Aplikacje Internetowe

*Aplikacja do wystawiania faktur on-line*

Robert Międlarz

EF/AA-DU L02

127181

# Nazwa projektu

VatApp - Aplikacja do wystawiania faktur online

# Narzędzia technologiczne

* **JavaScript -** Jest to skryptowy język programowania. Głównym autorem jest Brendan Eich. Pod koniec lat 90 organizacja ECMA wydała standaryzację tego języka pod nazwą ECMAScript.
* **AngularJs** - To framework przeznaczony do tworzenia działających w przeglądarce internetowej aplikacji charakteryzujących się dużą interaktywnością. Oferuje strukturę doskonale sprawdzającą się w programach wysoce modułowych. Głównymi możliwościami frameworka: dwukierunkowe wiązanie danych, routing, operacje na modelu DOM, wstrzykiwanie zależności, metody AJAX.
* **Node.js -** Jest to środowisko uruchomieniowe JavaScript przeznaczone do działania poza przeglądarką internetową. Najczęściej wykorzystywany jest do przygotowania serwerów WWW.
* **Express** - to pakiet instalowany za pomocą menadżera npm i przeznaczony do tworzenia serwerów WWW. Jest używany w środowisku Node.js. Znacząco ułatwia tworzenie i korzystanie z serwera przy budowie aplikacji sieciowych.
* **MongoDB -** Jest to oparta na dokumentach baza danych. Jest prostą i łatwą do nauczenia no-sql'ową bazą danych. MongoDB przypomina prace z obiektami w języku programowania. Zamiast przechowywać dane w serii rekordów i tabel, jak ma to miejsce w SQL, MongoDB zawiera kolekcje i dokumenty. Podstawową różnicą jest to, że dokumenty mogą przechowywać znacznie wiecej (rożnych) informacji.

# Gruby Klient

Jest to termin używany gdy programista przenosi odpowiedzialność za wykonywanie najcięzszych (obliczeniowo) operacji - zadań na komputer klienta. W takim wypadku serwer współdziałający z aplikacją na za zadanie jedynie dostarczać dane które klient będzie przetwarzał.

Sposób działania nowoczesnej architektury sieciowej:

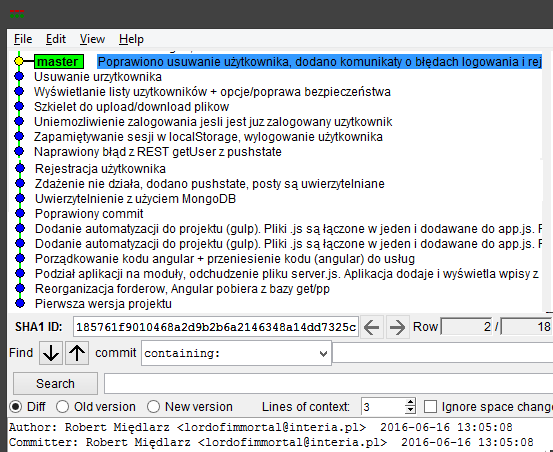
1. Przegladarka internetowa użytkownika wykonuje żądanie statyczne strony HTML.
2. Strona HTML zawiera kod JavaScript odpowiedzialny za utworzenie podstawowej struktury strony.
3. JavaScript wykonuje żądania Ajax w celu pobrania zawartości - zwykle w formacie JSON - za pomocą API.
4. Następnie kod JavaScript pobiera tę zawartośc i modyfikuje model DOM w celu jej wyświetlenia.

Zalety nowoczesnej architektury sieciowej:

* Prostsza infrastruktura.
* Wydajność
* Ustandaryzowane narzędzia
* Szybkie prototypowanie
* Duża modularność

# Opis aplikacji

Poniżej przedstawiono (rys 1) historię modyfikowania aplikacji w repozytorium git.

