AttendMe

Projekt zaliczeniowy

Aplikacji do sprawdzania obecności na zajęciach.

Przedmiot: Frameworki front-endowe

Semestr: 2025 / 2026

Opracował: Paweł Kołodziej

WPROWADZENIE

Aby zaliczyć zajęcia z przedmiotu “Frameworki front-endowe" należy wykonać i opublikować projekt aplikacji SPA do sprawdzania obecności.

Projekt należy zrealizować za pomocą frameworku Vue.js w wersji 3 (obowiązująca), implementację należy wykonać z użyciem Composition API w języku typescript.

W projekcie należy wykazać umiejętności w zakresie:

- Tworzenia i konfiguracji projektu (adekwatna struktura folderów i dołączenie niezbędnych bibliotek)
- Podziału aplikacji na komponenty (SFC) z prawidłowym wykorzystaniem komunikacji między komponentami.
- Prawidłowego wykorzystania składni szablonów vue.js.
- Konfiguracji routingu wraz z wykorzystaniem layout/u
- Prawidłowe połączenie z backendem wraz z obsługą błędów
- Przechowywania stanu (np. Kontekstu użytkownika zalogowanego) poza drzewem komponentów (pnia).
- Nadania właściwego wyglądu aplikacji poprzez użycie biblioteki zewnętrznej (np. Bootstrap, Tailwind, Bulma itp...) lub własnego arkusza styli.
- Użyteczności: organizacja nawigacji, informacja zwrotna dla użytkownika (np. Przy ładowaniu danych).
- Dobrych praktyk w zakresie kodowania: Modularyzacji kodu, SRP, DRY, prawidłowego typowania (typescript) i czytelnego formatowania.

Zadaniem aplikacji jest ułatwienie sprawdzania obecności na zajęciach. Sama czynność sprawdzenia obecności będzie polegała na wyświetleniu kodu QR na telefonie i zbliżeniu go do kamery urządzenia odczytującego.

W trakcie przygotowania aplikacji należy wykorzystać backend dostępny pod adresem:

https://attendme-backend.runasp.net/

Wersja demonstracyjna aplikacji: https://attendme.runasp.net/#/login

Dane do logowania:

Wykładowca:

```
usr: pk
pass: 123#Asd
```

Student:
login tworzymy wg wzorca: stu+numer indeksu
hasło to numer indeksu

przykładowo dla numeru indeksu 12345:

```
usr: stu12345
pass: 12345
```

Na potrzeby realizacji projektu aplikacja nosi nazwę AttendMe (attend-me).

Wykonanie projektu i zaliczenie:

1. Projekt można zrealizować dzieląc **prace w zespole** (maksymalnie 3 osobowym – osoby z tej samej grupy) **lub zrealizować całość zakresu samodzielnie**.
2. W przypadku realizacji projektu w zespole **należy określić, za którą cześć prac odpowiedzialna była dana osoba** zapisując tą informację w odrębnym pliku `dev-team.txt` w folderze głównym projektu. W pliku należy zapisać nazwisko i imię studenta, numer indeksu oraz zakres prac (opisowo lub jako lista plików).
3. Aby uzyskać zaliczenie, niezależnie od zakresu prac realizowanych w projekcie zespołowym, **wymagana jest znajomość działania całej aplikacji**. W ramach oceniania pracy prowadzący może poprosić o omówienie wybranych fragmentów rozwiązania.
4. Użycie AI w projekcie:
    a. Niezalecane jest używanie narzędzi SI (AI) do generowania kodu, z wyjątkiem oczywistych, typowych fragmentów (np. drobne sugestie, dane testowe/mockup itp.).
    b. W przypadku użycia AI do wygenerowania większego fragmentu kodu (kilka linii) należy oznaczyć ten fragment wraz podaniem promptu w komentarzu oraz użytego modelu.
    c. Nieoznaczenie kodu generowanego może uniemożliwić uzyskanie zaliczenia.
    d. W przypadku użycia modelu generatywnego do tworzenia kodu należy oczekiwać dodatkowego sprawdzenia wiedzy w zakresie przedmiotu.
5. Terminy (do północy):
    a. Termin “zerowy”: **11 stycznia 2026**
    b. I Termin: **22 luty 2026**
    c. II Termin: **27 luty 2026**
6. Przekazanie projektu
    a. Każda osoba zaliczająca projekt powinna przekazać własną wersję projektu wysyłając plik projektu w ramach indywidualnego czasu Teams z prowadzącym.

