**2. Analiza tematu**

W rozdziale przedstawiono główne założenia podczas projektowania i implementacji aplikacji. Opisane zostały również już istniejące rozwiązania, oraz ich wady w odniesieniu do otrzymanych wymagań.

**2.1 Założenia**

Głównym celem przy projektowaniu aplikacji było rozwiązanie problemu braku informacji o zajęciach odbywających się w danej sali laboratoryjnej. Aplikacja musiała stworzyć plan uwzględniając wszystkich nauczycieli mających zajęcia w danej sali. Wygenerować przejrzysty i zrozumiały dla użytkownika interfejs przedstawiający te dane. Musiała również dynamicznie zmieniać wyświetlane informację w zależności od otrzymanych danych. Każdy z nauczycieli musiał również mieć możliwość zmieniania sposobu w jaki dane są wyświetlane przy każdej sali oraz pisania wiadomości wyświetlanych z boku planu zajęć.

Plan zajęć miał zostać wyświetlony na tabletach znajdujących się przy drzwiach do sali. Tablet został umieszczony w szklanej szafce co pozwala tylko na kontakt wzrokowy. Ponadto musiał zostać stworzony edytor w formie strony internetowej służący do edytowania zajęć i pokoi , oraz baza danych przetrzymujące wszystkie informacje.

Drugim z założeń miała być stabilność aplikacji ze względu na brak możliwości kontaktu użytkownika z tabletem wyświetlającym plan zajęć. Wszystkie błędy, chwilowe braki kontaktu z serwerem oraz inne nieprzewidziane sytuacje musiały zostać obsłużone, oraz nie mieć wpływu na wyświetlany plan zajęć. Po rozwiązaniu problemu aplikacja musiała wrócić do normalnego trybu pracy.

Kolejną ważną kwestią było bezpieczeństwo danych. Serwer aplikacji musiał przetrzymywać znaczne ilości informacji, których wprowadzenie zabiera duży okres czasu. Utracenie takich danych musiałoby wiązać się z ich ponownym wprowadzeniem co byłoby dużym problemem dla użytkowników. Ponadto aplikacja musiała również zapewnić bezpieczeństwo podczas wprowadzania danych. Dane mogły pochodzić tylko nauczycieli posiadających konto w systemie. Dane przesyłane przez użytkowników nie posiadających odpowiednich praw miały zostać zignorowane. Otrzymane informacje musiały być również sprawdzane pod kątem ich poprawności, aby wykluczy zaistnienie potencjalnych błędów spójności danych.

Ważnymi kwestiami był również wydajność, skalowalność oraz przenośność.

**2.2 Dostępne rozwiązania**

Politechnika Śląska posiada już system tworzący już plan zajęć. Jest on dostępny pod adresem url <https://plan.polsl.pl/>. Jednak nie jest on idealnym rozwiązaniem i nie spełnia wszystkich założeń. Głównym problemem jest fakt iż strona wymaga odświeżenia po każdej aktualizacji danych. Co przy braku kontaktu z tabletem sprawia że edycja planu jest bardzo utrudniona. Nie ma również możliwości pisania wiadomości dla użytkowników. Interfejs użytkownika nie był również projektowany z myślą o wyświetlanie go na ekranie tabletu.

**3. Wybór narzędzi**

Przy wyborze narzędzi użytych w projekcie kierowano się przede wszystkim ich dostępnością i ich powszechnością. W rozdziale zostały one pokrótce opisane.

**3.1 Środowisko programistyczne**

Projekt został stworzony w środowisku InteliJ IDEA. Jest to środowisko firmy JetBrains. Można uzyskać darmową studencką licencję. Środowisko zawiera kompilator kodu Java. Obsługuje projekt zbudowane za pomocą narzędzia maven. Wspiera większość technologii używanych w projekcie. Posiada również rozbudowane funkcje debugowania kodu z możliwość dynamicznej edycji wszystkich składowych projektu.

**3.2 Kontrola wersji**

Narzędziem do kontroli wersji został GIT. Jest to rozproszony system kontroli wersji co oznacza że każdy użytkownik posiada na swoim komputerze kopię wszystkich poprzednich wersji i może na nich pracować nawet bez połączenia z głównym serwerem. Na głównym serwerze jest natomiast gromadzona aktualna wersja tzw. głowa projektu. Głównym serwerem projektu została platforma GITHUB, ze względu na jej szerokie zastosowanie oraz dobrą renomę.

