

Aplikacja do organizacji pracy i przechowywania obrazów

Zespół i podział odpowiedzialności:

Wiktoria Dębowska 159860 – Front-end, kierownik

Patrycja Makowska 159846 – Bazy danych, back-end

Aplikacja została zrobiona przez obie osoby w podobnej części, jednak w większości było podzielone na zadania określone powyżej: część Patrycji Makowskiej skupiała się na backendzie aplikacji, szczególnie listy zadań oraz kalendarza, większości podstron oraz podłączeniach z bazą danych, a część Wiktorii Dębowskiej na backendzie galerii oraz na wyglądzie UI, doświadczeniach UX oraz responsywności strony.

Opis:

Aplikacja powstała w celu pomocy w organizacji swoich zadań i archiwizowaniu plików, co ma wspierać użytkownika w organizacji jego czasu, poprawieniu produktywności i uporządkowaniu ważnych zdjęć. Dodatkowym elementem aplikacji jest możliwość wspólnego organizowania pracy użytkowników jak i samodzielne rozporządzanie. Grupą docelową jest firma, która chciałaby zarządzać swoimi zadaniami i udostępniać je między sobą, wraz ze zdjęciami.

Ograniczenia:

Ograniczenia techniczne:

- Aplikacja działa na localhoście
- Baza danych aplikacji jest oparta na bazie danych SQL uruchamianej w Xamppie
- Potrzebne jest środowisko, które obsługuje Typescript, Javascript oraz framework Angular
- Aby uruchomić aplikację potrzebny jest dostęp do konsoli cmd

Ograniczenia prawne:

- Użytkownicy musieliby wyrazić zgodę na przetwarzanie danych osobowych
- Użytkownicy musieliby wyrazić zgodę na ciasteczka
- Prawa autorskie do kodu, grafik i pomysłu
- Zgodność z obowiązującymi prawami dotyczącymi dostępności osób z niepełnosprawnościami

Ograniczenia biznesowe:

- Brakujące osoby do administracji
- Obecnie system jest dostosowany do jednej firmy, wszyscy użytkownicy widzą zadania i obrazy dodane przez innych. W przyszłości należałoby dodać tutaj dodatkowe ograniczenia
- Brak przeznaczonego budżetu na utrzymanie i rozwijanie aplikacji
- Brak czasu na dalsze rozwijanie, dodawanie funkcjonalności i utrzymywanie aplikacji

Założenia projektowe:

Głównym założeniem aplikacji była praca w zakresie danej firmy. Użytkownicy aplikacji mieli być zrzeszeni, co powodowałoby udostępnianie elementów aplikacji, tj. zadania i zdjęcia w obrębie wszystkich użytkowników, aktualnie bez możliwości nadania żadnych restrykcji. Aplikacja miała być korzystna dla małych jak i dużych firm, różnica polegała na ilości wyświetlanych elementów. Projekt był obsługiwany przez platformy, które posiadają środowisko obsługujące Javascript i jego

frameworki oraz usługę localhost. Ze względu na to, że została użyta baza danych w oprogramowaniu MySQL aplikacja połączona była z programem XAMPP.

Funkcjonalności w skrócie:

- Możliwość założenia konta
- Możliwość logowania i wylogowania z konta
- Możliwość usunięcia konta
- Możliwość zmiany hasła
- Możliwość dodania zadań do listy
- Możliwość dodania statusu zadania
- Możliwość dodania deadline zadania
- Możliwość edycji treści zadania
- Możliwość edycji daty zadania
- Możliwość edycji statusu zadania
- Możliwość usunięcia zadania z listy
- Możliwość filtrowania zadań po ich statusie
- Wyświetlanie zadań w liście
- Wyświetlanie zadań oznaczonych deadline'm w kalendarzu
- Wyświetlanie ilości zadań na miesiąc
- Możliwość wyświetlania zadań innych użytkowników wraz z loginem osoby, do której to zadanie przynależy w kalendarzu
- Rozdzielenie kalendarza na własny i całej społeczności
- Możliwość dodawania zdjęć do galerii
- Wyświetlanie zdjęć w galerii
- Usuwanie zdjęć w galerii
- Możliwość wyświetlania zdjęć użytkowników wraz z loginem osoby, do której to zdjęcie przynależy w galerii
- Rozdzielenie galerii na własną i społeczności
- Responsywność na desktop i ekrany mobilne

Funkcjonalności w formie User Stories:

- Jako użytkownik aplikacji chcę móc założyć konto, aby móc się zalogować.
- Jako użytkownik aplikacji chcę móc zalogować się, aby mieć dostęp do zawartości aplikacji.
- Jako użytkownik aplikacji chcę móc się wylogować, aby nikt nie miał dostępu do moich danych.
- Jako użytkownik aplikacji chcę móc usunąć konto, gdy nie będzie mi już potrzebne.
- Jako użytkownik chcę móc zmienić hasło, gdybym zapomniał, jakie było poprzednie.
- Jako użytkownik aplikacji chcę móc dodawać zadania do listy, aby móc planować i zarządzać swoją pracą.
- Jako użytkownik aplikacji chcę móc dodawać status do zadania, aby wiedzieć, jak ważne ono jest.
- Jako użytkownik aplikacji chcę móc ustawić deadline zadania, aby mieć kontrolę nad terminami.
- Jako użytkownik aplikacji chcę móc edytować treść zadania, aby móc wprowadzać w niej zmiany, gdy zajdzie taka potrzeba.
- Jako użytkownik aplikacji chcę móc edytować deadline zadania, aby dostosować termin wykonania do zmieniających się planów.

