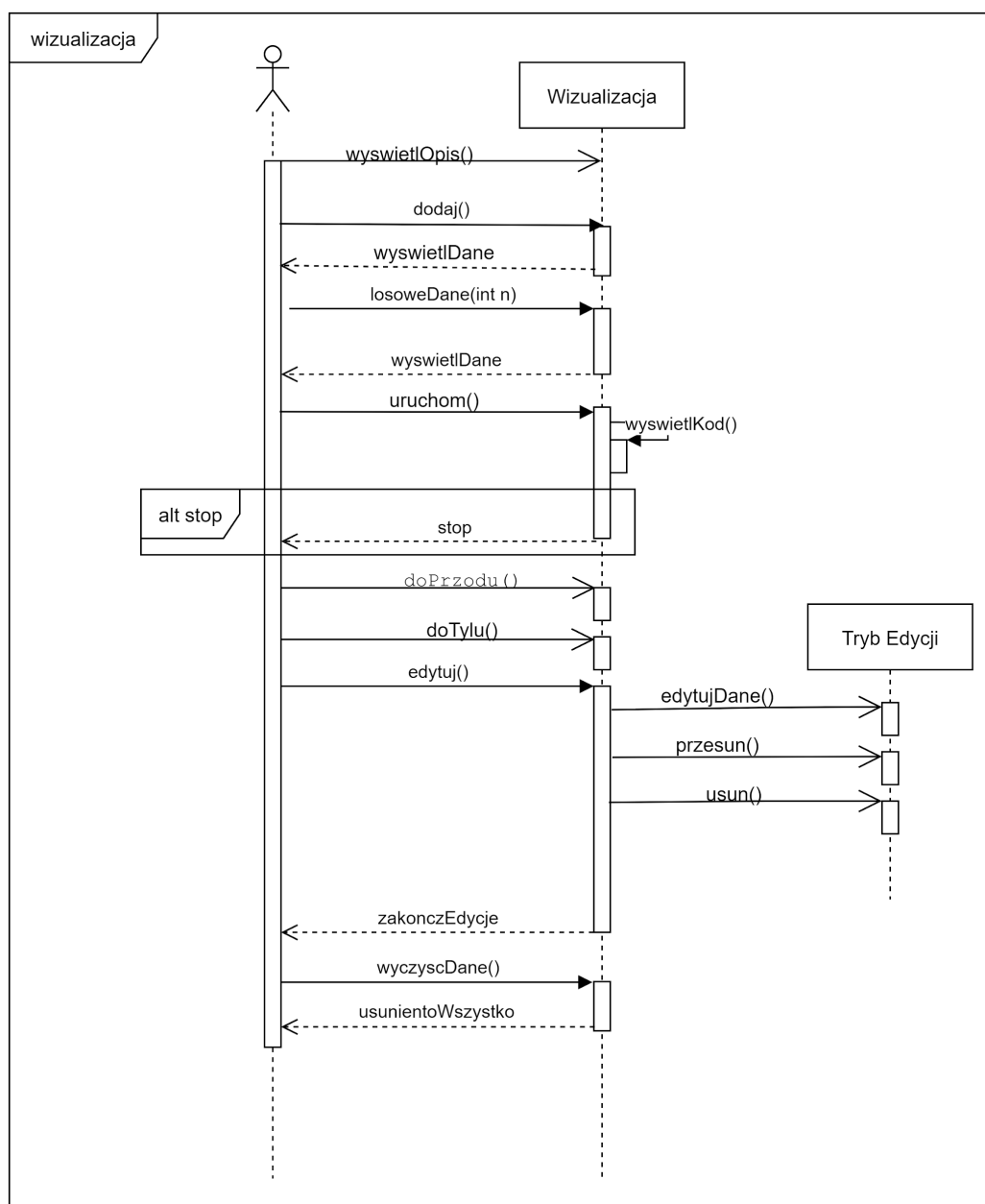


Diagram sekwencji systemu wizualizacji algorytmów

Scenariusze testowe

Rejestracja

Scenariusz testowy

Nazwa	Opis	Typ	Czynności przygotowawcze	Czynności końcowe
Rejestracja	Sprawdzenie poprawności działania rejestracji.	Testy funkcjonalne	Wejść na stronę rejestracji.	Utworzenie konta.

Przypadki testowe

Nazwa	Warunki wstępne	Kroki wykonania	Oczekiwany rezultat
Wprowadzenie prawidłowego identyfikatora e-mail i hasła.	Otwarcie strony z formularzem rejestracji.	1. Uzupełnienie pola nazwa użytkownika. 2. Uzupełnienie pola email. 3. Uzupełnienie pola hasło. 4. Kliknięcie w przycisk zarejestruj.	Na ekranie pojawia się komunikat o pomyślnym zarejestrowaniu.
Wprowadzenie prawidłowego identyfikatora e-mail i nieprawidłowego hasła.	Otwarcie strony z formularzem rejestracji.	1. Uzupełnienie pola nazwa użytkownika.. 2. Uzupełnienie pola email. 3. Pozostawienie pola hasło puste. 4. Kliknięcie w przycisk zarejestruj.	Na ekranie pojawia się komunikat o nieprawidłowo wprowadzonym hasle.

Wprowadzenie prawidłowego identyfikatora e-mail i nieprawidłowego hasła	Otwarcie strony z formularzem rejestracji.	<ol style="list-style-type: none"> 1. Uzpełnienie pola nazwa użytkownika. 2. Uzpełnienie pola email. 3. Uzpełnienie pola hasło ilością znaków mniejszą niż wymagana. 4. Kliknięcie w przycisk zarejestruj. 	Na ekranie pojawia się komunikat o nieprawidłowo wprowadzonym hasle.
Wprowadzenie nieprawidłowego identyfikatora e-mail i prawidłowego hasła.	Otwarcie strony z formularzem rejestracji.	<ol style="list-style-type: none"> 1. Uzpełnienie pola nazwa użytkownika. 2. Uzpełnienie pola email ciągiem znaków bez '@'. 3. Uzpełnienie pola hasło. 4. Kliknięcie w przycisk zarejestruj. 	Na ekranie pojawia się komunikat o nieprawidłowo wprowadzonym adresie email.
Wprowadzenie nieprawidłowego identyfikatora e-mail i prawidłowego hasła.	Otwarcie strony z formularzem rejestracji.	<ol style="list-style-type: none"> 1. Uzpełnienie pola nazwa użytkownika. 2. Pozostawienie pola email pustym. 3. Uzpełnienie pola hasło. 4. Kliknięcie w przycisk zarejestruj. 	Na ekranie pojawia się komunikat o nieprawidłowo wprowadzonym adresie email.
Wprowadzenie nieprawidłowego identyfikatora e-mail i nieprawidłowego hasła.	Otwarcie strony z formularzem rejestracji.	<ol style="list-style-type: none"> 1. Uzpełnienie pola nazwa użytkownika. 2. Pozostawienie pola email pustym. 3. Pozostawienia pola hasła pustym. 4. Kliknięcie w przycisk zarejestruj. 	Na ekranie pojawia się komunikat o nieprawidłowo wprowadzonym adresie email i hasle.
Nieprawidłowo wypełniony formularz rejestracyjny (niewłaściwe dane).	Otwarcie strony z formularzem rejestracji.	<ol style="list-style-type: none"> 1. Uzpełnienie pola nazwa użytkownika wartością liczbową. 2. Uzpełnienie pola email. 3. Uzpełnienie pola hasło. 4. Kliknięcie w przycisk zarejestruj. 	Na ekranie pojawia się komunikat o nieprawidłowo wprowadzonej nazwie użytkownika.

Logowanie

Scenariusz testowy

Nazwa	Opis	Typ	Czynności przygotowawcze	Czynności końcowe
Logowanie	Sprawdzenie poprawności działania funkcjonalności logowania.	Testy funkcjonalne.	Wejść na stronę logowania.	Zalogowanie się.