  
*Rys 1. Branch history*

## Diagram bazy danych MongoDB

VatAppERD

## Spis metod REST

|  |  |  |
| --- | --- | --- |
| Typ | Adres | Opis |
|  | | |
| post | /session | Metoda używana do sprawdzenia czy użytkownik jest zalogowany |
| get | (api) posts/ | Pobieranie wszystkich postów z bazy |
| post | (api) posts/ | Zapisywanie posta do bazy |
| post | (api) sessions/ | Metoda używana do zalogowania się |
| get | (api) users/ |  |
| get | (api) users/all | Wyświetlanie nazw użytkowników |
| post | (api) users/ | Dodawanie nowego użytkownika |
| post | (api) users/delete | Usuwanie użytkownika |
|  |  |  |

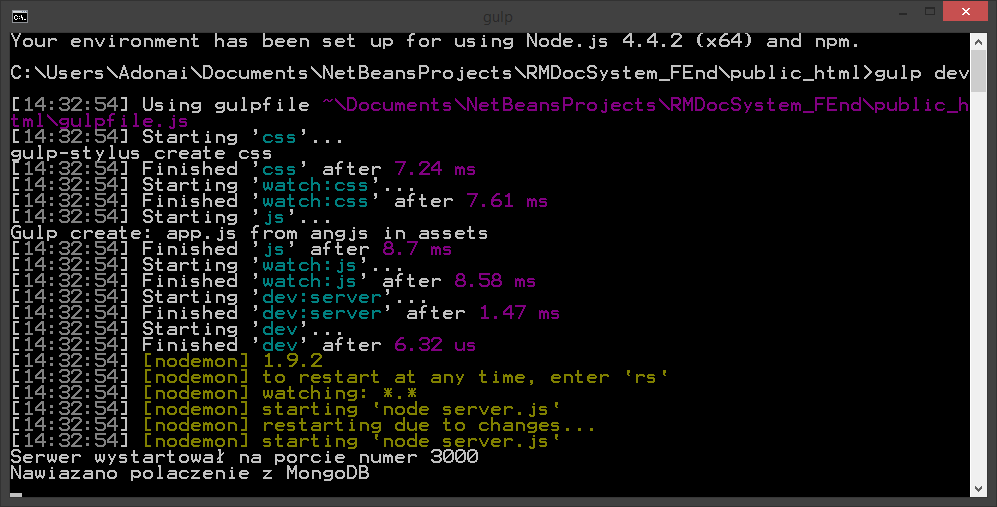
## Automatyzacja za pomocą Gulp

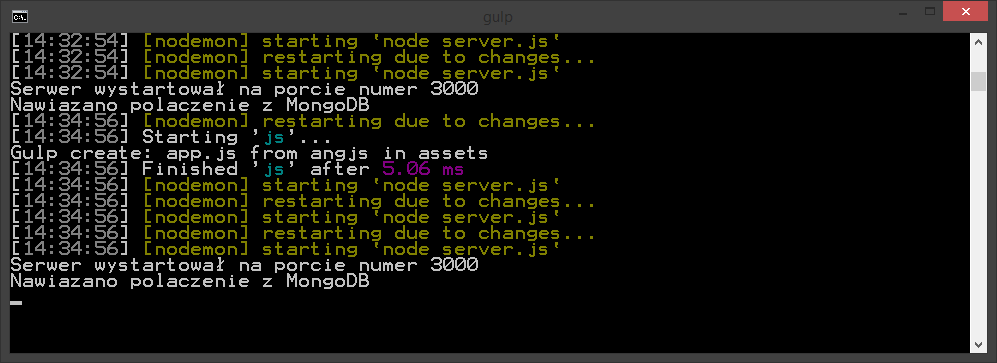
Wszystkie nowoczesne aplikacje JavaScript obejmują pewne kroki procesu kompilacji. W tworzonej aplikacji został wykorzystane potokowe przetwarzanie skryptów js oraz kompilacja arkuszy stylów css za pmocą narzędzia *Stylus*.

Proces przetwarzania obejmował:

* Łączenie wszystkich modułów (plików) w jeden
* Poprawa definiowania zależności angular (potrzebne do minimalizacji)
* Minimalizacja skryptów
* Dołączenie mapy źródeł

Utworzono również zadanie "gulp dev" umożliwiające uruchomienie serwera (rys 2) aplikacji z trigerami restartującymi (rys 3) serwer gdy zostanie wykryta zmiana zawartości ważnych plików.