- Jako użytkownik aplikacji chcę móc edytować status zadania, aby aktualizować jego wagę.
- Jako użytkownik aplikacji chcę móc usuwać zadania z listy, aby móc pozbyć się tych, które są już nieaktualne.
- Jako użytkownik aplikacji chcę móc filtrować zadania według ich statusu, aby łatwiej zarządzać pracą i skupić się na konkretnych zadaniach.
- Jako użytkownik aplikacji chcę widzieć zadania w formie listy, aby mieć szybki i przejrzysty przegląd swoich obowiązków.
- Jako użytkownik aplikacji chcę widzieć zadania z deadline'em w kalendarzu, aby mieć wizualne przypomnienie o zbliżających się terminach.
- Jako użytkownik aplikacji chcę widzieć liczbę zadań przypadających na dany miesiąc, aby móc lepiej zarządzać swoim obciążeniem pracą.
- Jako użytkownik aplikacji chcę widzieć w kalendarzu zadania innych użytkowników wraz z loginem ich właściciela, aby móc śledzić postępy zespołu.
- Jako użytkownik aplikacji chcę mieć rozdzielony widok kalendarza na mój własny oraz społeczności, aby łatwiej oddzielać zadania prywatne od zespołowych.
- Jako użytkownik aplikacji chcę móc dodawać zdjęcia do galerii, aby dzielić się wizualnymi materiałami.
- Jako użytkownik aplikacji chcę móc przeglądać zdjęcia w galerii, aby mieć dostęp do wszystkich dodanych obrazów.
- Jako użytkownik aplikacji chcę móc usuwać zdjęcia z galerii, aby móc zarządzać swoją kolekcją.
- Jako użytkownik aplikacji chcę widzieć zdjęcia innych użytkowników wraz z loginem właściciela, aby wiedzieć, kto je dodał.
- Jako użytkownik aplikacji chcę mieć rozdzieloną galerię na moją własną i społeczności, aby móc łatwo przeglądać zdjęcia według ich właścicieli.
- Jako użytkownik aplikacji chcę, aby interfejs był responsywny na komputerach i urządzeniach mobilnych, aby korzystać z aplikacji wygodnie na każdym ekranie.

Opis funkcjonalności:

Funkcjonalność	Opis funkcjonalności
Możliwość założenia konta	Użytkownik może założyć konto poprzez rejestrację. Jeśli dana nazwa jest już zajęta wyskoczy odpowiedni komunikat. Po wpisaniu danych do rejestracji (login i hasło) wyskoczy komunikat o sukcesie i użytkownika przeniesie na stronę logowania.
Możliwość logowania i wylogowania z konta	Użytkownik może zalogować się po uprzednim założeniu konta. Jeśli coś będzie nie tak komunikat poinformuje o tym użytkownika. Po zalogowaniu użytkownik może się wylogować co spowoduje brak możliwości zobaczenia reszty funkcjonalności. Jeśli użytkownik nie jest zalogowany wyświetla się jedynie strona główna, rejestracja oraz logowanie.
Możliwość usunięcia konta	Użytkownik może usunąć konto co spowoduje usunięcie go z bazy danych i przerzucenie na stronę logowania. Wszystkie zadania i pliki tego użytkownika są również usuwane.
Możliwość zmiany hasła	Użytkownik może zmienić swoje hasło. Musi wpisać je dwa razy: jeśli nie będą zgodne pojawi się taka informacja, jeśli będą w porządku wyskoczy komunikat o sukcesie.

Możliwość dodania zadania do listy	Użytkownik może dodać zadanie do listy zadań poprzez formularz.
Możliwość nadania statusu zadania	Użytkownik może wybrać status zadania – bardzo ważne, ważne, nieważne, luźne oraz może wybrać brak statusu.
Możliwość dodania deadline zadania	Użytkownik może wybrać datę z małego kalendarza lub wpisać ręcznie do kiedy dane zadanie ma być zrobione.
Możliwość edycji zadania, Możliwość dodania statusu zadania, Możliwość dodania deadline zadania	Użytkownik ma możliwość edycji treści, statusu lub deadline zadania po dodaniu go do listy poprzez kliknięcie przycisku „edytuj”, następnie „zapisz”. Może również zrezygnować z tych zmian klikając anuluj.
Możliwość usunięcia zadania z listy	Użytkownik może usunąć zadanie, wtedy jego dane automatycznie są usuwane się z listy i z tabeli z bazy danych.
Możliwość filtrowania zadań wg. statusu	Użytkownik może filtrować zadania po statusie. Z listy wybieranej może wybrać podane statusy oraz wszystkie lub bez statusu. Wyświetlają się jedynie zadania z wybranym statusem, wszystkie lub bez statusu.
Wyświetlanie zadań w liście	Dodane zadania wyświetlają się w formie listy – najnowsze wyświetlają się na samej górze. W przypadku braku statusu wyświetla się „ – „.
Wyświetlanie zadań oznaczonych deadline’em w kalendarzu	Kalendarz dopasowujący się do rzeczywistego. W formie tablicy wyświetlają się dni i tygodnie. Nad kalendarzem widnieje informacja jaki to miesiąc i rok. W komórkach podświetla się dzisiejszy dzień. Poprzez przycisk „Dzisiaj” widok kalendarza wraca na dzisiejszą datę. Można przesuwać miesiące poprzez < i >.
Wyświetlanie ilości zadań na miesiąc	W danym miesiącu wyświetla się licznik zadań, które są wpisane do kalendarza na wybrany miesiąc.
Wyświetlanie zadań w kalendarzu	W kalendarzu, dni z podanymi deadline’ami pokazują się za pomocą kropki pod numerem. Po wybraniu dnia obok wyświetla się lista zadań na dany dzień.
Rozdzielenie kalendarza na własny i całej społeczności	Kalendarz jest rozdzielony na część własną, gdzie wyświetlają się tylko zadania zalogowanego użytkownika oraz część wspólną, gdzie wyświetlają się zadania wszystkich użytkowników.
Możliwość wyświetlania zadań innych użytkowników wraz z loginem osoby, do której to zadanie przynależy w kalendarzu	W przypadku rozdzielonego kalendarza Społeczności, przy zadaniach wyświetla się informacja o użytkowniku, który dodał to zadanie.
Możliwość dodawania zdjęć do galerii	Użytkownik ma możliwość dodania zdjęcia poprzez kliknięcie przycisku w dolnym rogu aplikacji. Po wybraniu pliku i zatwierdzeniu pojawi się ono w galerii oraz folderze dedykowanym przechowywaniu plików w aplikacji – dane o nim przechowywane są w tabeli w bazie danych.
Wyświetlanie zdjęć w galerii	Zdjęcia wyświetlają się w galerii pod tytułem podstrony. Wyświetlają się one w kolumnach asymetrycznych, nadając aplikacji estetyczności.
Usuwanie zdjęć z galerii	Po najechnaniu na obraz wyświetla się możliwość usunięcia pliku. Po usunięciu znika on także z tabeli w bazie danych i folderu aplikacji