Przypadki testowe

Nazwa	Warunki wstępne	Kroki wykonania	Oczekiwany rezultat
Wprowadzenie prawidłowego identyfikatora e-mail i hasła.	1. Wejście na stronę logowania. 2. Posiadanie utworzonego konta.	1. Uzupełnienie pola email. 2. Uzupełnienie pola hasło. 3. Kliknięcie w przycisk zaloguj się.	Na ekranie pojawia się komunikat o pomyślnym zalogowaniu. Przekierowanie na stronę główną.
Wprowadzenie prawidłowego identyfikatora e-mail i nieprawidłowego hasła.	1. Wejście na stronę logowania. 2. Posiadanie utworzonego konta.	1. Uzupełnienie pola email. 2. Pozostawienie pola hasło puste.	Na ekranie pojawia się komunikat o nieprawidłowym adresie email lub hasle.

		3. Kliknięcie w przycisk zaloguj się.	
Wprowadzenie prawidłowego identyfikatora e-mail i nieprawidłowego hasła.	1. Wejście na stronę logowania. 2. Posiadanie utworzonego konta.	1. Uzupełnienie pola email. 2. Uzupełnienie pola hasło losowym ciągiem znaków. 3. Kliknięcie w przycisk zaloguj się.	Na ekranie pojawia się komunikat o nieprawidłowym adresie email lub haśle.
Wprowadzenie nieprawidłowego identyfikatora e-mail i prawidłowego hasła.	1. Wejście na stronę logowania. 2. Posiadanie utworzonego konta.	1. Uzupełnienie pola email ciągiem znaków bez '@'. 2. Uzupełnienie pola hasło. 3. Kliknięcie w przycisk zarejestruj.	Na ekranie pojawia się komunikat o nieprawidłowym adresie email lub haśle.
Wprowadzenie nieprawidłowego identyfikatora e-mail i prawidłowego hasła.	1. Wejście na stronę logowania. 2. Posiadanie utworzonego konta.	1. Pozostawienie pola email puste. 2. Uzupełnienie pola hasło. 3. Kliknięcie w przycisk zarejestruj.	Na ekranie pojawia się komunikat o nieprawidłowym adresie email lub haśle.
Wprowadzenie nieprawidłowego identyfikatora e-mail i prawidłowego hasła.	1. Wejście na stronę logowania. 2. Posiadanie utworzonego konta.	1. Uzupełnienie pola email losowym ciągiem znaków. 2. Uzupełnienie pola hasło. 3. Kliknięcie w przycisk zarejestruj.	Na ekranie pojawia się komunikat o nieprawidłowym adresie email lub haśle.
Wprowadzenie nieprawidłowego identyfikatora e-mail i nieprawidłowego hasła.	1. Wejście na stronę logowania. 2. Posiadanie utworzonego konta.	1. Uzupełnienie pola email ciągiem znaków bez '@'. 2. Uzupełnienie pola hasło. 3. Kliknięcie w przycisk zarejestruj.	Na ekranie pojawia się komunikat o nieprawidłowym adresie email lub haśle.

Wprowadzenie nieprawidłowego identyfikatora e-mail i nieprawidłowego hasła.	1. Wejście na stronę logowania. 2. Posiadanie utworzonego konta.	1. Pozostawienie pola email puste. 2. Pozostawienie pola hasło puste. 3. Kliknięcie w przycisk zarejestruj.	Na ekranie pojawia się komunikat o nieprawidłowym adresie email lub hasle.
Wprowadzenie nieprawidłowego identyfikatora e-mail i nieprawidłowego hasła.	1. Wejście na stronę logowania. 2. Posiadanie utworzonego konta.	1. Pozostawienie pola email pustym. 2. Uzupełnienie pola hasło. 3. Kliknięcie w przycisk zarejestruj.	Na ekranie pojawia się komunikat o nieprawidłowym adresie email lub hasle.
Wprowadzenie nieprawidłowego identyfikatora e-mail i nieprawidłowego hasła.	1. Wejście na stronę logowania. 2. Posiadanie utworzonego konta.	1. Uzupełnienie pola email. 2. Pozostawienie pola hasło puste. 3. Kliknięcie w przycisk zarejestruj.	Na ekranie pojawia się komunikat o nieprawidłowym adresie email lub hasle.
Wprowadzenie nieprawidłowego identyfikatora e-mail i nieprawidłowego hasła.	1. Wejście na stronę logowania. 2. Posiadanie utworzonego konta.	1. Uzupełnienie pola email losowym ciągiem znaków. 2. Uzupełnienie pola hasło losowym ciągiem znaków. 3. Kliknięcie w przycisk zarejestruj.	Na ekranie pojawia się komunikat o nieprawidłowym adresie email lub hasle.
Weryfikacja czy funkcja "nie pamiętasz hasła" działa zgodnie z oczekiwaniami.	1. Wejście na stronę logowania. 2. Posiadanie utworzonego konta.	1. Kliknięcie w przycisk zapomniałem hasła. 2. Uzupełnienie adresu email. 3. Kliknięcie w przycisk przypomnij hasło.	Na ekranie pojawia się komunikat o wysłanej na adres email wiadomości z odzyskiwaniem hasła.
Weryfikacja czy funkcja "nie pamiętasz hasła" działa zgodnie z oczekiwaniami.	1. Wejście na stronę logowania. 2. Posiadanie utworzonego konta.	1. Kliknięcie w przycisk zapomniałem hasła. 2. Wprowadzenie niepoprawnego maila. 3. Kliknięcie w przycisk przypomnij hasło.	Na ekranie pojawia się komunikat o tym, że konto o takim adresie email nie istnieje.

Quiz

Scenariusz testowy

Nazwa	Opis	Typ	Czynności przygotowawcze	Czynności końcowe
Moduł quiz	Sprawdzenie poprawności działania quizu.	Testy funkcjonalne	Zalogowanie się. Wybranie opcji rozwiązanie quizu z listy modułów.	Skończenie quizu.

Przypadki testowe

Nazwa	Warunki wstępne	Kroki wykonania	Oczekiwanie rezultat
Odpowiedzenie na pytanie w prawidłowy sposób.	1. Zalogowanie się na konto. 3. Przejście do listy modułów. 2. Wybranie opcji quiz w module.	1. Zaznaczenie prawidłowej odpowiedzi. 2. Kliknięcie w przycisk przejdź dalej.	Przyznanie punktów i przejście do następnego zadania lub zakończenie i podsumowanie quizu.

Odpowiedzenie na pytanie w nieprawidłowy sposób.	1. Zalogowanie się na konto. 3. Przejście do listy modułów. 2. Wybranie opcji quiz w module.	1. Zaznaczenie nieprawidłowej odpowiedzi. 2. Kliknięcie w przycisk przejdź dalej.	Na ekranie pojawia się komunikat o błędnej odpowiedzi.
Odpowiedzenie na pytanie w nieprawidłowy sposób.	1. Zalogowanie się na konto. 2. Przejście do listy modułów. 3. Wybranie opcji quiz w module.	1. Wpisanie losowego ciągu znaków. 2. Kliknięcie w przycisk przejdź dalej.	Na ekranie pojawia się komunikat o błędnej odpowiedzi.
Brak zaznaczonej/uzupełnionej odpowiedzi.	1. Zalogowanie się na konto. 3. Przejście do listy modułów. 2. Wybranie opcji quiz w module.	1. Kliknięcie w przycisk przejdź dalej. 2. Kliknięcie w przycisk przejdź dalej.	Na ekranie pojawia się komunikat o konieczności zaznaczenia lub uzupełnienia odpowiedzi.
Wyjście z aplikacji podczas quizu	1. Zalogowanie się na konto. 3. Przejście do listy modułów. 2. Wybranie opcji quiz w module.	1. Zamknięcie karty przeglądarki.	Na ekranie pojawi się komunikat, że po wyłączeniu karty przeglądarki postęp w quizie zostanie utracony.

