Dokumentacja

1. Zainstalować node.js -> <https://nodejs.org/en/>
2. Zainstalować paczki używając komendy ‘npm install’ w folderach backend i frontend
3. Backend używa domyślnie MariaDb jako bazy danych, więc trzeba pobrać MariaDb -> <https://mariadb.org/download/?t=mariadb&p=mariadb&r=11.3.0&os=windows&cpu=x86_64&pkg=msi&m=vpsFree>, automatycznie pobierze się również HeidiSQL, w którym musimy utworzyć połączenie z backendem, a następnie stworzyć instancję bazy danych nazwaną Shopping oraz dodać do niej dwie tabele używając następujących komend:  
   CREATE TABLE `Product` (

`Id` int(11) NOT NULL AUTO\_INCREMENT,

`Name` varchar(30) CHARACTER SET utf8 DEFAULT NULL,

PRIMARY KEY (`Id`)

) ENGINE=InnoDB AUTO\_INCREMENT=8 DEFAULT CHARSET=utf8mb4;

Oraz

CREATE TABLE `ShoppingList` (

`Id` int(11) NOT NULL AUTO\_INCREMENT,

`ProductId` int(11) DEFAULT NULL,

PRIMARY KEY (`Id`),

KEY `list\_product` (`ProductId`),

CONSTRAINT `list\_product` FOREIGN KEY (`ProductId`) REFERENCES `Product` (`Id`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8mb4;

1. Backend uruchamiamy poprzez wywołanie komendy: node index.js w folderze backend
2. Frontend uruchamiamy poprzez wywołanie komendy: npm start w folderze frontend

Do zrobienia tego projektu używany był javascript, nodejs i react

Pracowaliśmy w programie Visual Studio Code oraz HeidiSQL

Wymagania funkcjonalne:

-Dodaje wpisaną wartość do tabeli shoppinglist  
-Zczytuje dodane wartości z tabeli shoppinglist  
-Usuwa wybrane wartości z tabeli shoppinglist

Wymagania niefunkcjonalne:

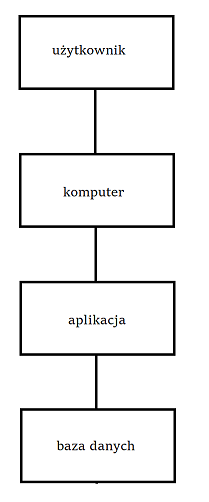
-Aplikacja jest wydajna, w realnym czasie aktualizuje tabele  
-Aplikacja jest czytelna i intuicyjna, bardzo łatwa w obsłudze  
-Aplikacja jest/może być powszechnie stosowana

1. Diagramy UML

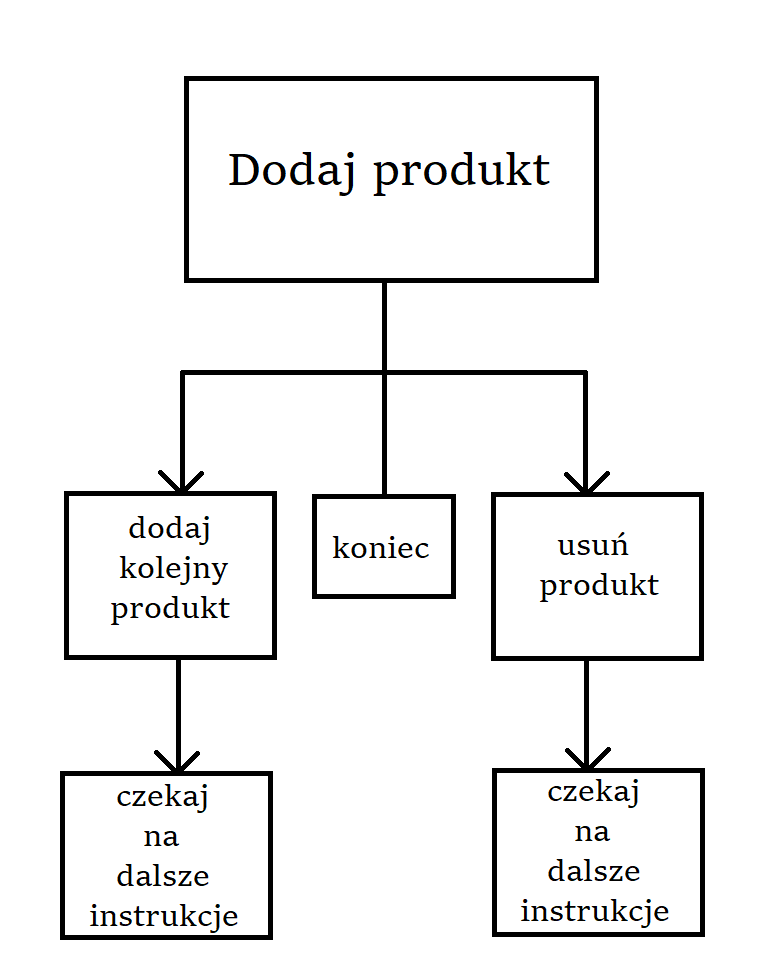
- Diagram przypadków użycia



- Diagram wdrożenia



- Diagram czynności



1. Diagramy ERD