        UWAGA:
        - Plik należy nazwać numerem indeksu np. 12356_Imię_Nazwisko.zip
        - Przed spakowaniem ZIP proszę **usunąć folder node_modules**.
        - Sprawdzanie będą wyłącznie pliki archiwum .zip

    b. Po rygorem nieważności: o przekazaniu projektu **należy powiadomić prowadzącego** wysyłając wiadomość na czacie indywidualnym w aplikacji MS Teams. Prowadzący powinien potwierdzić otrzymanie pliku.

7. Minimalny zakres projektu:
    a. Wykładowca:
        i. Ekran logowania
        ii. Pulpit wykładowcy (lista zajęć z filtrami)
        iii. Ekran szczegółów zajęć z listą obecności
        ~~iv. Funkcja do odczytu kodu qr dla danych zajęć (skanowanie kodu obecności)~~
        niewymagane, testowanie skanowania można wykonać w aplikacji demonstacyjnej.
    b. Student:
        i. Ekran logowanie (ten sam co dla wykładowcy)
        ii. Ekran autoryzacji/rejestracji urządzenia
        iii. Pulpit studenta (lista zajęć z filtrami)
        iv. Ekran generowania kodu obecności (QR)
        v. Ekran szczegółów zajęć (sygnatura zajęć + informacja o obecności)
8. ~~Opcjonalny zakres projektu: Ekrany administracyjne umożliwiające edycję danych źródłowych (lista wykładowców, studentów, lista przedmiotów, lista zajęć i przedmioty zajęć wraz z formularzami edycyjnymi).~~

Funkcjonalność aplikacji:

ROLE UŻYTKOWNIKÓW

W aplikacji przewidziane są 3 role użytkowników: **Wykładowca, Student** ~~oraz **Administrator**~~.

“Wykładowca” loguje się do aplikacji z użyciem loginu i hasła, ma do dyspozycji pulpit, na którym widoczna jest lista prowadzonych przez niego zajęć. Może otworzyć szczegóły zajęć i pobrać link otwierający ekran listy obecności wraz z funkcją skanowania.

“Student” loguje się do aplikacji z użyciem loginu i hasła oraz może skanować obecność po wcześniejszym aktywowaniu tej funkcji na danym urządzeniu (z użyciem linku rejestrującego urządzenie dostarczonego przez “**Wykładowcę**”). Na stronie głównej aplikacji ma do dyspozycji pulpit, na którym widoczna jest lista zajęć, których jest uczestnikiem. Z pulpitu może aktywować funkcję “rejestruj obecność”. Może sprawdzić frekwencję w ramach zajęć z danego przedmiotu oraz wyświetlić obecność na poszczególnych zajęciach.

~~(poza zakresem minimalnym projektu)~~

~~**Administrator** posiada uprawnienia do zarządzania użytkownikami aplikacji. Może edytować przedmioty, grupy, zajęcia i terminy.~~

AUTENTYKACJA

W aplikacji stosujemy autentykację tokenową JWT. Dla **wykładowcy i studenta** token autentykacyjny uzyskujemy po zalogowaniu z użyciem danych (login + hasło). Dla **studenta** token autentykacyjny

urządzenia uzyskujemy po zarejestrowaniu urządzenia z użyciem linków dostarczonego przez wykładowcę (link można wygenerować w aplikacji demonstracyjnej - nie trzeba implementować tej funkcji w projekcie zaliczeniowym).

BACKEND

Do komunikacji z backendem wykorzystujemy bibliotekę kliencką wygenerowaną na podstawie specyfikacji OpenAPI. Preferowanym narzędziem do wygenerowania biblioteki jest NSwagStudio.

