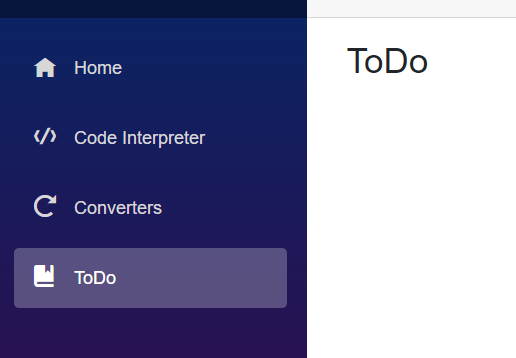
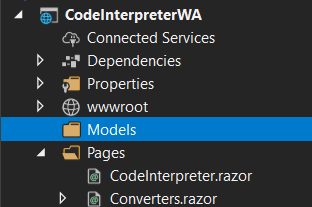
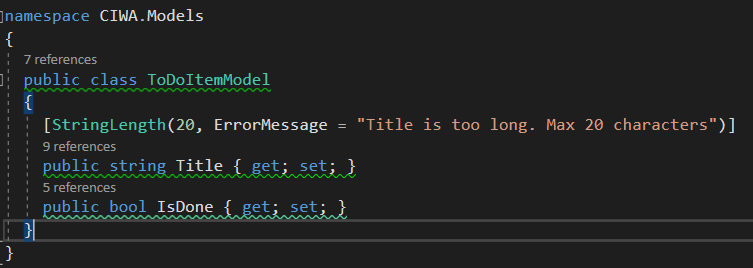
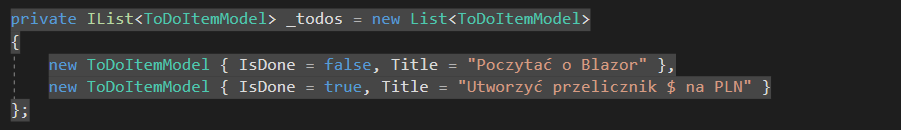
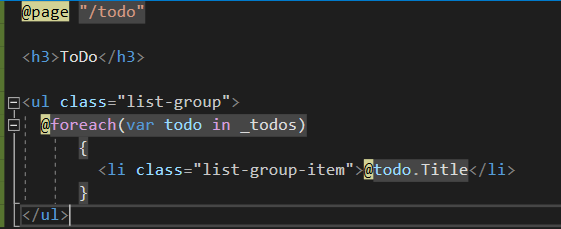
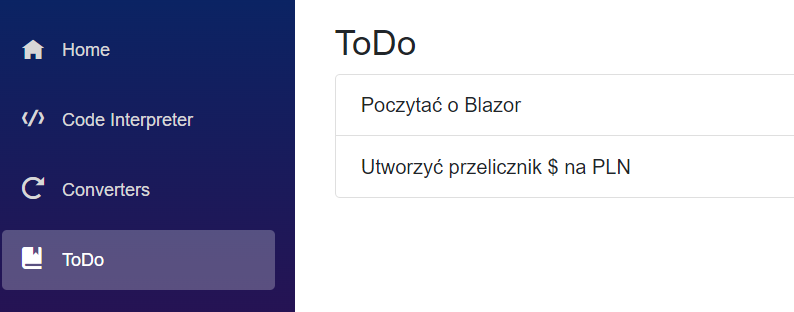
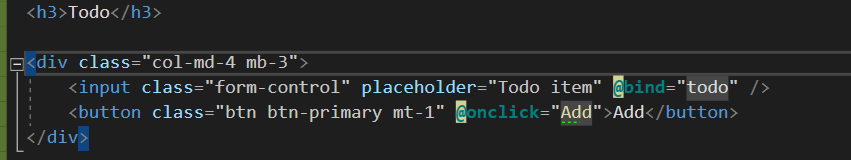
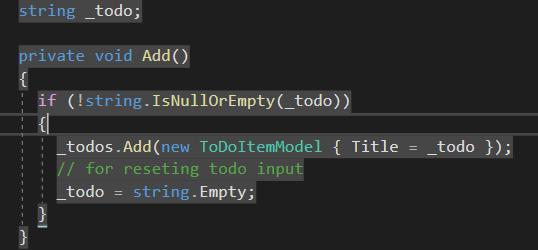
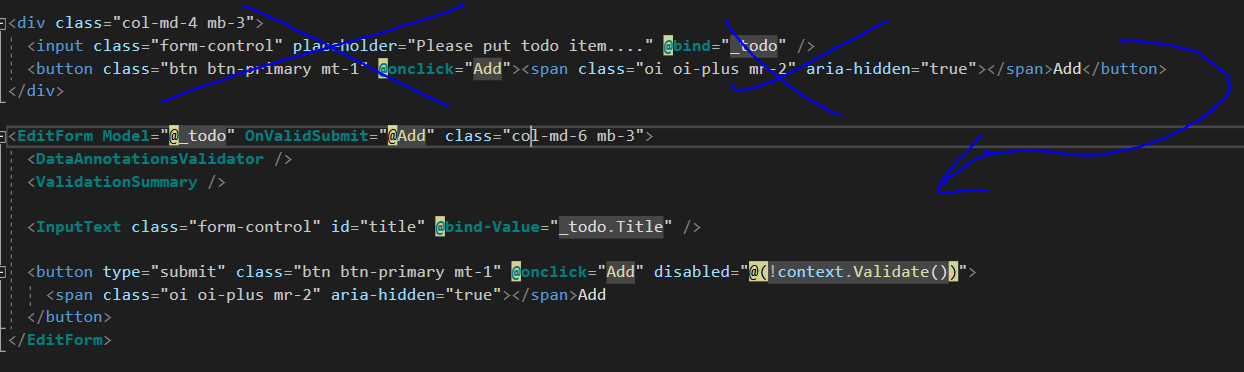
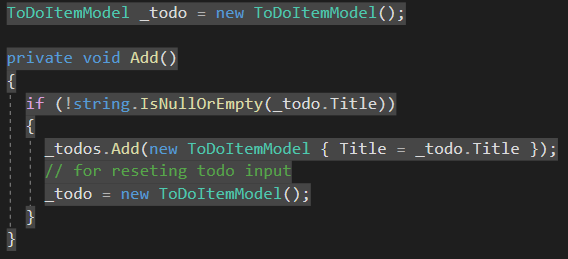
