Spis treści

[1. Wstęp 1](#_Toc56560519)

[1.1 Wprowadzenie 1](#_Toc56560520)

[1.2 Cele pracy 2](#_Toc56560521)

[1.3 Przegląd rynkowych rozwiązań 2](#_Toc56560522)

[2. Opis teoretyczny 2](#_Toc56560523)

[2.1 Azure 2](#_Toc56560524)

[2.1.1 Azure IoT Explorer 2](#_Toc56560525)

[2.1.2 Azure Portal 2](#_Toc56560526)

[2.2 Technologie Webowe 2](#_Toc56560527)

[2.2.1 DOTNET 2](#_Toc56560528)

[2.2.2 JavaScript 2](#_Toc56560529)

[2.2.3 HTML 2](#_Toc56560530)

[2.3 Bazy danych 2](#_Toc56560531)

[2.3.1 Entity Framework 2](#_Toc56560532)

[2.3.2 Microsoft SQL Server Management Studio 2](#_Toc56560533)

[2.3.3 Azure Data Studio 2](#_Toc56560534)

[2.4 C# 2](#_Toc56560535)

[2.4.1 ASP.NET MVC 3](#_Toc56560536)

[2.4.2 LINQ 3](#_Toc56560537)

[2.4.3 3](#_Toc56560538)

[3. Wymagania systemu 3](#_Toc56560539)

[3.1 Opis ogólny 3](#_Toc56560540)

[3.2 Wymagania funkcjonalne 3](#_Toc56560541)

[3.2.1 Rejestrowanie nowych urządzeń 3](#_Toc56560542)

[3.2.2 Przewidywanie przyszłych zdarzeń 4](#_Toc56560543)

[3.2.3 Sterowanie urządzeniami za pośrednictwem aplikacji. 4](#_Toc56560544)

[3.2.4 Funkcjonalność aplikacji 4](#_Toc56560545)

[3.3 Wymagania niefunkcjonalne 4](#_Toc56560546)

[4. 5](#_Toc56560547)

# 1. Wstęp

## 1.1 Wprowadzenie

## 1.2 Cele pracy

## 1.3 Przegląd rynkowych rozwiązań

(przykłądy, o co chodzi z iot bibliografia pokazać co wim w jakich systemach)

# 2. Opis teoretyczny

## 2.1 Azure

### 2.1.1 Azure IoT Explorer

### 2.1.2 Azure Portal

## 2.2 Technologie Webowe

### 2.2.1 DOTNET

### 2.2.2 JavaScript

### 2.2.3 HTML

## 2.3 Bazy danych

### 2.3.1 Entity Framework

### 2.3.2 Microsoft SQL Server Management Studio

### 2.3.3 Azure Data Studio

## 2.4 C#

### 2.4.1 ASP.NET MVC

### 2.4.2 LINQ

### 2.4.3

# 3. Wymagania systemu

## 3.1 Opis ogólny

Program powinien być stworzony w języku C# przy wykorzystaniu frameworka ASP.NET MVC. Aplikacja ma służyć do zarządzania ogromną ilością urządzeń za pośrednictwem aplikacji webowej. System ma wykorzystywać usługi dostępne za platformy chmurowej Microsoft Azure, takich jak Azure IoT Hub, Azure SQL Database, Storage account, Logic App.   
Aplikacja ma być gotowa do zarządzania dużą ilością ekspresów do kawy wyposażonych czujniki umożliwiające zdalne łączenie, ale również powinna być na tyle skalowalna, aby można ją rozszerzyć do administrowania innymi rodzajami urządzeń. System ma być również wyposażony w symulator umożliwiający łączenie się z portalem Azure i wysyłać na niego dane telemetryczne z symulowanego urządzenia.