  
*Rys. 2 Uruchomienie aplikacji za pomocą jednego polecenia*

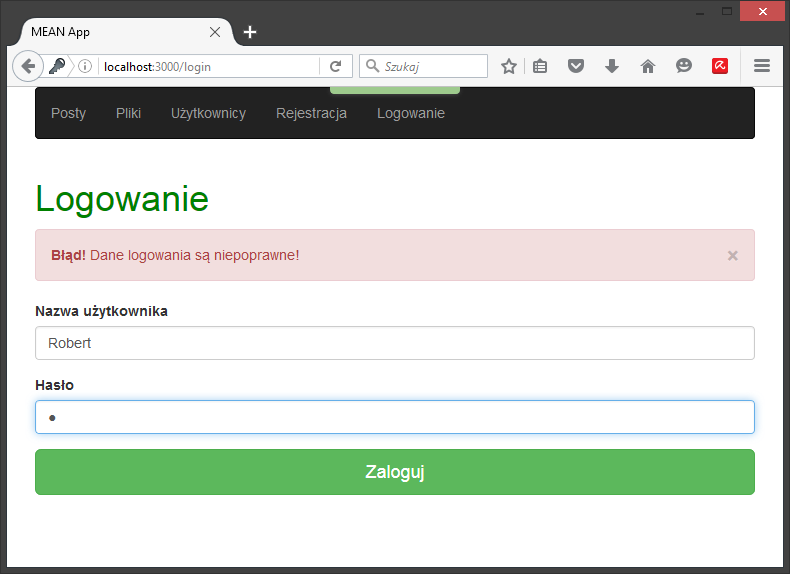
  
*Rys 3 Automatyczny restart serwera po wykryciu zmian w plikach.*

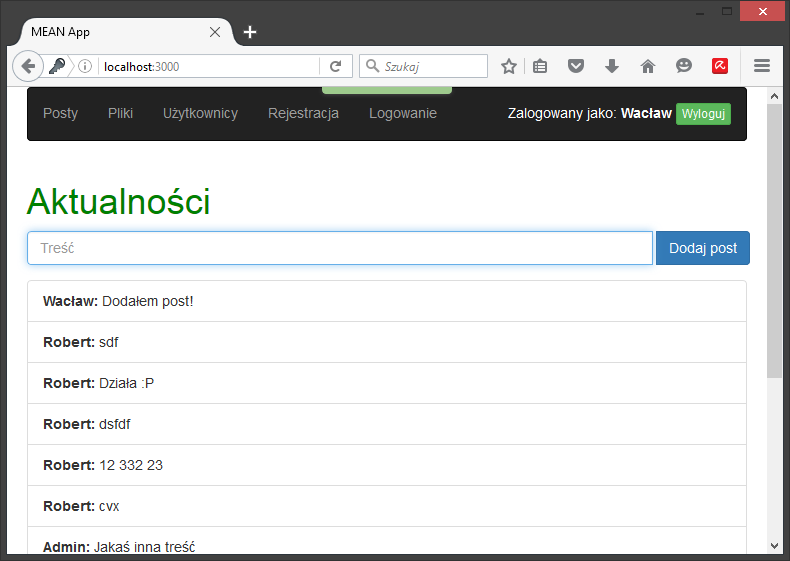
## Wykaz funkcji

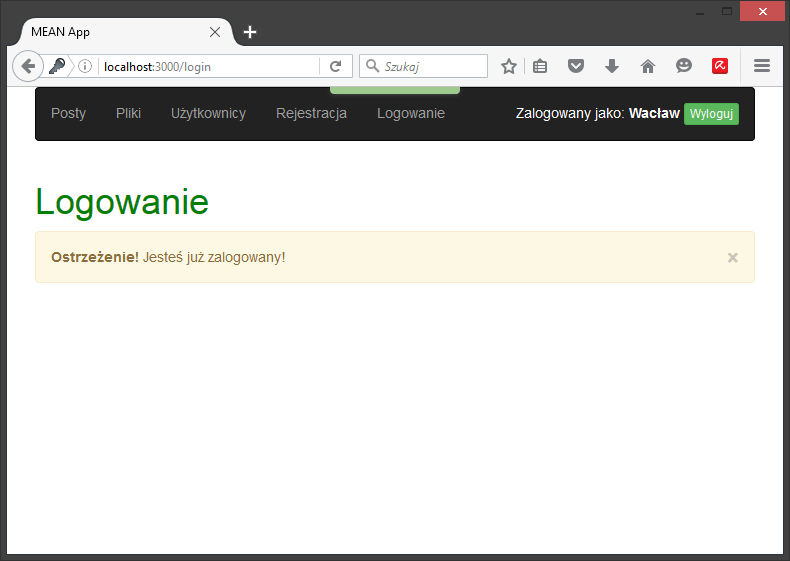
Aplikacja dzięki zastosowaniu responsywności wyświetla się jednakowo dobrze na urządzeniach mobilnych i desktopowych.