Backend aplikacji dostępny jest pod adresem*:

https://attendme-backend.runasp.net/

(*ze względu na potencjalne ograniczenia hostingowe możliwa jest zmiana lokalizacji backendu)

Dokumentacja endpointów dostępna jest w postaci interfejsu użytkownika pod adresem:

https://attendme-backend.runasp.net/swagger/index.html

Specyfikację OpenAPI można pobrać ze strony:

https://attendme-backend.runasp.net/swagger/v1/swagger.json

WAŻNE: Do zaliczenia projektu nie jest wymagane samodzielnie wygenerowanie biblioteki klienckiej - można użyć kodu dostępnego w poniższym przykładzie.

Przykład jak dołączyć wygenerowaną bibliotekę kliencką do projektu jest dostępny w poniższym repozytorium:

https://github.com/true-vue/attend-me-client-demo/tree/main/src

Przykład obejmuje również autentykację użytkownika.

SKANOWANIE KODÓW

Do sprawdzania obecności wykorzystywany jest odczyt kodów QR. Do generowania kodów (Student) i ich odczytu (Wykładowca) należy wykorzystać komponenty bibliotek zewnętrznych kompatybilnych z vue.js v3.

Wskazówki dotyczące developmentu:

Testowanie skanowania

Przykładowe biblioteki:

Generowanie kodów: https://qr-vue.tie.pub/

Odczyt: https://gruhn.github.io/vue-qrcode-reader/

EKRANY APLIKACJI - WSPÓLNE

LOGOWANIE

Jeżeli użytkownik nie dokonał wcześniej autentykacji (pobrany token) aplikacja przekierowuje go do ekranu logowania. Na ekranie logowania do formularza wpisuje login oraz hasło i naciska przycisk “Zaloguj”.

Korzystając w wygenerowanej biblioteki (OpenApi) wysyłamy żądanie autentykacyjne do serwera. Jeżeli żądanie się **powiedzie** przekierowujemy użytkownika do pulpitu. Jeżeli logowanie **nie powiedzie** się wyświetlamy użytkownikowi stosowny komunikat.

Wykorzystywane metody backendu:

Logowanie użytkownika do aplikacji (student, nauczyciel, admin) `userLogin(loginName: string, password: string): Promise<TokenResult>` Przy prawidłowej odpowiedzi (udało się zalogować) biblioteka zadba o zapisanie tokenu w przeglądarce oraz jego odtworzenie przy restarcie aplikacji (odświeżeniu strony – sessionStorage) - ważne jest to rozwiązanie zwiększające wygodę developmentu (odświeżenie strony nie powoduje utraty tokenu). W systemach produkcyjnych utrwalanie tokenu nie jest zalecane.
Pobiera informację o zalogowanym użytkowniku (userId = undefined): `AttendMeBackendClientBase.userGet(userId: number | undefined): Promise<User>` Na podstawie rezultatu możemy sprawdzić jakie role posiada zalogowany użytkownik - czyli rozróżnić funkcjonalność wykładowcy i studenta.

EKRANY APLIKACJI - WYKŁADOWCA

PULPIT WYKŁADOWCY

Po zalogowaniu wykładowca kierowany jest do pulpitu, na którym wyświetlana jest lista prowadzonych przez niego zajęć. Dane do wyświetlenia listy pobieramy z backendu.

W ramach listy powinna być dostępna opcja filtrowania: *aktualne (dziś), jutro, następny tydzień, minione, wszystkie: wg dat, wg tekstu

Element listy powinien zawierać podstawowe informacje do danych zajęciach:
Nazwa przedmiotu, Grupa, Termin (data, godzina), Sala/Zdalnie.

Po kliknięciu elementu listy wykładowca przechodzi do Szczegółów Zajęć.

Wykorzystywane metody backendu:

Pobiera listę zajęć dla zalogowanego prowadzącego:

```
courseTeacherSessionsGet(body: CourseSessionListFiltersPagedListParams
| undefined): Promise<CourseSessionListItemPagedList>
```

Aby wykonać metodę parametr body powinien zawierać co najmniej:

```
{
  pageNumber: 1,
  pageSize: 999999,
}
```

W implementacji należy również uwzględnić przekazanie filtrów z ui

SZCZEGÓŁY ZAJĘĆ WYKŁADOWCY

Po wejściu na ekran szczegółów zajęć wykładowca widzi następujące informacje:

Sygnatura zajęć: Nazwa przedmiotu, Grupa, Termin (data, godzina).