Aplikacja została stworzona przy użyciu języka C#, w środowisku Visual Studio 2019. Aplikacja służy do zarządzania przez użytkownika jego urządzeniami zarejestrowanymi w usłudze Azure IoT hub. Na potrzeby pracy stworzyłem aplikację umożliwiającą zarządzanie ekspresami do kawy(ale aplikacja w łatwy sposób zmienić ją na usługę administrującą inne urządzenia). Aplikacja jest wyposażona w symulator, który symuluje działanie ekspresu, czyli na podstawie wcześniej zdefiniowanych wartości przez użytkownika aplikacja odbiera dane wysłane przez symulator na Azure portal.

## 3.2 Wymagania funkcjonalne

### 3.2.1 Zarządzanie aplikacją przez administratora

Administrator ma mieć dostęp do danych użytkownika szczególnie do danych kontaktowych, wyłączając hasła, hasła mają być szyfrowane utrudniając dostęp do nieautoryzowanych logowań. Administrator może modyfikować dane związane z użytkownikiem, takich jak ilość możliwych urządzeń które użytkownik może zarejestrować w usłudze, dodawanie poszczególnych urządzeń w imieniu użytkownika oraz usuwanie ich. Pracownik z uprawnieniami administratora ma dostęp do danych statystycznych dotyczących zużycia zasobów w usłudze Azure.

### 3.2.2 Zarządzanie aplikacją przez użytkownika

Aplikacja ma umożliwiać użytkownikowi swobodne zarządzanie jego urządzeniami i otrzymywanie danych informujących o stanie urządzenia oraz prognozowanych koniecznych czynności w przyszłości.

### 3.2.3 Rejestrowanie nowych urządzeń

Użytkownik może rejestrować nowe urządzenia nowe urządzenia oraz je usuwać. Do zarządzania swoimi urządzeniami za pomocą wspomnianej usługi wystarczy poprawnie odbyty proces rejestracji lub zalogowania w przypadku posiadania już wcześniej konta, oraz podanie odpowiednich informacji, o swoim urządzeniu

-nazwa modelu

-maksymalna możliwa ilość wody w zbiorniku

-maksymalna waga ziaren w pojemniku

-W przypadku, gdy ekspres nie posiada czujników odpowiadających za sprawdzanie stanu zapełnienia pojemników na

<obrazek przedstawiających dodawanie nowego urządzenia oraz konfigurację>

### 3.2.4 Informowanie o statusie urządzeń, oraz o przewidywanych zdarzeniach

Użytkownik ma mieć dostęp obecnego statusu urządzenia, takich jak stan włączony/wyłączony, czy aktualne działania przeprowadzane przez urządzenia. W przypadku ekspresów do kawy użytkownik ma mieć wgląd do stanu działania przy robieniu nowej kawy, takich jak stan wykonania zadania. Powinna występować możliwość odczytania przyszłych zdarzeń, takich jak ilość cykli pracy urządzenia przed kolejnym opróżnieniem zbiorników na odpady lub uzupełnienia pojemników na wodę lub ziarno.

### 3.2.5 Sterowanie urządzeniami za pośrednictwem aplikacji.

Aplikacja umożliwia sterowanie działaniami urządzeń zarejestrowanych w usłudze. Z poziomu aplikacji możliwe jest włączenie/ wyłączenie urządzenia, i wykonanie dostępnej przez urządzenie akcji, takiej jak rozpoczęcie procesu robienia kawy lub płukanie.

### 3.2.4 Funkcjonalność aplikacji

Aplikacja niezalogowanych użytkowników wita oknem umożliwiającym zalogowanie się lub rejestrację.

<screen z oknami logowania i rejestracji>

Po zalogowaniu użytkownik może zmienić dane dotyczące swojego konta takie jak nazwa, hasło, lub numer telefonu. Hasła użytkowników są szyfrowane, co oznacza że nikt nie ma dostępu do oryginalnego hasła, co znacząco utrudnia logowanie się na konto użytkownika przez osoby nieodpowiednie.

## 3.3 Wymagania niefunkcjonalne

System jest w stanie sprostać zarządzaniu milionów urządzeń IoT. System powinien przesyłać dane telemetryczne w formacie JSON.

# 4.