**3.3 Baza danych**

W projekcie użyto bazy danych MySql. Jest to relacyjna baza danych. Dane trzymane są w tabelach posiadających wzajemne relacje. Baza zapewnia szybki dostęp do danych.

**3.4 Technologie**

**3.4.1 Java**

Obiektowy język programowania.

**3.3.2 Spring**

Biblioteka napisana w języku java. Oparta o wzorzec projektowy wstrzykiwania zależności (ang. Dependency Injection). Zależności między komponentami zostają usunięte. Wszystkie potrzebne zależności zostają dostarczone przez środowisko co ułatwia projektowanie złożonych systemów informatycznych. W bibliotece Spring wszystkie komponenty nazywane są „Bean”, tworzone są i przetrzymywane w kontenerze aplikacji. Po stworzeniu wszystkich kontener sprawdza potrzebne im zależności i dostarcza je. Spring jest kompleksowym narzędziem do budowy aplikacji internetowych. Posiada wiele przydatnych modułów takich jak:

* Spring web – moduł obsługujący zapytania http. Pozwala tworzyć serwer aplikacji internetowych. W zależności od typu oczekiwanych danych zwraca strony HTML, lub dane w postaci JSON. Moduł oparty jest o wzorzec model, widok, kontroler. Model przetrzymuj dane aplikacji. Są to zazwyczaj tabele w bazie danych. Kontroler tworzy widok na podstawie danych z modelu. Widok jest odpowiedzialny za wizualizację danych.
* Spring security – moduł odpowiedzialny za bezpieczeństwo danych. Zabezpiecza on dostęp do zewnętrznych kontrolerów aplikacji które są najbardziej narażone na ataki. Moduł oparty jest o mechanizm sesji. Po zalogowaniu każdy z użytkowników otrzymuje unikalny numer sesji który otwiera mu dostęp do z góry zdefiniowanej ilość praw w aplikacji. Moduł pozwala stworzyć różne rodzaje użytkowników posiadających określone role i prawa w aplikacji.
* Spring data – moduł odpowiedzialny za łatwą obsługę bazy danych. Mapuje on klasy w javie na tabele w bazie danych. Pozwala na transakcje z baza danych bez znajomości języka SQL. Wszystko odbywa się za pomocą repozytoriów służących do obsługi poszczególnych obiektów reprezentowanych jako klasy w języku java. Ponadto bierze na siebie obsługę zależności danych i automatycznie dostarcza je w momencie gdy są potrzebne.
* Spring boot – moduł posiadający swój wewnętrzny serwer aplikacji (tomcat lub jboss). Ponadto ustawia on wszystkie parametry na domyślne wartości, przez co programista oszczędza wiele czasu na konfiguracji środowiska. Może skupić się na biznesowych założeniach projektu.

**3.3.3 Maven**

Maven to narzędzie służące do budowy projektów oraz zarządzania jego zależnościami jakim mogą być biblioteki, pliki jar oraz kody źródłowe. Służy ono również do generowania z projektu plików jar i war oraz do uruchamiania testów jednostkowych.

**3.3.4 HTML**

**HTML (HyperText Markup Language)** jest językiem używanym do tworzenia stron internetowych. Jest to także język znaczników, czyli oprócz głównego tekstu dokument zawiera w sobie informację opisujące go, jego strukturę, wygląd oraz fukcjonalność.

**3.3.5 Java Script**

Skryptowy język programowania obsługiwany przez przeglądarki internetowe. Używany głownie do budowania dynamicznych stron internetowych. Pozwala edytować kod html strony w czasie jej wyświetlania. Zwykle obsługuję on wydarzenia od użytkownika, takie jak kliknięcia komponentów na stronie, naciśnięcie przycisków i różnego rodzaju inne akcje. Pozwala on również wysyłanie zapytań http pod wskazany adres bez przeładowania strony dzięki technologii AJAX (*ang. Asynchronous JavaScript and XML*).

**3.3.6 Bootstrap**

Jest to darmowa biblioteka już z gotowymi komponentami, których można użyć bezpośrednio na stronie. Mają one zaimplementowany wygląd oraz w zależności o typu, swoją funkcjonalność. Komponenty używają biblioteki jQuery korzystającej z języka java script.