Lista obecności: Imię Nazwisko, nr indeksu, status obecność (obecny / nieobecny).

Lista powinna być wyposażona w przycisk “Odśwież”, być odświeżana cyklicznie (w tle) ~~lub od odpowiedzi na zdarzenie serwera (wersja zaawansowana)~~.

W ramach ekranu powinien być dostępna możliwość otwarcia lub skopiowania linku do “Ekran skanowania”.

Wykorzystywane metody backendu:

Zwraca zajęcia o wskazanym identyfikatorze (szczegóły)

```
courseTeacherSessionGet(sessionId: number | undefined):
Promise<CourseSessionListItem>
```

Zwraca listę studentów wraz z informacją o obecności:

```
courseSessionAttendanceListGet(sessionId: number | undefined):
Promise<CourseSessionAttendanceRecord[]>
```

EKRAN SKANOWANIA

Na ekranie skanowania należy wyświetlić zachętę do zbliżenia telefonu do kamery i aktywować skaner kodów QR. Za każdym razem, gdy skaner odczyta kod powinien go wysłać do backendu. Backend w odpowiedzi dostarczy informacje czy obecność została zarejestrowana i dla którego studenta. Informację zwrotną należy wyświetlić na ekranie w postaci komunikatu widocznego przez kilka sekund.

Umożliwia pobranie tokenu sesji skanera, aby można było go otworzyć również na innym urządzeniu (np. Na tablecie do którego dostęp mają studenci, podczas gdy wykładowca zalogowany jest na desktopie).

Aplikacja kliencka powinna uzyskać wynegerowany w tej metodzie token (np. Odczytać z wygenerowanego adresu url) a następnie przypisać go do Backend.deviceTokenResult zanim zostanie wykonana metoda `courseSessionAttendanceRegister`

```
AttendMeBackendClientBase.courseSessionAttendanceScannerTokenGet(courseSessionId: number | undefined): Promise<TokenResult>
```

Rejestruje obecność użytkownika bazując na tokenie sesji skanera(token należy przekazać przez adres url i przypisać do Backend.deviceTokenResult) . attenderToken to wartość tokenu wygenerowanego po stronie uczestnika (studenta) i zeskanowanego przez skaner.

```
courseSessionAttendanceRegister(attenderToken: string | undefined): Promise<User>
```

EKRANY APLIKACJI - STUDENT

REJESTROWANIE URZĄDZENIA STUDENTA

Aby student mógł rejestrować obecność na zajęciach musi uwierzytelnić urządzenie (np. telefon komórkowy) na którym będzie wyświetlany kod qr do zeskanowania przez prowadzącego.

Uwierzytelnienie urządzenia jest odrębną operacją od logowania (bazuje na odrębnym tokenie jwt)

W tym celu: otwiera link przesłany przez wykładowcę (link z tokenem można wygenerować w aplikacji demonstracyjnej po otwarciu dialogu “rejestracja urządzenia” ze strony danych zajęć).

Link powinien prowadzić w do ekranu “Rejestracja urządzenia” na którym student podaje dane zgodne z typem `DeviceRegisterDTO`. Ścieżka linku będzie zawierać token tymczasowy, który należy wykorzystać do zarejestrowania urządzenia (`userDeviceRegisterWithToken`). W odpowiedzi backend dostarczy token “trwały”, który zostanie zapisany lokalnie i będzie wykorzystywany w kolejnych żądaniach (do uzyskania kodu obecności).