Wizualizacja algorytmów

Scenariusz testowy

Nazwa	Opis	Typ	Czynności przygotowawcze	Czynności końcowe
Wizualizacja algorytmów	Sprawdzenie poprawności systemu wizualizacji algorytmów.	Testy funkcjonalne.	Zalogowanie się. Przejsie do strony z wizualizacją algorytmów.	Zakończenie wizualizacji algorytmu i powrót do strony głównej.

Przypadki testowe

Nazwa	Warunki wstępne	Kroki wykonania	Oczekiwany rezultat
-------	-----------------	-----------------	---------------------

Wprowadzenie poprawnych danych	1. Zalogowanie się na konto. 2. Przejście do strony z wizualizacją algorytmów.	1.Podanie całkowitej liczby losowych danych. 2. Kliknięcie przycisku "losowe dane".	W oknie wizualizacji wyświetlają się losowe dane oraz przycisk "dodaj" jest zablokowany. Zablokowana możliwość dodawania nowych danych bez trybu edycji.
Wprowadzenie poprawnych danych	1. Zalogowanie się na konto. 2. Przejście do strony z wizualizacją algorytmów.	1. Wprowadzenie liczby lub litery w polu przeznaczonym na dodanie własnych danych. 2. Kliknięcie przycisku "Dodaj".	W oknie wizualizacji wyświetlają się wprowadzone dane oraz przycisk "losowe dane" jest zablokowany. Zablokowana możliwość losowania danych.
Wprowadzenie błędnych danych.	1. Zalogowanie się na konto. 2. Przejście do strony z wizualizacją algorytmów.	1.Podanie zmiennoprzecinkowej liczby losowych danych. 2. Kliknięcie przycisku "losowe dane".	System powinien usunąć tekst i pozostawić pole pustym i wyświetlić komunikat informujący o tym, że użytkownik powinien wpisać liczbę od 1 do 20.
Wprowadzenie błędnych danych.	1. Zalogowanie się na konto. 2. Przejście do strony z wizualizacją algorytmów.	1.Wprowadzenie tekstu w polu oznaczającym liczbę losowych danych. 2. Kliknięcie przycisku "losowe dane".	System powinien usunąć tekst i pozostawić pole pustym i wyświetlić komunikat informujący o tym, że użytkownik powinien wpisać liczbę od 1 do 20.
Wprowadzenie błędnych danych.	1. Zalogowanie się na konto. 2. Przejście do strony z wizualizacją algorytmów.	1.Pozostawienie pola oznaczającego liczbę losowych danych pustym. 2. Kliknięcie przycisku "losowe dane".	System powinien poprosić o wprowadzenie liczby w zakresie od 1 do 20.
Wprowadzenie błędnych danych.	1. Zalogowanie się na konto. 2. Przejście do strony z wizualizacją algorytmów.	1.Pozostawienie pola przeznaczonego na dodanie własnych danych pustym.	System powinien poprosić o uzupełnienie pola przeznaczonego na dodanie własnych danych.

		2. Kliknięcie przycisku "Dodaj".	
Wprowadzenie błędnych danych.	1. Zalogowanie się na konto. 2. Przejście do strony z wizualizacją algorytmów.	1. Wypełnienie pola przeznaczonego na dodanie własnych danych losowym ciągiem znaków. 2. Kliknięcie przycisku "Dodaj".	System powinien usunąć tekst i pozostawić pole pustym i wyświetlić komunikat informujący o tym, że wprowadzone dane są niepoprawne.
Sprawdzenie opisu.	1. Zalogowanie się na konto. 2. Przejście do strony z wizualizacją algorytmów.	1. Kliknięcie w przycisk opis.	System powinien wyświetlić opis danego algorytmu.
Sprawdzenie działania przycisków odpowiedzialnych za odtwarzanie algorytmu.	1. Zalogowanie się na konto. 2. Przejście do strony z wizualizacją algorytmów.	1. Dodanie danych. 2. Kliknięcie w przycisk play.	W oknie wizualizacji powinna uruchomić się wizualizacja algorytmu oraz przyciski "dodaj", "losowe dane" i "edytuj" są zablokowane.
Sprawdzenie działania przycisków odpowiedzialnych za odtwarzanie algorytmu.	1. Zalogowanie się na konto. 2. Przejście do strony z wizualizacją algorytmów.	1. Dodanie danych. 2. Kliknięcie w przycisk play. 3. Kliknięcie w przycisk stop.	W oknie wizualizacji powinna zatrzymać się wizualizacja algorytmu.
Sprawdzenie działania przycisków odpowiedzialnych za odtwarzanie algorytmu.	1. Zalogowanie się na konto. 2. Przejście do strony z wizualizacją algorytmów.	1. Dodanie danych. 2. Kliknięcie w przycisk play. 3. Kliknięcie w przycisk stop. 4. Kliknięcie w przycisk stop.	System nie powinien reagować.

Sprawdzenie działania przycisków odpowiedzialnych za odtwarzanie algorytmu.	1. Zalogowanie się na konto. 2. Przejście do strony z wizualizacją algorytmów.	1. Dodanie danych. 2. Kliknięcie w przycisk stop.	System nie powinien reagować.
Sprawdzenie działania przycisków odpowiedzialnych za odtwarzanie algorytmu.	1. Zalogowanie się na konto. 2. Przejście do strony z wizualizacją algorytmów.	1. Dodanie danych. 2. Kliknięcie w przycisk play. 3. Kliknięcie w przycisk play.	System nie powinien reagować.
Sprawdzenie działania trybu edycji.	1. Zalogowanie się na konto. 2. Przejście do strony z wizualizacją algorytmów.	1. Dodanie danych. 2. Kliknięcie w przycisk edytuj.	Okno wizualizacji przejdzie w tryb edycji, w którym można będzie edytować, dodawać i usuwać dane.
Sprawdzenie działania trybu edycji.	1. Zalogowanie się na konto. 2. Przejście do strony z wizualizacją algorytmów.	1. Dodanie danych. 2. Kliknięcie w przycisk edytuj. 3. Kliknięcie w przycisk edytuj element. 4. Edycja elementu.	Okno wizualizacji przejdzie w tryb edycji. Po edycji wartości elementu i zatwierdzeniu nowa wartość powinna zastąpić poprzednią.
Sprawdzenie działania trybu edycji.	1. Zalogowanie się na konto. 2. Przejście do strony z wizualizacją algorytmów.	1. Dodanie danych. 2. Kliknięcie w przycisk edytuj. 3. Kliknięcie w przycisk usuwania.	Okno wizualizacji przejdzie w tryb edycji. Po kliknięciu w przycisk usuń element powinien zostać usunięty.

Sprawdzenie działania trybu edycji.	<ol style="list-style-type: none">1. Zalogowanie się na konto.2. Przejście do strony z wizualizacją algorytmów.	<ol style="list-style-type: none">1. Dodanie danych.2. Kliknięcie w przycisk edytuj.3. Kliknięcie w przycisk dodawania	Okno wizualizacji przejdzie w tryb edycji. Po kliknięciu w przycisk dodawania można będzie utworzyć nowy element.
-------------------------------------	--	--	---

Technologie i motywacja

Frontend

React

React to najpopularniejsza front-endowa biblioteka JavaScript typu open source do tworzenia aplikacji internetowych. Służy do obsługi warstwy widoku dla aplikacji internetowych i mobilnych. Do właściwości Reacta możemy odnieść jego prostotę, łatwość w testowaniu oraz zrozumieniu, co umożliwia szybką i wygodną pracę.

Motywacja:

- jego komponenty mogą być wielokrotnie używane
- jest stabilny, stoi za nim spora społeczność, cały czas się rozwija
- pozwala na zbudowanie dynamicznego interfejsu

Material-UI

Material-UI to biblioteka, która pozwala importować i używać różnych komponentów do tworzenia interfejsu użytkownika w aplikacjach React. Gotowe komponenty zapewnią nam spójność wyglądu całej aplikacji.