**3.3.7 AngularJS**

Angular JS to framework **JavaScript.** Służy on do prostego budowania tak zwanych „**Single**-**Page Applications**”. Czyli stron które nie przeładowują się podczas korzystania z nich. AngularJS oferuje wiele przydatnych funcji które pomagają w tworzenie stron po stronie klienta jak i po stronie serwera:

* możliwość łatwego dostępu do danych w każdym miejscu strony, ze względu na to że strony korzystaj z wspólnego modelu.
* znacząco wpływ na wydajność strony
* dzięki mechanizmowi wstrzykiwania zależności, w łatwy sposób możemy zaprojektować dość złożone strony internetowe nie martwiąc się szczególnie o architekturę
* biblioteka posiada również mechanizm „bindingów” czyli bezpośredniego połączenia ze sobą pól w formularzach html z danymi zawartymi w zmiennych modelu aplikacji w języku java script
* posiada również walidację powiązanych pól
* udostępnia wiele przydatnych dyrektyw które możemy wpisywać w odpowiednie komponenty html i tym samym wpływać na ich wygląd (ng-style), widoczność(ng-show), ilość pojawień(ng-repeat), powiązanie z modelem(ng-model), wybór odpowiedniego kontrolera (ng-controller) oraz wiele wiele innych

**3.3.8 Apache Tomcat**

Jest to kontener dla servletów. Pozwala na uruchamianie aplikacji internetowych. Cechuje się dużą wydajnością oraz możliwością konfiguracji wielu aspektów jak prawa użytkowników, porty, zabezpieczanie, szyfrowanie itd.

**4. Interfejs aplikacji**

W niniejszym rozdziale zostanie przedstawiony wygląd i budowa interfejsów użytkownika. Aplikacja posiada interfejs klienta, na którym wyświetlany jest plan zajęć, oraz interfejs edytora, służący do odczytu, dodawania i edycji danych przez uprawnionego nauczyciela. Przedstawione również zostaną sposoby obsługi interfejsów w celu uzyskania lub dodania określonych danych w aplikacji.

**4.1 Interfejs klienta**

Jest to strona internetowa napisana przy użyciu języka HTML. Wykorzystywana jest również biblioteka AngularJS, dzięki jej możliwością dynamicznego zmieniania elementów widoku HTML, oraz stylów CSS. Interfejs jest generowany dynamicznie na podstawie danych uzyskanych z głównego serwera aplikacji. Uzyskiwanie tych danych odbywa się poprzez cykliczne wysyłanie zapytań korzystając z funkcji AngularJS. Przy każdorazowej zmianie danych widok również ulega zmianom.

**4.1.1 Kontakt użytkownika z interfejsem**

Wymaganiami projektu było aby użytkownik nie miał fizycznego kontaktu z interfejsem klienta, ponieważ tablet na którym wyświetlany jest plan znajduje się w szklanej szafce. Aby sprostać wymaganiom interfejs został zaimplementowany tak, że do poprawnej pracy nie wymaga żadnej ingerencji ze strony użytkownika. Wystarczy otworzyć w przeglądarce odpowiedni adres url i strona zacznie sama obsługiwać wszystkie zdarzenia.

**4.1.2 Auto dopasowanie wielkości elementów interfejsu**

Wszystkie elementy widoku dobierają swoje wymiary na podstawie procentowej wartości wymiarów komponentu rodzica. Nie posiadają sztywno określonych wysokości oraz szerokości w pikselach. Również minimalne i maksymalne wartości nie są określone. Rozwiązanie to na ogół nie jest stosowane w projektowaniu stron internetowych ze względu na różnice w rozmiarach dzisiejszych monitorów i fakt że twórcy zazwyczaj chcą aby układ ich interfejsu był jednakowy na wszystkich rodzajach sprzętu. Ze względu jednak na fakt, że w założeniach interfejs klienta będzie wyświetlany tylko na wąskiej liczbie tabletów zdecydowano się na rozwiązanie oparte na wielkościach procentowych aby lepiej wykorzystać całe dostępne miejsce na ekranie ponieważ interfejs również ma nie posiadać możliwości przewijania widoku

**4.1.3 Wizualne różnice pomiędzy różnymi interfejsami klienta**

Widok ma możliwość kilku wizualnych zmian zadeklarowanych przez użytkownika w celu lepszego przedstawienia danych. Aplikowane są automatycznie po otrzymaniu nowych danych z serwera.

1 – wielkość czcionki wiadomości w celu lepszego rozmieszczenia tekstu.

2 – kolor szczegółów planu zajęć. Możliwa jest zmiana koloru kolumn z nazwami dni oraz wierszy zawierających godziny.

3 – kolor tła planu zajęć.

4 – kolor obramowania elementów planu zajęć

5 – kolor tła wydarzeń

6 – wielkość czcionki wydarzeń

**4.1.4 Elementy interfejsu**

1 – panel wiadomości. Są to wiadomości napisane przez nauczycieli. Wyświetlane są w kolejności chronologicznej (najmłodsze u góry panelu).