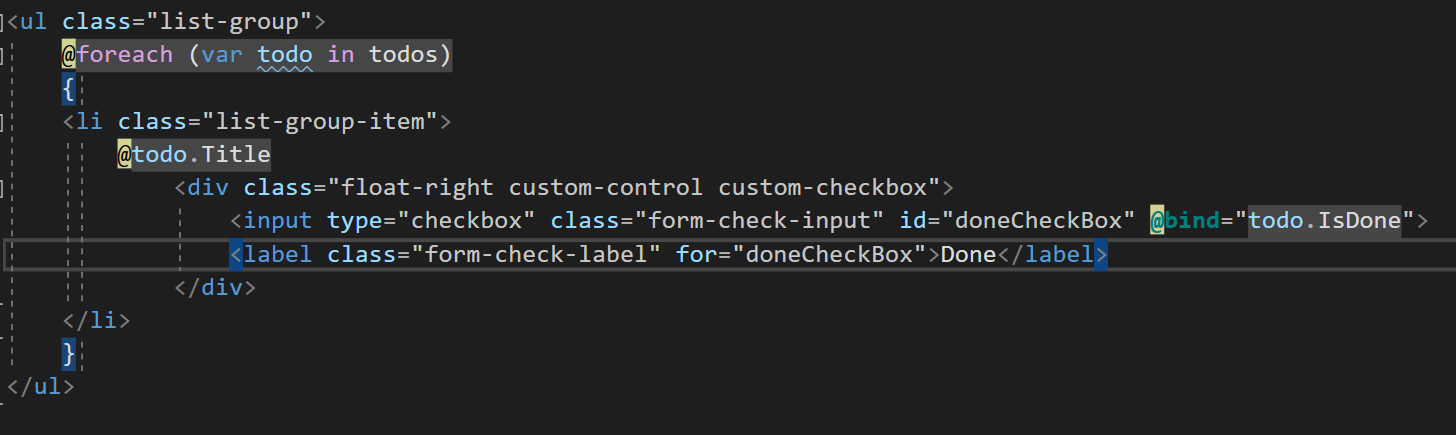
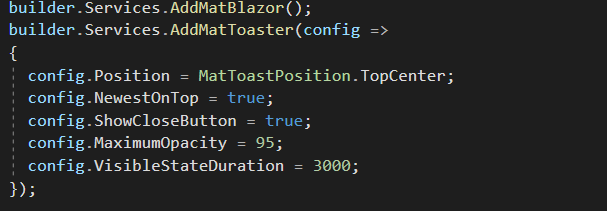
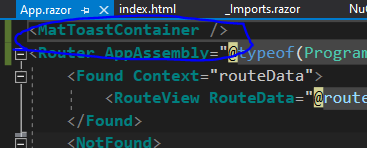
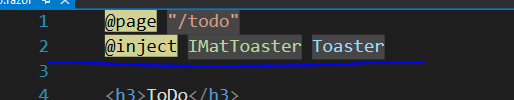
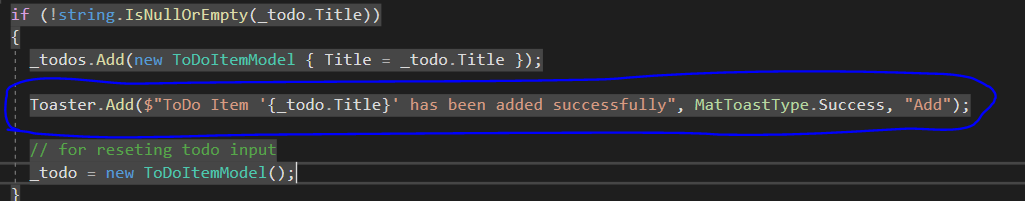
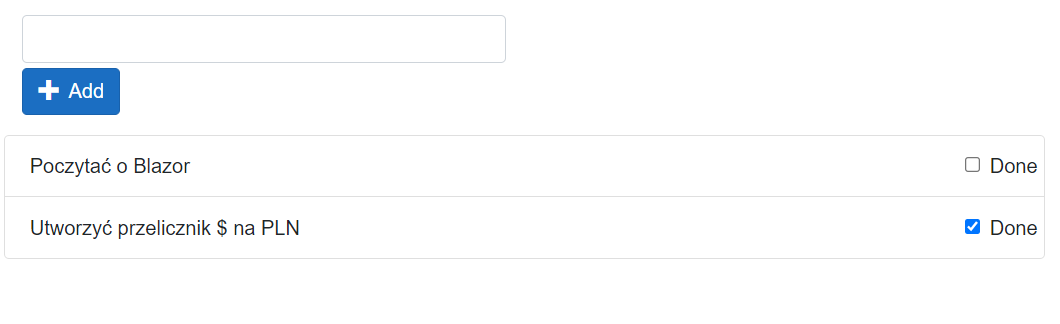
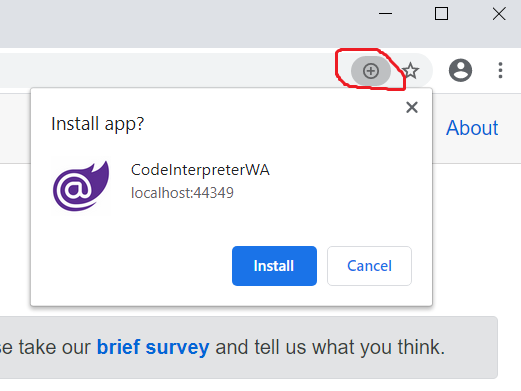
1. Proszę dodać nowy komponent o nazwie **ToDo.** Komponent ten będzie pozwalał na tworzenie oraz zarządzanie listą zadań do zrobienia.
2. Następnie proszę zdefiniować routing dla nowego komponentu. **@page "/todo".**
3. Podobnie jak w poprzednich krokach proszę dodać nowy komponent *Todo* do panelu nawigacji. Korzystając z dostępnych darmowych ikonek na stronie <https://useiconic.com/open> proszę wybrać jedną i przypisać ją do nowo utworzonej nawigacji.
4. Proszę skompilować i uruchomić aplikację. W moim przypadku wygląda to tak:  
   