Motywacja:

- Ma ogromną, gotową kolekcję powszechnie używanych komponentów interfejsu użytkownika.
- Material UI umożliwia uniknięcie typowych błędów interfejsu użytkownika i tworzenie projektów bezbłędnych.
- Pomaga w tworzeniu spójnego interfejsu użytkownika.
- Material UI zyskał dużą popularność, oznacza to, że istnieje wiele samouczków i grup wsparcia dostępnych dla programistów React js korzystających z interfejsu Material UI do tworzenia projektów frontend.
- Składniki interfejsu Material UI można również w dużym stopniu dostosować.

Backend

Django

Django jest open-sourcowym frameworkiem umożliwiającym tworzenie w szybki i łatwy sposób webowych aplikacji. W Django dostarczane jest wiele gotowych funkcjonalności, które są często wykorzystywane w webowych aplikacjach, między innymi ORM czy autoryzacja, które ułatwią nam pisanie całego backendu.

Motywacja:

- Kod napisany w Django cechuje się dużą czytelnością, framework Django jest wyposażony w dobrą dokumentację. Jest stale rozwijany, udoskonalony, poprawiany, testowany.
- Django jest frameworkiem, który pozwala o wiele łatwiej i efektywniej kontrolować dane oraz zachować spójność danych.
- Zapewnia kontrolę wersji bazy danych, wsparcie procesów uwierzytelniania, silnik szablonów i widoków, caching oraz routing URL.

Django REST Framework

Django REST Framework, które zapewni nam obsługę REST API do tworzenia nowych użytkowników i naszych quizów.

Motywacja:

- Prostota, elastyczność, jakość i pokrycie testowe kodu źródłowego.
- Silnik serializacji zgodny zarówno ze źródłami danych ORM, jak i innymi niż ORM.
- Podłączane i łatwe do dostosowania emitery, parsery, walidatory i uwierzytelniacze.
- Przejrzyste, proste widoki zasobów, wykorzystujące nowe widoki oparte na klasach Django.

PostgreSQL

PostgreSQL to potężny, otwarty, obiektowo-relacyjny system baz danych, poza tym cieszy się doskonałą reputacją dzięki swojej sprawdzonej architekturze, niezawodności, integralności danych oraz bogatemu zestawowi funkcji. Django ma również wbudowaną integrację z PostgreSQL.

Test

Selenium

Selenium jest frameworkiem, który pozwala automatyzować czynności wykonywane za pomocą przeglądarki internetowej. Jest to wygodne narzędzie służące do skutecznego i szybkiego przeprowadzania testów automatycznych. Wykorzystanie tego narzędzia zapewni nam oszczędności czasu, który będziemy mogli efektywnie spożytkować na inne zadania.

Motywacja:

- Umożliwia nagrywanie i odtwarzanie w celu testowania aplikacji internetowych i może uruchamiać wiele skryptów w różnych przeglądarkach.

Burp Suite

Burp pozwala kontrolować i analizować ruch przesyłany między przeglądarką a serwerem WWW. Zawiera w sobie kilka dobrze współpracujących ze sobą narzędzi, które dają bardzo duże możliwości testowania również złożonych aplikacji webowych. Narzędzie pozwoli nam na przykład na śledzenie i analizę błędów w odpowiedzi serwera na złośliwie zmodyfikowane zapytania.

Devops

GitLab

GitLab to platforma DevOpsowa, wersja community jest open-source, zezwala na rozmiar repozytorium do 10gb, co dla naszego projektu będzie spokojnie wystarczające. Dodatkowo serwis udostępnia toole do CI/CD, przy wielkości naszego projektu powinny być wystarczające, nie ma sensu używać oddzielnych dedykowanych tooli jak np jenkins.

Heroku

Do zdeployowania naszej aplikacji webowej użyjemy serwisu Heroku, wspiera on używane przez nas technologie: Javascript, Python oraz PostgreSQL. Do naszych zastosowań póki co zakładamy, że wystarczy nam free tier. Dodatkowo deployowanie web aplikacji stworzonych przy użyciu naszych technologii na tym serwisie jest bardzo dobrze udokumentowane w sieci.

Proces CI/CD

1. Source control

- a. Każdy merge request (push na branch development) na GitLab musi dostać +1 zanim może zostać zmergowany do main'a

2. Build

- a. Na kodzie zostaje uruchomiony linter, który sprawdza go pod kątem formatowania
- b. nowa wersja repozytorium zostaje zbuildowana -> failure w tym etapie uniemożliwia merge commita

3. Test








- a. Na zbuildowanym repo odpalane są unit testy, (testy regresyjne o ile będą takie)

4. Deploy









- a. Jeżeli commit przeszedł wszystkie etapy to zostaje on zmergowany, a nowa wersja repozytorium zostaje zdeployowana na heroku




Spis kroków milowych





[M1] Przygotowanie projektu, jiry

- ▼  AV-16 [M1] Wstępne przygotowanie do projektu
 -  ~~AV-243~~ Wybranie technologii
 -  ~~AV-31~~ Przygotowanie repozytorium projektu
 -  ~~AV-17~~ Opis zawartości projektu
 -  ~~AV-28~~ Motywacja i cel powstania projektu
 -  ~~AV-18~~ Określenie ról
 -  ~~AV-273~~ Określenie środowisk

[M2] Diagramy, zawartość strony


-  AV-34 [M2] Design
 -  ~~AV-91~~ Utworzenie projektu na Figmie
 -  ~~AV-38~~ Strona wyświetlająca wizualizacje algorytmów
 -  ~~AV-125~~ Typy pytań w quizie
 -  ~~AV-39~~ Strona wyświetlająca quiz do danego modułu
 -  ~~AV-37~~ Lista modułów
 -  ~~AV-36~~ Logowaniem/rejestracją
 -  ~~AV-35~~ Logo projektu

-  AV-43 [M2] Diagramy
 -  ~~AV-46~~ Diagram przypadków użycia
 -  ~~AV-113~~ Diagramy sekwencji

-  AV-86 [M2] Zawartość strony
 -  ~~AV-88~~ Typy pytań i odpowiedzi
 -  ~~AV-74~~ Opis algorytmów
 -  ~~AV-75~~ Opis struktur danych

[M3] Scenariusze testowe, schemat bazy danych, poprawki

AV-49 [M3] Baza danych

 ~~AV-56~~ Schemat bazy danych

AV-115 [M3] Scenariusze testowe


 ~~AV-135~~ Scenariusze testowe dla wizualizacji algorytmów

 ~~AV-134~~ Scenariusze testowe dla quizu

 ~~AV-130~~ Scenariusze testowe dla logowania

 ~~AV-127~~ Scenariusze testowe dla rejestracji

AV-143 [M3] Poprawki

 ~~AV-259~~ Poprawki do dokumentacji

 ~~AV-262~~ Poprawki do designu

[M4] Inicjalizacja projektów, wizualizacje algorytmów, quiz

✚ AV-202 [M4] Podstawowe elementy projektu strony

- 📖 AV-203 Inicjalizacja projektu po stronie frontu
- 📖 AV-204 Inicjalizacja projektu po stronie backendu

✚ AV-179 [M4] Wizualizacje algorytmów

- 📖 AV-189 Strony do wyświetlania wizualizacji algorytmów
- 📖 AV-248 Wizualizacja algorytmów sortujących
- 📖 AV-249 Wizualizacja algorytmów grafowych
- 📖 AV-250 Wizualizacja podstawowych struktur danych
- 📖 AV-251 Wizualizacja list

✚ AV-252 [M4] Quiz

- 📖 AV-190 System quiz
- 📖 AV-253 Komponenty typów pytań

[M5] Użytkownicy, pozostałe elementy aplikacji

✚ AV-176 [M5] Użytkownicy

- 📖 AV-186 Formularz logowania
- 📖 AV-187 Formularz rejestracji

✚ AV-274 [M5] Pozostałe elementy aplikacji

- 📖 AV-185 Strona główna
- 📖 AV-188 Lista modułów

[M6] Dodatkowe funkcjonalności

✚ AV-210 [M6] Dodatkowe funkcjonalności

- 📖 AV-211 Przypomnienie hasła

Wersjonowanie

Wersje naszego produktu będą w następującym formacie **A.B-NazwaSprintu**

Gdzie **A** - major version (1.0-NazwaSprintu) ma być minimalną wersją gotową przedstawienia na ocene

B - minor version - przewidujemy B in (1,2,3) odpowiadające milestoneom z drugiego semestru

NazwaSprintu - jako iż mamy zamiar stosować CD, buildy naszego produktu będą opierać się o sprinty - tj każdy sprint będzie zakończony nową wersją software'u.