Wiadomości składają się z:

1. Nazwy adresata
2. Czasu zamieszczenia wiadomości
3. Treści

2 – plan zajęć. Są tu wyświetlane odbywające się zajęcia.

Panel składa się z kilku komponentów:

1. Tytuł – zawarta jest w nim informacja o tym czy wyświetlany plan należy do sali czy nauczyciela oraz o tym czy jest to plan dla tygodnia parzystego czy nieparzystego
2. Kolumny dni tygodnia – każda kolumna zawiera listę zajęć dla podanego dnia. Prefix „np” w nazwie dnia oznacza tydzień nieparzysty.
3. Wiersze godzin – wizualizują początek i koniec zajęć.
4. Wydarzenie - komponent z informacjami o wydarzeniu. Zawiera opis wydarzenia, nauczyciela prowadzącego.

**4.2 Interfejs edytora**

Strona HTML posiadająca komponenty z biblioteki Bootstrap. Są one używane, aby uatrakcyjnić widok oraz dodać pewne złożone mechaniki. Obsługą strony zajmuje się AngularJS. Do każdej z podstron przypisany jest odpowiedni kontroler.

**4.2.1 Logowanie**

Dostęp do strony edytora wymaga podania nazwy użytkownika oraz hasła do konta w systemie.

Strona logowania wygląda następująco.

**4.2.2 Strona główna**

Strona główna składa się zasadniczo z trzech elementów. Z komponentu navbar zawierającego nazwę strony oraz przycisk z możliwością wylogowania z konta. Komponentu nawigacji po podstronach pozwalającego na poruszanie się i korzystanie z funkcjonalności oferowanej przez edytor. Panelu wyświetlającego podstrony w zależności od wybranych opcji w panelu nawigacji. Strona główna w czasie pracy nie zostaje przeładowana, podmieniane zostają tylko podstrony w odpowiednim panelu.

**4.2.3 Panel nawigacji**

Panel nawigacji służy do poruszania się po stronie. Zawiera następujące odnośniki:

1 – moje wydarzenia. Tabela wyświetlające wszystkie wydarzenia przypisane do użytkownika.

2 – dodaj wydarzenie. Formularz służący do dodawania nowych wydarzeń.

3 – wszystkie pokoje. Tabela wyświetlająca wszystkie dostępne pokoje wraz z ich właścicielami.

4 – dodaj pokój. Formularz służący do stworzenia nowego pokoju.

5 –wszystkie wiadomości. Tabela zawierająca informację o wszystkich stworzonych przez użytkownika wiadomościach. Wyświetlane są chronologicznie (najmłodsze wydarzenie znajduje się na górze tabeli).

6 – dodaj wiadomość. Edytor nowych wiadomości.

7 – dodaj nauczyciela. Podstrona służąca do dodawania nowych nauczycieli.

**4.2.4 Podstrona „Moje wydarzenia”**

Podstrona składa się z jednej dużej tabeli posiadającej //To do kolumn.

Ostatnia kolumna zawiera dwa przyciski:

- służy do edycji wybranego wydarzenia i przekierowuje na odpowiednią stronę, na której użytkownik może tego dokonać.

- służy do usunięcia wybranego wydarzenia. Po pomyślnym usunięciu dane w tabeli zostaną przeładowane.

**4.2.5 Podstrona dodaj/edytuj wydarzenie**

Składa się z formularza umożliwiającego dodanie lub edycję wydarzenia. Edytowanie wydarzenia jest możliwe tylko poprzez przycisk na stronie ,,Moje wydarzenia” natomiast dodawanie poprzez odnośnik w panelu nawigacji. Po lewej stronie formularza znajdują się opisy pól. Natomiast bezpośrednio po lewej ich stronie pola w które należy wprowadzić dane. Nie poprany typ lub zakres danych jest sygnalizowany czerwonym podświetleniem tytułu jak i pola z danymi.

Krótki opis pól oraz sposobu ich walidacji:

1 – tytuł. Pole zawierające pełną nazwę lub skrót odbywających się zajęć. Może również zawierać nazwę grupy studentów. Pole nie może być puste. Może zawierać znaki specjalne. Jego długość jest ograniczona do //To do

2 – godzina rozpoczęcia. Jest to czas w którym rozpoczynają się zajęcia. Poprawne formaty danych wyglądają następująco g:m gg:m g:mm lub gg:mm gdzie g oznacza cyfrę godzin a m cyfrę minut. Sprawdzany również jest