UWAGA: W przypadku samodzielnej implementacji komunikacji z backendem należy odtworzyć logikę wykorzystania tokenów zgodnie z implementacją z przykładu: https://github.com/true-vue/attend-me-client-demo/tree/main/src

Wykorzystywane metody backendu:

Rejestracja urządzenia, z którego można wyświetlać kody qr służące sprawdzeniu obecności:

```
userDeviceRegisterWithToken(token: string, data: DeviceRegisterDTO): Promise<TokenResult>
```

Po wykonaniu zapisuje w localStorage token i przywraca go przy restartcie aplikacji (odświeżeniu strony)

Token (pierwszy argument `userDeviceRegisterWithToken`) można uzyskać po zalogowaniu na konto nauczyciela w aplikacji demonstracyjnej i wykonaniu metody:

```
AttendMeBackendClientBase.userDeviceRegisterTokenGet(deviceUserId:
number | undefined): Promise<TokenResult>
```

W panelu wykładowcy aplikacji demonstracyjnej link można uzyskać otwierając dialog "rejestracja urządzenia" dostępny na szczegółach danych zajęć.

```
Backend.deviceAuthReset()
```

Umożliwia wyzerowanie tokenu urządzenia zapisanego na danym telefonie (czyści localStorage). Nie resetuje autentykacji urządzenia po stronie serwera (ta akcja zastrzeżona jest dla użytkownika z rolą teacher).

W panelu wykładowcy aplikacji demonstracyjnej zresetowanie urządzenia można wykonać po otwarciu dialogu "rejestracja urządzenia" dostępnego na szczegółach danych zajęć.

PULPIT STUDENTA

Ekran pulpitu dostępny jest po zalogowaniu na konto studenta. Po wejściu pulpit użytkownik widzi listę zajęć, w których uczestniczy oraz akcję/przycisk "Rejestruj obecność".

Dane do wyświetlenia listy pobieramy z backendu korzystając z właściwego end-point'u.

W ramach listy powinna być dostępna opcja filtrowania: wg zakresu czasu (analogicznie jak dla nauczyciela), wg tekstu (przedmiot, lokalizacja).

Element listy powinien zawierać podstawowe informacje o danych zajęciach:
Nazwa przedmiotu, Termin (data, godzina), Status obecności (obecny / nieobecny), Lokalizacja

Po kliknięciu elementu listy student przechodzi do Szczegółów Zajęć.

Kliknięcie "Rejestruj obecność" powoduje przejście do ekranu "Rejestrowanie obecności"

Zwraca listę zajęć dla zalogowanego studenta:

```
courseStudentSessionsGet(body: CourseSessionListFiltersPagedListParams
| undefined): Promise<CourseSessionListItemPagedList>
```

EKRAN SZCZEGÓŁÓW ZAJĘĆ STUDENTA

Po wejściu na ekran szczegółów zajęć student widzi następujące informacje:

Sygnatura zajęć: Nazwa przedmiotu, Grupa, Termin (data, godzina).

- informacja o obecności - aktywne zajęcia

- frekwencja całkowita + obecność na wcześniejszych zajęciach

- % zaawansowania kursu

Ekran powinien być wyposażony w przycisk “Odśwież” bądź być odświeżany cyklicznie (w tle) lub od odpowiedzi na zdarzenie serwera (wersja zaawansowana).

W ramach ekranu powinna być dostępna akcja: “Rejestruj obecność”

Zwraca listę ze szczegółami zajęć dla danego studenta i danej grupy, do której należy. Aby uzyskać informację o konkretnych zajęciach należy wybrać element od właściwym sessionId. `courseStudentGroupSessionsGet(courseGroupId: number | undefined): Promise<CourseSessionListItem[]>`
Zwraca informacje o obecności studenta na zajęciach z danego przedmiotu: `courseStudentAttendanceGet(courseGroupId: number | undefined): Promise<AttendanceLog[]>`

EKRAN REJESTROWANIE OBECNOŚCI

Na ekranie rejestrowania obecności należy wyświetlić kod QR, wraz zachętą do zbliżenia telefonu do kamery urządzenia, na którym załadowany jest ekran skanowania obecności (wykładowca).

Kod QR należy generować cyklicznie - co 2s należy pobrać nowy “ticket” z backendu.

Wraz z ticketem dostarczana jest informacja o ostatnio zarejestrowanej obecności, którą należy wykorzystać do wyświetlenia komunikat, gdy obecność zostanie zarejestrowana.

Zwraca token (ticket), który należy wyświetlić jako kod QR na urządzeniu celem zeskanowania. `userAttendanceTicketGet(): Promise<TokenResult>`