Wycena zadań

Wycenę historyjek będziemy opierać na podstawie ciągu fibonacciego (1, 2, 3, 5, 8, 13, 21). Estymowanie będzie polegać na wspólnym głosowaniu do każdej historyjki podczas planowania (*planning poker*) oraz na podstawie podobnie wykonanych zadań z poprzednich sprintów.

Środowiska

Przy projekcie zapewnimy trzy środowiska:

Dev - Wprowadzanie najnowszych funkcjonalności oraz poprawek.

Test - Wykrywanie ewentualnych bugów, testy automatyczne.

Prod - Stabilne środowisko, które już będzie gotowe do użycia przez użytkowników.

Priorytetyzacja

Do priorytetyzacji zadań zastosujemy metodę MoSCoW. Naszą listę priorytetów będziemy się kierować oraz na bieżącą ją aktualizować dostosowując się do zmian, które zajądą podczas rozwoju projektu.

Must have

- Wizualizacje algorytmów
- Quiz
- Użytkownicy

Should have

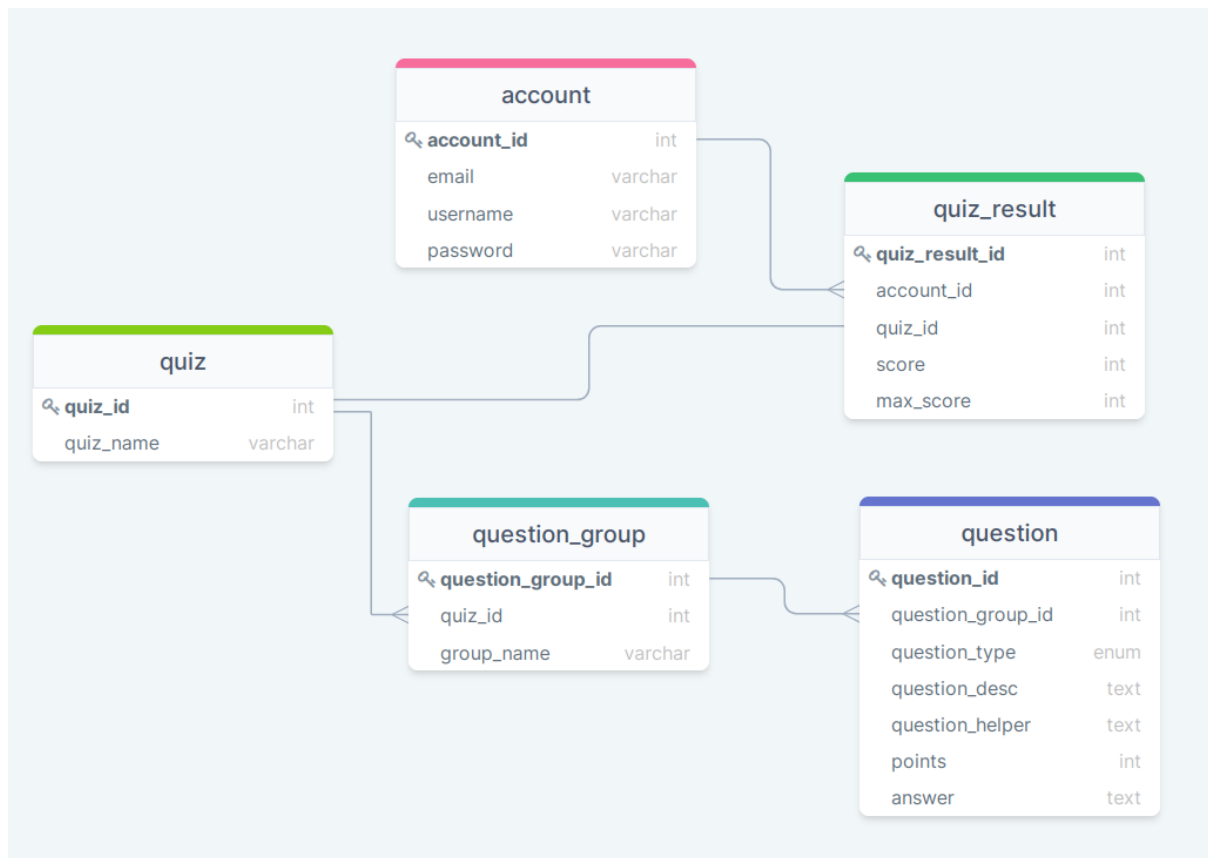
- Strona główna

Could have

- Przypominanie hasła

Won't have

Schemat bazy danych



Baza danych w naszym przypadku jest potrzebna ze względu na możliwość utworzenia konta w naszej aplikacji oraz za sprawą quizów.

W tabeli **account** będą znajdowały się podstawowe dane użytkowników - email, nazwa użytkownika i hasło. W tabeli **quiz** będą znajdować się nazwy quizów. W tabeli **question_group** będą konkretne nazwy algorytmów i struktur danych, podzielone odpowiednio do danego quizu. Pytania w tabeli **question** będą przypisane do konkretnych algorytmów i struktur danych z tabeli **question_group** oraz typ pytania, opis, pomocnik do pytan (np. możliwe odpowiedzi przy pytaniu (a,b,c,d)), ilość punktów i poprawna odpowiedź. Tabela **quiz_result** będzie zawierać wyniki użytkownika z danego quizu wraz z jego uzyskanym wynikiem.

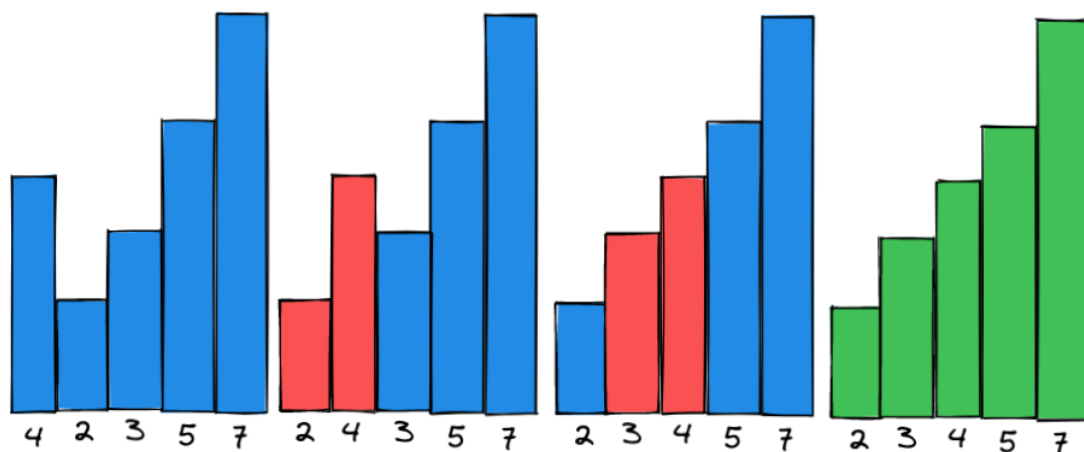
Najczęściej będziemy wykorzystywać zapytania dot. treści samych pytań według odpowiedniej kategorii pytania i do jakiego quizu należy. Poza tym również zapisywanie wyników po zakończonym quizie.