5. Kolejnym krokiem jest dodanie folderu o nazwie ***Models***, w tym miejscu jak sama nazwa wskazuje będziemy przechowywali różnego rodzaju modele w naszej aplikacji. W kroku **35** (poprzednie zadanie) utworzyliśmy model dla przeliczników, ale dla uproszczenia umieściliśmy go bezpośrednio w komponencie. Nie jest to dobra praktyka dlatego od teraz nasze modele będziemy przetrzymywali w folderze *Models.*  
   
6. Do folderu *Models* proszę dodać nową klasę o nazwie **ToDoItemModel** która będzie reprezentować pojedynczy element z naszej listy zadań do zrobienia. Dodatkowo dodajmy atrybuty które pozwolą nam na walidację danych wprowadzonych przez użytkownika.   
   
7. Następnym krokiem jest wykorzystanie pliku **\_Imports.razor.** Plik ten pozwala na implementację dyrektyw zapisanych w tym pliku do wszystkich komponentów. Proszę dodać dyrektywę *@using* wskazującą nowy folder z modelami. Od tego momentu każdy z komponentów powinien mieć dostęp do wszystkich modeli z tego folderu.  
   
8. Teraz proszę powrócić do komponentu ToDo i w sekcji *@code* proszę dodaj listę typu *TodoItemModel* o nazwie \_**todos**. Do listy proszę dodać dwa domyśle zadania  
   
9. Kolejnym krokiem jest wyświetlenie zadań do zrobienia. W tym celu proszę dodać następujący kod do komponentu *Todo*.
10. Proszę sprawdzić czy aplikacja działa poprawnie.  
    
11. Musimy umożliwić użytkownikowi dodawanie nowych zadań. W tym celu proszę dodać przycisk oraz input. Akcja kliknięcia przycisku powinna uruchomić metodę **Add** którą dodasz w następnym kroku. Aby pobrać wartość wpisaną przez użytkownika należy powiązać input ze zmienną o nazwie \_**todo.** Zmienną utworzysz w następnym kroku.  
    
12. W tym kroku proszę utworzyć prywatną zmienną typu string o nazwie \_**todo,** oraz metodę o nazwie **Add**, której zadaniem jest dodanie nowego zadania do listy oraz wyczyszczenie zawartości zmiennej **todo**.  
      
    
13. W tym kroku dodamy obsługę walidacji wprowadzonych danych przez użytkownika. W tym celu musimy nieco zmodyfikować kod dodając formularz (EditForm).  
      
    
14. Teraz aby umożliwić użytkownikowi zaznaczenie które zadania zostały już zrobione proszę zmienić kod wyświetlający zadania w następujący sposób:  
    
15. Na koniec [dodajmy darmowe komponenty od Material Design (MatBlazor)](https://www.matblazor.com/), wykorzystamy komponent Toast który będzie się wyświetlał po każdym poprawnym dodaniu elementu do listy ToDo. Najpierw proszę zainstalować MatBlazor przy pomocy nuget. Następnie w pliku \_Imports.razor dodać @using MatBlazor, pozwoli nam to na wykorzystanie zainstalowanych komponentów w naszych komponentach. Teraz musimy dodac pliki statyczne do index.html  
      
    Następnym krokiem jest rejestracja serwisu w Program.cs  
      
    Teraz musimy zdefiniować gdzie ma się wyświetlać wiadomość zrobimy to globalnie w pliku App.razor.  
    
16. Aby wyświetlić wiadomość musimy ją utworzyć. Od tego momentu każde poprawne dodanie nowego zadania do listy ToDo powinno wyświetlić informacje na ekranie. Proszę pamiętać o wstrzyknięciu serwisu odpowiedzilnego na Toaster messages.  
      
    
17. Proszę skompilować i uruchomić aplikację oraz sprawdzić czy działa poprawnie
18. Ponieważ podczas tworzenia projektu zaznaczyliśmy opcję PWA teraz możemy zainstalować naszą aplikację i korzystać z niej jak z natywnej aplikacji systemu. Proszę zainstalować naszą aplikację korzystając z przycisku jak na obrazku poniżej  
    
19. Proszę uruchomić aplikację i sprawdzić czy działa poprawnie.  
    