Rozdzielenie galerii na własną i społeczności	Podstrony galerii podzielone są na osobistą i publiczną. Osobista wyświetla pliki danego użytkownika, a podstrona społeczności wyświetla wszystkie pliki. W obu galeriach najnowsze obrazy pokazują się na samej górze.
Możliwość wyświetlania zdjęć użytkowników wraz z loginem osoby, do której to zdjęcie przynależy w galerii	We wspólnej galerii pliki wyświetlają się wraz z informacjami o użytkowniku, który przesłał dany plik. W zależności, kto wysłał najnowszy plik ten pojawi się na górze.
Responsywność na desktop i ekrany mobilne	Aplikacja zmienia swój wygląd responsywnie w zależności od wielkości ekranu.

Planowane technologie:

Planowane technologie, które miały zostać użyte do tworzenia aplikacji to: Typescript lub Javascript, Angular lub React, Node.js, Next.js, Python z frameworkiem Flask lub Django, HTML, CSS oraz framework Bootstrap, bazy danych SQL i Figma.

Użyte technologie:

Ostateczną decyzją zostało użycie frameworka Angular w najnowszej wersji, a w związku z tym użycie Typescript oraz zestawu HTML z kaskadowymi arkuszami stylów (CSS) w celu estetycznego oraz tematycznego layoutu strony. Zadbano także o responsywność i czytelność. W części back-endowej został wykorzystany framework Express.js, na którym została oparta logika łączenia z bazą danych, przekazywania i edycji informacji (zadań, zdjęć, userów). W pracy z datami i kalendarzami użyto biblioteki „luxon”, która buduje pracę kalendarza, usprawnia edycję dat oraz ich formatowanie. Do utworzenia bazy danych zostało wykorzystane oprogramowanie MySQL. Do stworzenia wstępnego wyglądu aplikacji (wireframe) i ostatecznego wyglądu została użyta aplikacja Figma.

Architektura:

Aplikacja SPA ma posiadać podział na moduł zarządzania użytkownikiem, moduł zarządzania zadaniami, moduł organizacji plików, moduł przechowywania danych i moduł interfejsu użytkownika. Wybrany środowiskiem programistycznym jest Visual Studio Code.

Aplikacja ma 9 podstron: logowanie i rejestracja, strona główna, lista zadań, kalendarz, galeria i ustawienia, gdzie kalendarz i galeria mają dodatkowy podział: moja galeria/mój kalendarz i społeczność dla kalendarza, jak i galerii

Moduły:

1. Moduł zarządzania użytkownikiem:

Funkcjami tego modułu jest możliwość rejestracji użytkownika, zapisywania jego logowania i możliwości wylogowania. Technologia użyta do tego modułu jest język Javascript za pomocą frameworka Express.js. Logowanie użytkownika jest zapisane w sesji, a do zarejestrowania użytkownika na stronie odpowiada baza danych przy pomocy oprogramowania MySQL. Zapisywane są w tabeli users dane użytkownika: user ID, login i password. Za pomocą alerta wyświetlana jest również informacja czy dany login już istnieje.

2. Moduł zarządzania zadaniami:

Użytkownik może dodawać zadania, usuwać je, dodawać wagę(bardzo ważny, ważny, nieważny, luźny, brak statusu) oraz datę ukończenia(deadline). Użytkownik może edytować zadanie poprzez zmianę treści, statusu i deadline. Dane o zadaniach trzymane są w tabeli tasks: ID zadania, task_name, task_status, deadline i pobiera ID zalogowanego użytkownika (userID). Zadania z określoną datą dodają się do kalendarza własnego oraz wspólnego i są wyświetlane w

kalendarzach. Wspólny kalendarz ma również informację o użytkowniku, do którego to zadanie należy (pobiera wszystkie dane z tabeli tasks gdzie jest podany deadline w połączeniu z userID z tabeli users). Interakcje użytkownika ze stroną są zaprogramowane poprzez język Typescript w frameworku Angular.

3. Moduł zarządzania zadaniami - kalendarz:

Użytkownik ma możliwość wyświetlania zadań z określonym deadline właśnie w kalendarzu. Pojawiają się one wtedy w postaci listy, gdy zostanie wybrany dzień z zadaniami. Technologie, których użyliśmy to przede wszystkim biblioteka luxon, Javascript oraz Typescript. Kalendarz pobiera zadania, w zależności od zalogowanego usera oraz wszystkie zadania z tabeli z bazy danych, ale jedynie te, które mają deadline. Wyświetla je następnie w komponentach kalendarza.

4. Moduł organizacji plików:

Użytkownik ma możliwość dodawania swoich plików zdjęciowych na stronę. Dane zdjęć trzymane są w tabeli files, gdzie są tylko: ID plików i użytkownika, który dany plik wstawił, nazwa, która przy dodaniu do serwera zmienia nazwę poprzez dodanie do istniejącej nazwy daty wstawienia pliku oraz czas publikacji zdjęcia aby móc wyświetlać elementy w galerii najnowsze zdjęcia na górze strony. Istnieje podział na swoją galerię oraz wspólną, gdzie w publicznej przy każdym zdjęciu wyświetla się przypisany użytkownik poprzez złączenie elementów galerii wszystkich użytkowników. Technologia jaka została użyta do tej części projektu to Typescript, język Javascript w frameworku Express.js i baza w MySQL.

5. Moduł przechowywania danych:

Dane są przechowywane w bazie danych, do której połączenie poprowadzone jest przez back-end w Express.js, który przekazuje dane do plików w folderze services – zajmują się one podstawowymi elementami łączenia frontendu aplikacji z jego backendem. Użyte technologie to baza danych SQL w MySQL, framework Express.js oraz Angular.