Członkowie projektu

- [\[Jira\]](#) Inżynier (Frontend) (*Maria Sydor*)
- [\[Jira\]](#) Inżynier (Frontend) (*Marcin Stefanowicz*)
- [\[Jira\]](#) Inżynier (Backend) (*Jakub Kamiński*) - lider zespołu
- [\[Jira\]](#) Inżynier (Backend) (*Klaudia Bielak*)
- [\[Jira\]](#) Tester (*Mateusz Gruszkiewicz*)
- [\[Jira\]](#) Devops (*Wiktor Wajszczuk*)

Wizualizacje algorytmów i struktur danych

Algorytmy sortujące



Opis wizualizacji:

1. Na ekranie pojawiają się elementy tablicy w losowym ułożeniu - sugerowana wizualizacja → kolumny o wybranym kolorze.
2. Elementy obecnie przenoszone zostają zaznaczone innym kolorem, następnie zamienione (dobrze dodać w tym miejscu też jakąś animację zamiany miejsc, np. przesuwanie się jednego elementu z miejsca na miejsce), następnie nadajemy im oryginalny kolor
3. Powtarzamy krok 3 do skończenia działania algorytmu

4. Dodatkowo można dać gdzieś czasomierz, który zlicza czas sortowania

Opis interakcji z użytkownikiem:

- Użytkownik ma suwak z wybieraniem jaki czas ma zajmować jedna akcja (najlepiej użyć skali logarytmicznej)
- Użytkownik może zmieniać wartości pojedynczych komórek tablicy (sugerowana implementacja → kolumny to tak naprawdę do czasu uruchomienia suwak wertykalne które można zwiększać/zmniejszać)
- Użytkownik może w odpowiednim polu wpisać/wybrać wielkość tablicy do posortowania
- Użytkownik ma do dyspozycji przyciski losuj, stop/start oraz przewiń które odpowiednio tworzą tablicę losowych elementów (można założyć, że wszystkie docelowo wynoszą zero gdyby użytkownik chciał sam sobie wszystkie dopasować) wznowią/zatrzymają działanie algorytmu albo przewiną na sam koniec, gdzie tablica jest już posortowana

Sortowanie bąbelkowe

Opis:

Metoda sortowania o złożoności czasowej i pamięciowej . Polega na porównywaniu dwóch kolejnych elementów i zamianie ich kolejności, jeżeli zaburza ona porządek, w jakim się sortuje tablicę. Sortowanie kończy się, gdy podczas kolejnego przejścia nie dokonano żadnej zmiany.

Sortowanie przez wstawianie

Opis:

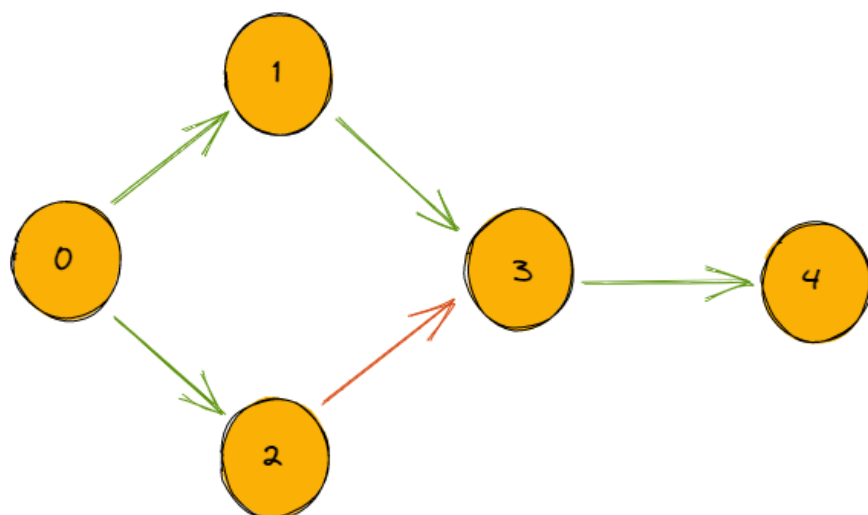
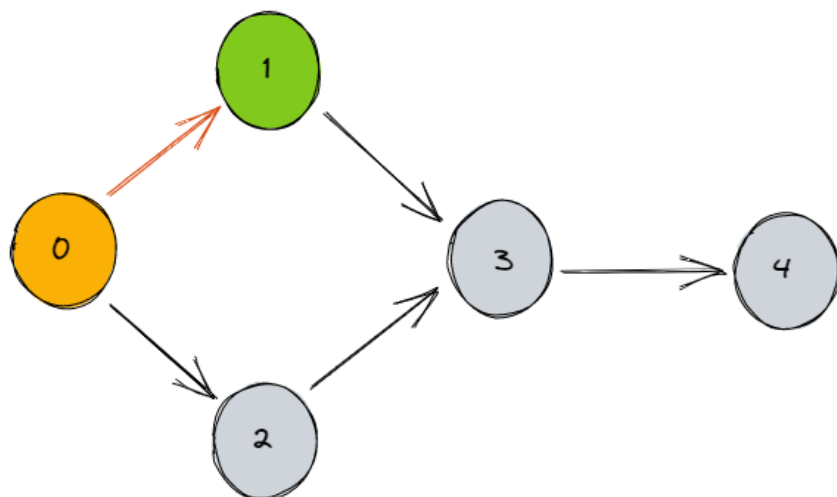
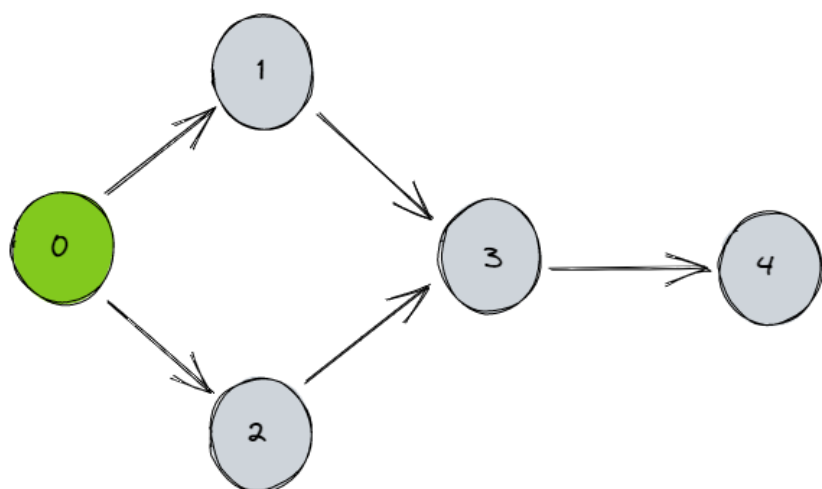
Każdą iterację zaczynamy od wybrania elementu (startując od pierwszego miejsca w tablicy), który będziemy przyrównywać, do elementów znajdujących się na pozycjach poprzedzających. Jeżeli element znajdujący się przed wybranym przez nas elementem jest większy, przesuwamy się o jedno miejsce wstecz.

Sortowanie szybkie

Opis:

Algorytm wykorzystuje technikę "*dziel i zwyciężaj*". Według ustalonego schematu wybierany jest jeden element w sortowanej tablicy, który będziemy nazywać **pivot**. Pivot może być elementem środkowym, pierwszym, ostatnim, losowym lub wybranym według jakiegoś innego schematu dostosowanego do zbioru danych. Następnie ustawiamy elementy nie większe na lewo tej wartości, natomiast nie mniejsze na prawo.

Algorytm grafowe



Opis wizualizacji:

Stworzony lub wybrany przez użytkownika graf przedstawiony jest w postaci wierzchołków połączonych ze sobą.

Wierzchołek, od którego zaczynamy przeszukiwanie grafu oraz każdy kolejny rozpatrywany wierzchołek jest zaznaczony innym kolorem.

Każdy odwiedzony wierzchołek zmienia kolor.

Na koniec zaznaczona jest wybrana, najkrótsza ścieżka.