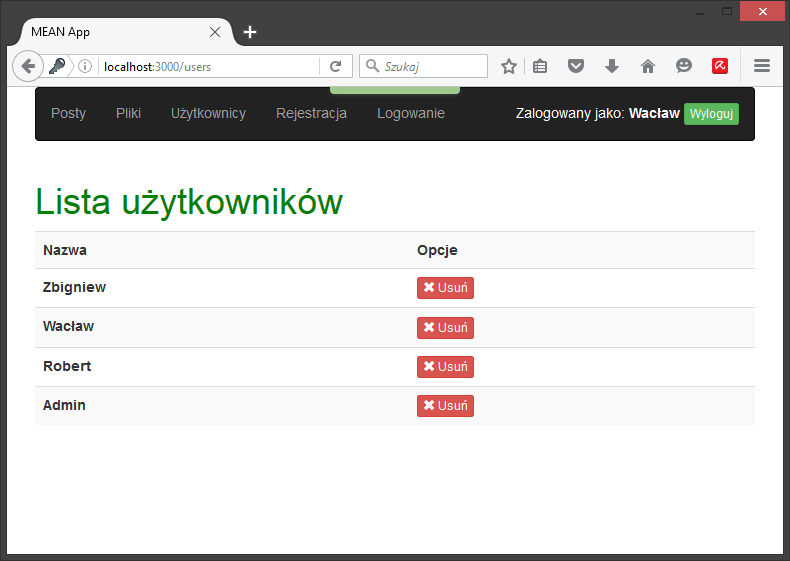
1. **Posty** - dostęp do tej funkcjonalności ma każdy użytkownik, również niezalogowany. Pozwala ona na szybką komunikację z użytkownikami systemu. Aby dodać post należy sie zalogować na swoje konto lub je utworzy.
2. **Rejestracja** - funkcjonalność ta pozwala stworzyć nowego użytkownika. Aby to zrobić należy podać nazwę oraz dwa razy wpisać to samo hasło. W przypadku błędnego wpisania hasła w polu "powtórz hasło" zostanie wyświetlony komunikat. Jeżeli dane wprowadzono poprawnie nastąpi automatyczne zalogowanie.
3. **Logowanie** - Aby mieć dostęp do wszystkich funkcji systemu należy się zalogować. Opcja ta jest dostępna tylko gdy nie jesteśmy zalogowani. W przeciwnym wypadku wyświetli nam się komunikat uniemożliwiający ponowne zalogowanie bez uprzedniego wylogowania
4. **Zapamiętanie sesji** - zalogowany użytkownik jest uwierzytelniany tokenem *jwt.* Informacja o tym czy jesteśmy zalogowani jest przechowywana w *local.storage*. Do zaimplementowania tej funkcjonalności użyto HTML5 *pushstate*.
5. **Wylogowanie** - opcja ta pozwala na wylogowanie użytkownika, usuniecie tokenu z pamięci przeglądarki.
6. **Wyświetlanie listy użytkowników** - umożliwia podgląd obecnie utworzonych kont użytkowników.
7. **Opcja Usuń użytkownika** - opcja ta umożliwia usunięcie użytkownika z systemu. Na tym etapie każdy użytkownik może usunąć każdego.

## Widok aplikacji

*Rys 4 Widok komunikatu wpisanych niepoprawnych danych.*

*Rys 5 Widok aplikacji po zalogowaniu.*

*Rys 6. Widok próby zalogowania podczas istniejącej sesji.*

*Rys 7. Widok listy użytkowników.*