6. Moduł interfejsu użytkownika:

W module zawarta jest budowa wizualna oraz dynamika i responsywność aplikacji. Przy użyciu programu Figma powstał pomysł interfejsu użytkownika, a następnie HTML, CSS, Typescript oraz ich frameworki zostały użyte do zbudowania samej aplikacji.

Proces powstawania aplikacji:

Na samym początku w planach było użycie Pythona jako backendowego języka, jednak ostatecznie użyty został framework Express.js, który działa poprzez Javascript. Jako, że celem było wykorzystanie Angulara i języka Typescript ostatecznie decyzja zapadła na takim połączeniu. Baza danych w MySQL została wybrana poprzez znajomość działania tego oprogramowania, jednak na etapie planowania nie zostały wykluczone inne typy jak np. MongoDB. Planem na to jak umożliwić użytkownikowi przesyłanie plików na stronie i ich wyświetlanie ostatecznie zostały dwie wersje REST API – poprzez trzymanie danych obrazów w tabeli już istniejącej bazy danych lub poprzez JSON. Kod został napisany poprzez łączenie się z istniejącą bazą danych dla ułatwienia pracy.

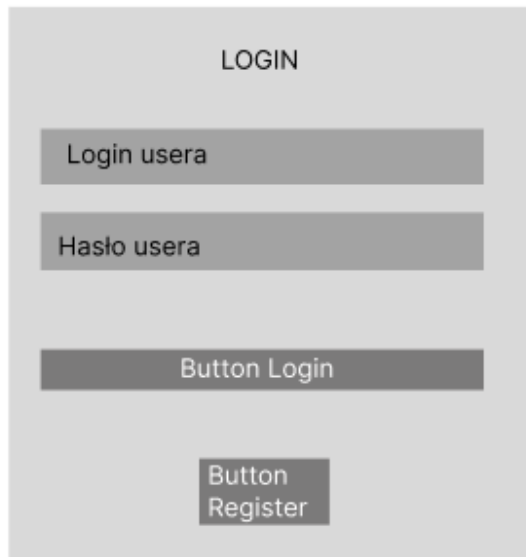
Problemy z jakimi się spotkałyśmy:

1. Problem z konwersją dat – biblioteka luxon zapewnia formatowanie dat za pomocą np. Europe/Warsaw, ale też utc+1. Na początku skorzystałyśmy z drugiej opcji, ale okazało się, że data zadania cofała się do tyłu o 1 godzinę, przez co cofało dzień o 1 do tyłu.
2. Drobnny błąd, który wynikł z nieuwagi, to postawienie dwóch serwerów backendowych na tych samym porcie 3000. Przez przypadek oba pliki (index.js oraz server.js) uruchamiały się, na tym samym porcie, przez co pojawiał się błąd.

3. Połączenie back-endu plików z folderem przechowującym obrazy – folder przechowujący element ma tendencję do znikania przy pobraniu plików z GitHuba – konieczna była uważność czy element ten został dodany.

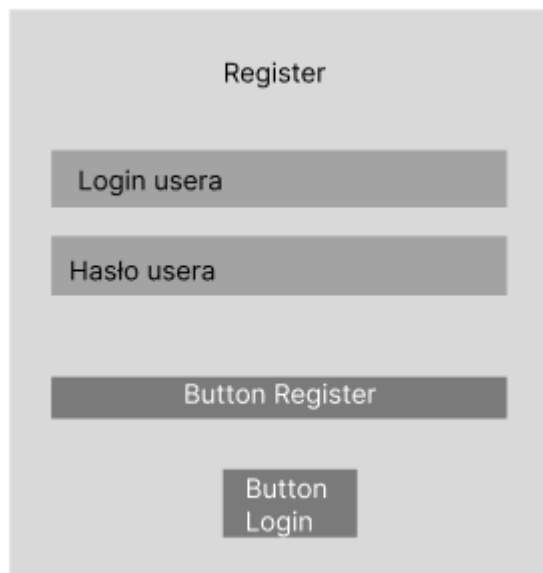
Koncepcja wyglądu podstron – lista pogładowa:

Strona logowania:



A wireframe diagram of a login page. It features a light gray background with a darker gray title bar at the top containing the text "LOGIN". Below the title bar, there are two input fields: the first is labeled "Login usera" and the second is labeled "Hasło usera". Below these fields is a wide button labeled "Button Login". At the bottom of the form is a smaller button labeled "Button Register".

Strona rejestracji:



A wireframe diagram of a registration page. It features a light gray background with a darker gray title bar at the top containing the text "Register". Below the title bar, there are two input fields: the first is labeled "Login usera" and the second is labeled "Hasło usera". Below these fields is a wide button labeled "Button Register". At the bottom of the form is a smaller button labeled "Button Login".

Strona dodawania zadań:

Your To-Do List

New task

Date dd.mm.rrrr Priority

Add Task

Task Edit Delete

Task Edit Delete

Strona kalendarza:

Calendar

month, year Counter of tasks

			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

Tasks for day

task

task

Strona galerii:

Galeria

photo photo photo

photo photo photo

add photo

Strona ustawień:

Edit Profile

Login usera

Hasło usera

Update Profile

Delete account

Dwa rodzaje menu: 1. Dla dużych ekranów, 2. Dla małych ekranów.