Opis interakcji z użytkownikiem:

- Rysowanie grafu:

Pojedyncze kliknięcie tworzy wierzchołek (automatycznie taki wierzchołek dostaje numerik zaczynając od 0)

Kliknięcie na istniejący wierzchołek A przytrzymanie i puszczenie na innym wierzchołku B powinno stworzyć krawędź skierowaną ($A \rightarrow B$)

- Użytkownik powinien mieć możliwość zamiast rysować graf samemu wybrać jakiś z listy już istniejących np.:
-
- Użytkownik może wybrać algorytm, który chce wykorzystać do przeszukiwania stworzonego/wybranego grafu.
- Użytkownik może wybrać czy graf jest skierowany czy nie.
- Użytkownik może wybrać wierzchołek, od którego zaczyna przeszukiwanie.

BFS

Opis:

Przeszukiwanie wszerz (ang. *breadth-first search*, *BFS*) – jeden z najprostszych algorytmów przeszukiwania grafu. Przechodzenie grafu rozpoczyna się od zadanego wierzchołka s i polega na odwiedzeniu wszystkich osiągalnych z niego wierzchołków. Wynikiem działania algorytmu jest drzewo przeszukiwania wszerz o korzeniu w s , zawierające wszystkie wierzchołki osiągalne z s . Do każdego z tych wierzchołków prowadzi dokładnie jedna ścieżka z s , która jest jednocześnie najkrótszą ścieżką w grafie wejściowym. Algorytm działa prawidłowo zarówno dla grafów skierowanych jak i nieskierowanych.

DFS

Przeszukiwanie w głąb (ang. *Depth-first search*, w skrócie *DFS*) – algorytm przeszukiwania grafu. Przeszukiwanie w głąb polega na badaniu wszystkich krawędzi wychodzących z podanego wierzchołka. Po zbadaniu wszystkich krawędzi wychodzących z danego wierzchołka algorytm powraca do wierzchołka, z którego dany wierzchołek został odwiedzony.

Dijkstra

Algorytm służy do znajdowania najkrótszej ścieżki z pojedynczego źródła w grafie o nieujemnych wagach krawędzi. (U nas możemy założyć, że każda krawędź ma wagę 1)

Podstawowe struktury danych

Stos

Liniowa struktura danych, w której dane dokładane są na wierzch stosu i z wierzchołka stosu są pobierane (bufor typu **LIFO**, *Last In, First Out*; *ostatni na wejściu, pierwszy na wyjściu*). Ideę stosu danych można zilustrować jako stos położonych jedna na drugiej książek – nowy egzemplarz kładzie się na wierzch stosu i z wierzchu stosu zdejmują się kolejne egzemplarze. Elementy stosu poniżej wierzchołka można wyłącznie obejrzeć, aby je ściągnąć, trzeba najpierw po kolei ściągnąć to, co jest nad nimi.

Obsługiwane operacje to odłóż (push) - dołożenie czegoś do góry stosu oraz zdejmij (pop) - zdjęcie czegoś z góry stosu.



Opis wizualizacji:

Elementy stworzonego stosu ułożone są jeden na drugim.

Dodawany element pojawia się na samej górze, najlepiej w innym kolorze.

Przy zdejmowaniu element na wierzchu zmienia kolor na inny i zostaje usunięty.

Opis interakcji z użytkownikiem:

- Użytkownik ma przycisk Stwórz, który tworzy stos z podanymi przez niego wartościami.
- Użytkownik może dołożyć wybraną przez siebie wartość do stosu za pomocą przycisku Umieść na stosie oraz zdejmować elementy z góry stosu za pomocą przycisku Zdejmij ze stosu.

Kolejka

Opis:

Kolejka – liniowa struktura danych, w której nowe dane dopisywane są na końcu kolejki, a z początku kolejki pobierane są dane do dalszego przetwarzania (bufor typu **FIFO**, *First In, First Out*; *pierwszy na wejściu, pierwszy na wyjściu*).

Powinniśmy mieć dostęp do pierwszego i ostatniego elementu aby sprawnie zakolejkować (queue) i zdekolejkować (dequeue) elementy.



Opis wizualizacji:

Elementy stworzonej kolejki ułożone są jeden za drugim.

Dodawany element wędruje na koniec(z lewej strony) i ma inny kolor niż pozostałe.

Przy zdejmowaniu pierwszy(najbardziej z prawej) element zmienia na chwilę kolor na inny i zostaje usunięty.

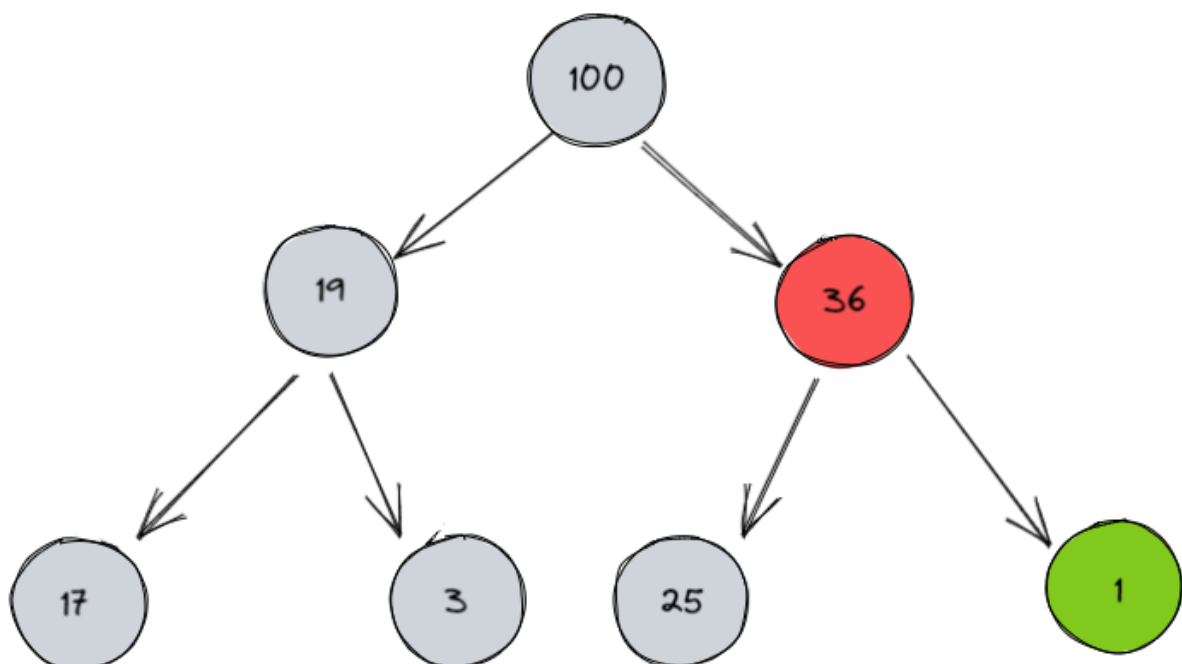
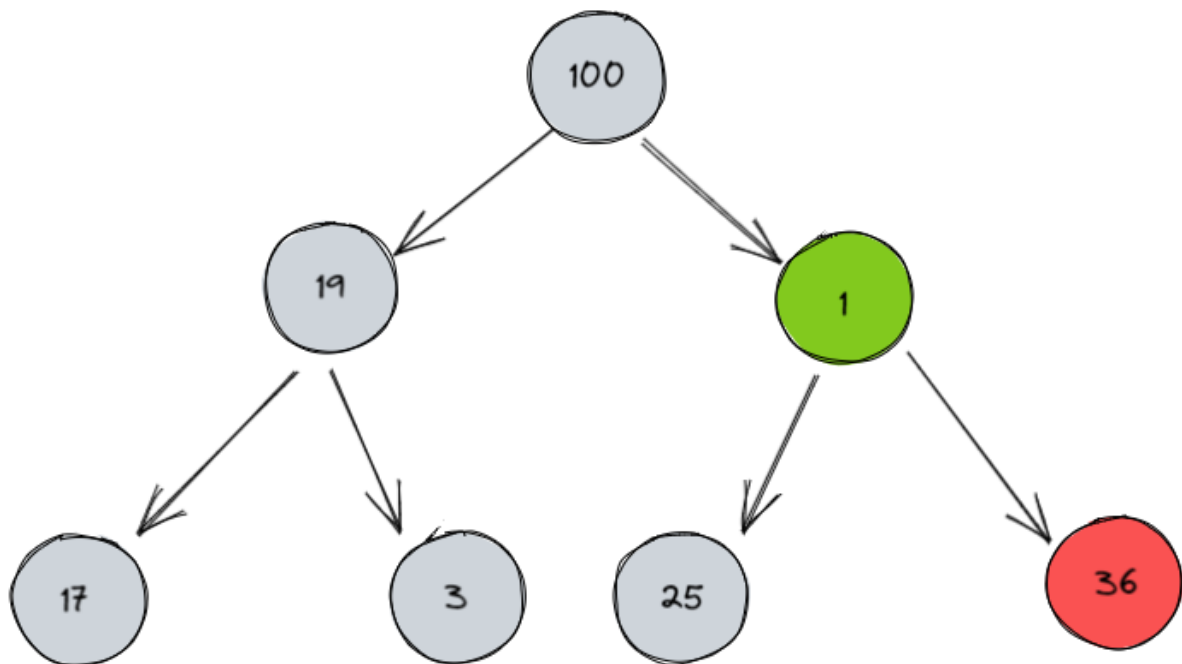
Opis interakcji z użytkownikiem:

- Użytkownik ma przycisk Stwórz, który tworzy kolejkę z podanymi przez niego wartościami.
- Użytkownik może dopisać wybraną przez siebie wartość na koniec kolejki za pomocą przycisku Zakolejkuj oraz zabrać element z początku kolejki za pomocą przycisku Zdekolejkuj.

Kopiec

Opis:

Kopiec (ang. *heap*, tłumaczone też jako *stóg* lub *sterta*) – struktura danych oparta na drzewie, w której wartości potomków węzła są w stałej relacji z wartością rodzica (na przykład wartość rodzica jest nie mniejsza niż wartości jego potomka).



Opis wizualizacji:

Elementy przedstawione są w postaci drzewa (dodatkowo może być też reprezentacja tablicowa).

Przy tworzeniu kopca, dodawaniu oraz zdejmowaniu dane elementy podświetlają się na inny kolor i odpowiednio zamieniają się ze sobą.

Opis interakcji z użytkownikiem:

- Użytkownik może stworzyć kopiec z podanych przez siebie elementów, może też wybrać rodzaj kopca (typ max lub typ min).
- Użytkownik może wstawiać dowolne elementy do kopca za pomocą przycisku Wstaw.
- Użytkownik może zdejmować wybrane elementy, na przykład element najmniejszy lub największy.

Listy

Lista – struktura danych służąca do reprezentacji zbiorów dynamicznych, w której elementy ułożone są w liniowym porządku. Rozróżniane są dwa podstawowe rodzaje list: lista jednokierunkowa w której z każdego elementu możliwe jest przejście do jego następnika oraz lista dwukierunkowa w której z każdego elementu możliwe jest przejście do jego poprzednika i następnika.



Opis wizualizacji:

Każdy element przedstawiony jest w kształcie prostokąta (Node), w którego polu Data znajduje się liczba. Liczby przyporządkowane są rosnąco, zgodnie z kolejnością dodania, mogą być też modyfikowane i edytowane przez użytkownika. Pierwszy dodany element będzie posiadał etykietę head. Zależnie od typu listy, prostokąty będą miały widocznie wydzieloną część wskazującą na następny (next) i/lub poprzedni (prev) element. Element listy nieprzechowywujący żadnej wartości będzie charakteryzował X w prostokącie. W momencie wykonywania na poszczególnym elemencie listy konkretnej akcji, dany element podświetlają się na inny kolor.

Opis interakcji z użytkownikiem:

- Użytkownik może wybrać rodzaj listy (jedno lub dwukierunkowa).
- Użytkownik ma możliwość stworzenia listy samemu bądź wygenerowania jej.
- Użytkownik ma możliwość usuwania i dodawania własnych elementów za pomocą przycisków Wstaw/Usuń.
- W momencie wykonywania operacji, wybrany aktualnie element podświetla się na inny kolor.
- Elementy tworzonej listy układane są jeden obok drugiego.
- Aktualnie dodawany element wędruje na koniec i jest wyróżniony innym kolorem.

Quiz

Użytkownik będzie miał możliwość przechodzenia do kolejnego modułu po pozytywnym zaliczeniu quizu (próg procentowy od ok. 85%). Suma punktów będzie widoczna na górze strony oraz na liście modułów, gdzie będą one odpowiednio oznaczone (w trakcie nauki/zaliczone).

Pula pytań do każdej kategorii quizu będzie podzielona na poziomy trudności. Każde kolejne pytanie w quizie będzie trudniejsze. Wraz ze wzrostem trudności zwiększa się liczba punktów do zdobycia za poprawne odpowiedzi. Po każdym pytaniu użytkownik dostaje informację czy udzielona odpowiedź jest poprawna (jeśli nie, wyświetlana jest prawidłowa odpowiedź) i przechodzi do następnego. Na koniec widoczny jest wynik ogólny podany w procentach, liczba zebranych punktów oraz możliwy jest przegląd prawidłowych odpowiedzi.

Typy pytań:

1. [1pkt] Pytanie testowe, w którym należy wybrać jedną prawidłową spośród podanych odpowiedzi.

Przykład:

Jaką złożoność ma algorytm sortowania bąbelkowego w najgorszym przypadku?

- a) $O(n \log n)$
- b) $O(\log n)$
- c) $O(n)$
- d) $O(n^2)$

2.[2pkt] Pytanie teoretyczne, w którym należy wpisać własną odpowiedź.

Przykład:

Dana jest tablica liczb $t = \{1, 2, 3, 4\}$. Ile zamian zostanie wykonanych aby posortować tablicę rosnąco metodą sortowania bąbelkowego?

Miejsce na odpowiedź:

3.[3pkt] Analiza algorytmu, podany jest fragment kodu, należy napisać jaką wartość będzie miała podana zmienna.

Schemat:

Jaką wartość będzie miała zmienna x [wylosowana spośród możliwych] po wykonaniu poniższych operacji?

Miejsce na kod

Miejsce na odpowiedź:

Przykład:

Dana jest tablica liczb $t = \{1, 2, 3, 5, 1\}$, $n = 5$. Jaką wartość będzie miała zmienna **temp** po wykonaniu poniższych operacji?

```
for (j = 0; j < n-1; j++)
```

```
for (i = 0; i < n-j-1; i++)
```

```
if (t[i] > t[i+1]){
```

```
int temp = t[i];
```

```
t[i] = t[i+1];
```

```
t[i+1] = temp;
```

```
}
```

Miejsce na odpowiedź:

4.[4pkt] Pytanie testowe, może być kilka poprawnych odpowiedzi.

Schemat:

Jaką wartość będzie miała zmienna [wylosowana spośród możliwych] w kroku [wylosowany numer kroku z podanego zakresu] algorytmu?

t - podana tablica liczb

n - rozmiar tablicy

x - zmienna

y - zmienna

a) $t[6*x]$

b) $t[n-1+y]$

c) $n-x$

d) $t[n-x]$

Przykład:

Jaką wartość będzie miała zmienna **t[x]** w kroku **3.** sortowania bąbelkowego?

t = {0, 1, 5, 12, 3, 5, 1, 13}

$n = 8$

$x = 1$

a) $t[n - 2 * x]$

b) $n - x$

c) $n - 7 * x$

d) $t[n + 2 * t[0]]$

5.[5pkt] Zadanie polegające na uzupełnieniu wybranego fragmentu kodu.

Przykład:

Uzupełnij brakujący fragment sortowania bąbelkowego.

```
for (j = 0; j < n-1; j++)
```

```
for (i = 0; i < n-j-1; i++)
```

```
if (A[i] > A[i+1]){
```

```
... }
```