Logo

My lists

Calendar

Gallery

Settings

Login/ Register
Logout

My calendar

My gallery

Social

Social

Menu 1. Dla dużych ekranów

-
-
-

logo

Login/ Register
Logout

My lists

Calendar

My calendar

Social

Gallery

My gallery

Social

Settings

Menu 2. Dla małych ekranów

Wireframe interfejsu użytkownika:

Strona główna:

BeFreepared [Home](#) [My Lists](#) [Calendar](#) [My Archives](#) [Zaloguj](#) [Zarejestruj się](#)

Strona rejestracji/logowania:

BeFreepared [Home](#) [My Lists](#) [Calendar](#) [My Archives](#) [Zaloguj](#) [Zarejestruj się](#)

Login

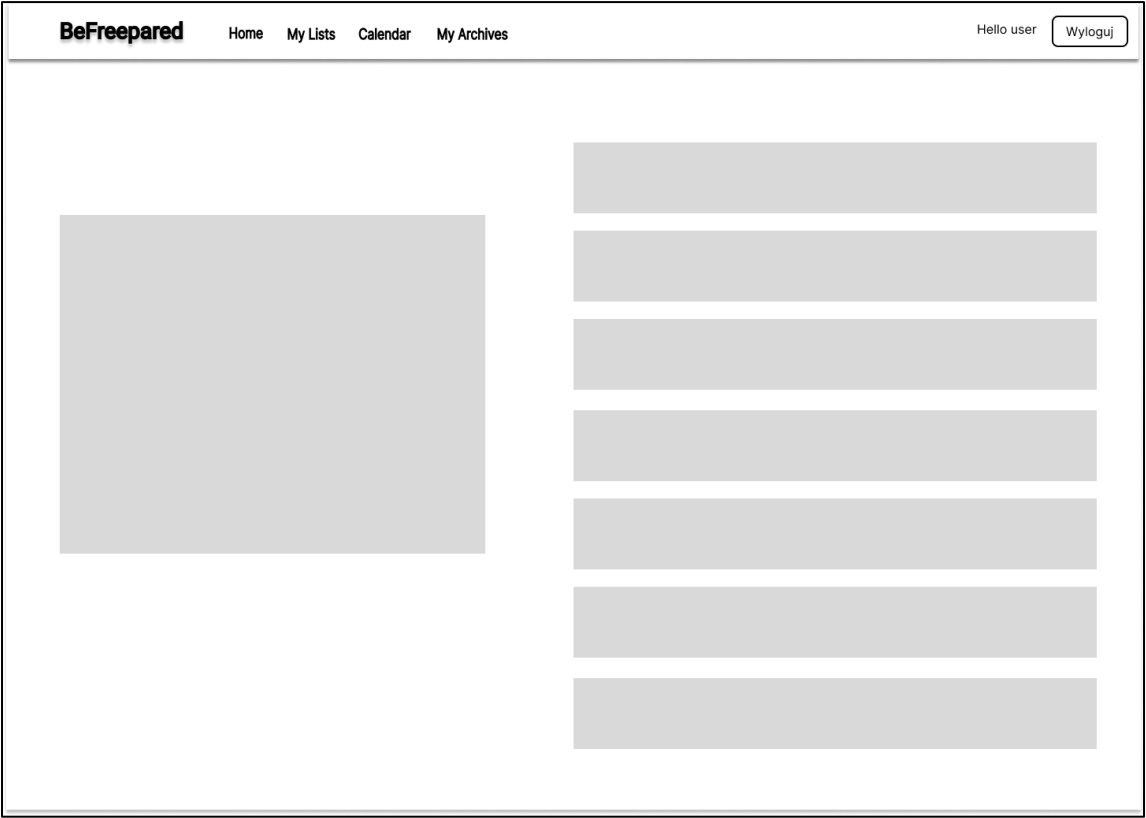
Hasło

Zarejestruj się

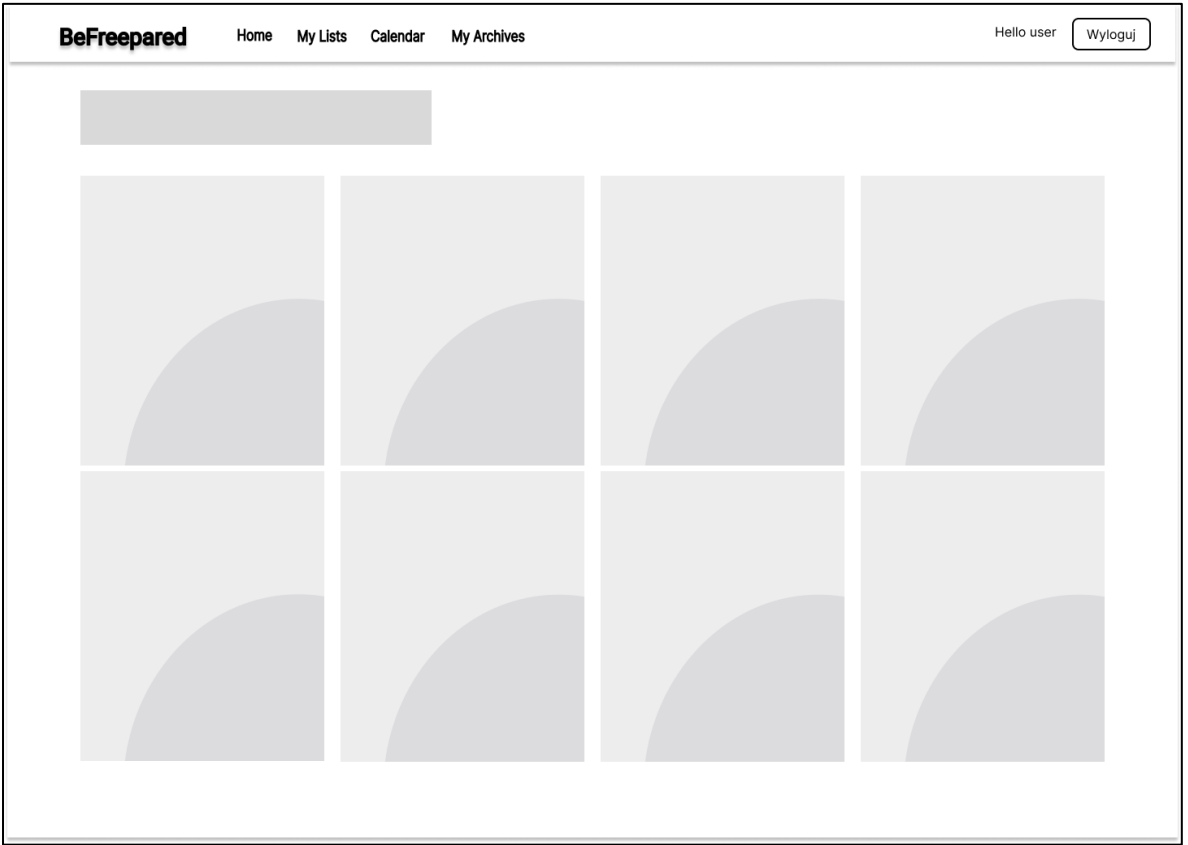
Strona listy zadań:



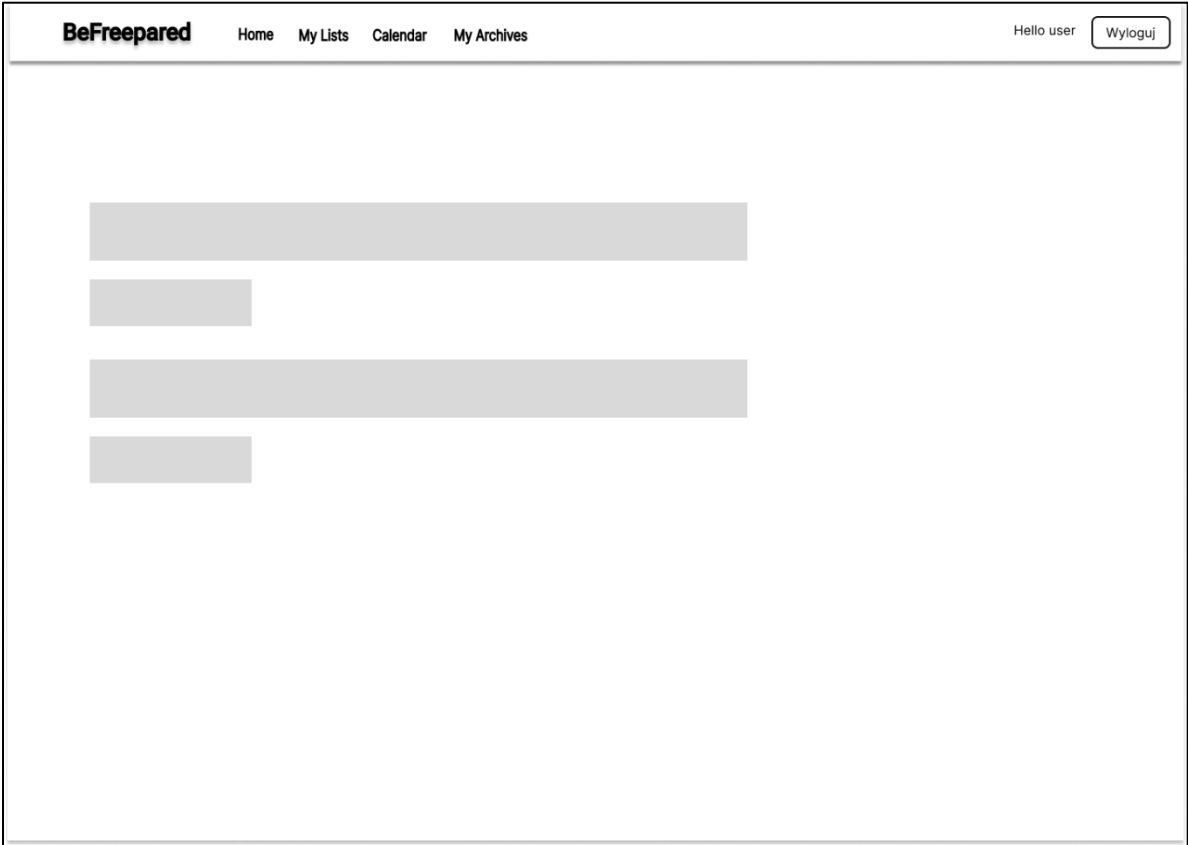
Strona kalendarza:



Strona galerii:



Strona ustawień:



Opis użytkownika:

Użytkownik, aby skorzystać z naszej aplikacji musi założyć konto. Bez zalogowania dostępna będzie jedynie strona główna, logowania oraz rejestracji. Po kliknięciu na inne rzeczy w menu, użytkownik jest przenoszony na stronę logowania. Użytkownik może się zarejestrować, ale w przypadku, gdy istnieje już podany login wyskoczy takie powiadomienie. Po poprawnej rejestracji użytkownik jest przenoszony na stronę logowania, gdzie może wpisać swoje dane. Po poprawnym zalogowaniu przenosi się na stronę główną, jednak przyciski „zaloguj” oraz „zarejestruj” znikają, a pojawia się przycisk wyloguj obok którego widnieje „cześć user”, gdzie user to nazwa użytkownika podana przy rejestracji. Użytkownik może teraz wykonać kilka czynności. Może przejść w menu do zakładki „Moje zadania” gdzie ma możliwość dodania zadania. Aby to zrobić musi wpisać nazwę zadania i opcjonalnie wybrać status z podanej listy rozwijanej oraz również opcjonalnie wybrać datę deadline, do kiedy to zadanie ma być wykonane. Po zatwierdzeniu dodania zadania poprzez przycisk „dodaj zadanie”, wyświetli się ono na liście poniżej. W przypadku, gdy użytkownik pominie wybranie statusu lub daty, w miejscu, w którym powinny się one wyświetlać będzie zwykły znak „ – „. W każdej chwili użytkownik może edytować zadanie poprzez przycisk „edytuj”. Wtedy pojawia się formularz edycji zadania, gdzie jest możliwość zmiany treści zadania, zmiany statusu również z listy rozwijanej oraz zmiana daty deadline. W trakcie edycji można anulować zmiany, poprzez wyjście z okna formularza poprzez „anuluj”. Użytkownik ma również możliwość usunięcia dodanego zadania, po tym jak pojawi się na liście. Po dodaniu zadania z deadline pojawi się ono w zakładce kalendarz. Dzieli się on na dwie podkategorie: Mój kalendarz oraz Społeczność. Cechami wspólnymi obu podstron są sposób wyświetlania kalendarza, licznik zadań, podświetlanie dnia dzisiejszego, menu poruszania się po miesiącach. Kalendarz wyświetla się w postaci tablicy 30/31 dni, 4/5 tygodni (w dostosowaniu do miesiąca). Można przesuwać się pomiędzy miesiącami poprzez strzałki << i >>, a także pomiędzy nimi znajduje się przycisk „Dzisiaj”. Po jego kliknięciu kalendarz wraca do dnia dzisiejszego, bez znaczenia w jakim fragmencie kalendarza byłby użytkownik. Obok tego menu wyświetla się informacja na jakim miesiącu obecnie się znajdujemy oraz licznik zadań na dany miesiąc. Dodatkowo w kalendarzu podświetla się dzień dzisiejszy innym kolorem. W kalendarzu użytkownik może znaleźć również kropki, które oznaczają jakieś zadanie/zadania na dany dzień. Po kliknięciu na taki dzień wyświetlą się wszystkie zadania z tego dnia. Po wybraniu dnia bez zadań wyświetli się prosta informacja o braku zadań. W moim kalendarzu wyświetlają się zadania jedynie zalogowanego użytkownika, natomiast we „wspólnym” kalendarzu, który wyświetla się po wybraniu podkategorii „Społeczność” wyświetlane będą wszystkie zadania, wszystkich użytkowników. W celu odróżnienia czyje zadanie, należy do kogo po wybraniu w kalendarzu wspólnym dnia z zadaniem, obok jego treści wyświetli się również nazwa osoby, która to zadanie dodała. W podstronie galerii, po najechaniu na pasek podstron i wybraniu górnej podstrony z elementu, „dropdown” użytkownik ma możliwość wstawienia własnych zdjęć. Jest to prosta i minimalistyczna podstrona w celu ułatwienia korzystania różnym typom użytkowników. Poprzez kliknięcie w jeden przycisk w dolnym rogu strony aplikacja otwiera okienko wyboru plików, co prowadzi do prostego wyboru zdjęcia, które pojawi się zamiast napisu nakierowującego użytkownika, aby wstawił element do galerii. Podkategorią galerii, tak jak w kalendarzu, jest zakładka społeczność, którą można wybrać po najechaniu na „Galeria” i wybranie dolnej podstrony z elementu, „dropdown”. Tam pojawiają się elementy wrzucone przez wszystkich użytkowników w celu wymiany zdjęciami poprzez użytkowników aplikacji. Dodatkowo użytkownik ma możliwość edycji swojego hasła. Od tego są ustawienia, gdzie użytkownik po wpisaniu dwukrotnie tego samego nowego hasła i zapisaniu zmian, dokona jego zamiany. Jeśli te hasła nie byłyby takie same pojawiłaby się taka informacja. Użytkownik ma również możliwość usunięcia swojego konta. Po potwierdzeniu swojej decyzji, wszystkie zadania użytkownika zostają usunięte, on powraca na stronę logowania.

Plan realizacji, szacowany harmonogram:

Szacowany czas wykonania projektu to ok. miesiąc (dokładnie zajęło nam to 23 dni).

Etapy projektu umiejscowione po kolei:

1. Wygenerowanie środowiska dla Angulara oraz wszystkich potrzebnych komponentów.
2. Stworzenie szkieletu strony głównej, logowania oraz rejestracji w html.
3. Stworzenie bazy danych. Połączenie aplikacji z nią.
4. Dodanie logiki rejestracji oraz logowania.
5. Rozwinięcie komponentu navbar.
6. Dodanie routes do pliku app.routes.
7. Rozwinięcie komponentu dodawania zadań.
8. Stworzenie serwisu, logiki oraz struktury dodawania zadań.
9. Rozwinięcie komponentu kalendarza.
10. Stworzenie logiki wyświetlania kalendarza, zadań, licznik zadań oraz różnych miesięcy.
11. Rozwinięcie komponentu galeria.
12. Stworzenie logiki, mechaniki i wyświetlania dodawanych zdjęć.
13. Rozwinięcie komponentu ustawień.
14. Dodanie logiki zmiany hasła oraz usuwania konta.
15. Rozwinięcie komponentu kalendarza oraz galerii, rozdzielenie ich na własne i wspólne zadania oraz zdjęcia.
16. Dodanie spójnego wyglądu wszystkich komponentów oraz głównych stron aplikacji.
17. Zadbanie o responsywność strony.
18. Stworzenie dokumentacji, kończącej i podsumowującej projekt.

Diagram przypadków użycia użytkownika:

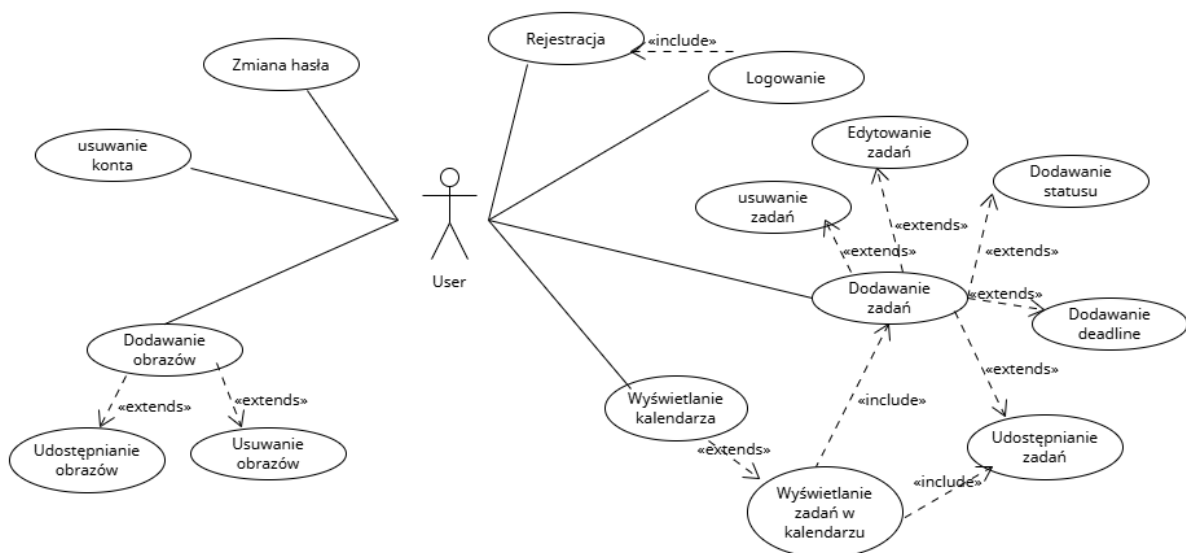


Diagram klas:

class: CalendarPageComponent -allMeetings: { [date: string]: string[] } = {}; -ngOnInit(): void	class: CalendarComponent -private _externalMeetings = signal<Meeting[]>(); -private userId: string = ''; -today: Signal<DateTime> = signal(DateTime.now()); -firstDayOfActiveMonth -activeDay: WritableSignal<DateTime> -weekDays: Signal<string[]> -daysOfMonth: Signal<DateTime[]> -monthlyTaskCount: Signal<number> -ngOnInit() -fetchTasks() -goToPreviousMonth() -goToNextMonth() -goToToday() -selectDay()
class: MyCalendarPageComponent -tasks: Task[] = []; -newTask: Task{} -userId: string = ''; -formatDate(dateString: string) -ngOnInit() -fetchTasks()	
class: CommunitygalleryComponent -gallery: GalleryImage[] = []; -searchUsername: string = ''; -ngOnInit() -loadGallery()	
class: GalleryComponent -gallery: GalleryImage[] = []; -userId: any; -selectedFile -ngOnInit() -loadGallery() -onFileSelected(event: any) -uploadImage(event: Event) -deleteImage(fileId: number)	class: LoginComponent -login = ''; -password = ''; -loginUser() -goToSignUp()
	class: SignInComponent -login = ''; -password = ''; -register(form: NgForm) -goToLogin()
class: HomeComponent -dayOfWeek() -isLoggedIn()	class: MyListsComponent -tasks: Task[] = []; -newTask: Task{} -userId: string = ''; -editingTask -selectedStatus: string = ''; -formatDate(dateString: string) -ngOnInit() -addTask() -startEdit(task: Task) -saveEdit() -cancelEdit() -deleteTask(id: string) -applyFilter() -fetchTasks()
class: NavbarComponent -unsubscribe\$: Subject<boolean> -username: string = ''; -menuOpen = false; -toggleMenu() -ngOnInit() -ngOnDestroy() -isLoggedIn() -logout()	class: SettingsComponent -newLogin: string = ''; -newPassword: string = ''; -confirmPassword: string = ''; -loginTaken: boolean = false; -passwordMismatch: boolean = false; -updateProfile() -saveChanges(userId: number) -deleteAccount()
class: UserService -private user\$ -getUser() -setUser() -removeUser()	class: GalleryService -private baseUrl -uploadImage(file: File, userId: number) -getGallery(userId: number) -deleteImage(userId: number, fileId: number) -getCommunityGallery()
class: TaskService -private apiUrl -addTask(task: Task) -getTasksForUser(userId: string, status) -updateTask(task: Task) -deleteTask(id: string) -getAllTasks()	

Linki do repozytoriów github, na których projekt został opracowany:

<https://github.com/Pola75m/BeFreepared.git>

<https://github.com/WikDebo/upload-app.git>

Ostateczne repozytorium:

<https://github.com/Pola75m/BeFreeparedApp.git>

Pseudonimy:

WikDebo / wikd – Wiktoria Dębowska 159860

Pola75m – Patrycja Makowska 159846

Instrukcja instalacji aplikacji:

1. Aby zainstalować aplikację należy wejść w repozytorium ostatecznej aplikacji zapisany powyżej.
2. Aby aplikacja została włączona należy zainstalować najnowszą wersję frameworka angular poprzez wpisanie do konsoli „Command prompt”:
npm install -g @angular/cli
3. Wpisać następnie: npm install oraz npm i luxon-angular
4. Następnie należy pobrać lub sklonować aplikację poprzez git
5. Aplikacja powinna mieć folder „node modules” obu folderów back-end, jednak w razie konieczności należy wpisać:
Do folderu backend-files:
npm init -y
npm i express multer mysql2 cors

Do folderu task-backend:

npm init -y

npm i express multer mysql2 cors

6. Do uruchomienia aplikacji należy mieć zainstalowaną aplikację XAMPP z modulem MySQL, po uruchomieniu aplikacji należy wejść w przeglądarce w localhost/phpmyadmin, gdzie następnie w konsoli SQL należy wpisać:

```
CREATE DATABASE befrepareddb;
```

7. Następnie stworzyć tabele plików, użytkowników i zadań`:

```
CREATE TABLE files (  
  fileId INT AUTO_INCREMENT PRIMARY KEY,  
  userId INT,  
  file_name VARCHAR(255),  
  uploaded_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  FOREIGN KEY (userId) REFERENCES users(Uid) ON DELETE CASCADE  
);
```

```
CREATE TABLE users (  
  Uid INT AUTO_INCREMENT PRIMARY KEY,  
  login varchar(255),  
  password VARCHAR(255)  
);
```

```
CREATE TABLE tasks (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  task_name VARCHAR(255),  
  task_status VARCHAR(255),  
  deadline DATE,  
  userId INT,  
  FOREIGN KEY (userId) REFERENCES users(Uid) ON DELETE CASCADE  
);
```


W kodzie ze względu na stałe testy na różnych komputerach aplikacja nie posiada konkretnego użytkownika w phpmyadmin

8. Aby uruchomić aplikacje należy:

- W folderze task-backend w konsoli cmd wpisać: `node index.js`
- W folderze backend-files w konsoli cmd wpisać: `node server.js`
- W głównym folderze BeFreepared w konsoli cmd wpisać: `ng serve`

Następnie udac się do wskazanego linku (`localhost:4200`)