



Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie

Wydział Zarządzania

Praca licencjacka

Zarządzanie czasem oraz wykorzystanie technik zarządzania czasem
w aplikacji internetowej utworzonej w technologiach Java i Angular

Time management and use of time management techniques in application
created with Java and Angular technologies.

Autor: Dominik Brzęk

Kierunek studiów: Zarządzanie

Opiekun pracy: dr Beata Basiura

Spis treści

Spis treści	2
Wstęp	4
1. Teoria zarządzania czasem	6
1.1 Wprowadzenie	6
2. Popularne techniki oraz metody planowania zadań	8
2.1 Opis Technik zarządzania czasem	8
2.1.1 Technika Pomodoro	8
2.1.2 Macierz Eisenhowera	11
2.1.3 Prawo Parkinsona	12
2.1.4 Zasada Pareto	14
2.2 Skuteczność treningów zarządzania czasem	15
3. Specyfikacja techniczna wykonanej aplikacji	18
3.1 System kontroli wersji	18
3.2 Wymagania biznesowe	18
3.3 Technologia wykorzystana po stronie serwera (Backend)	19
3.3.1 Opis stosu technologicznego	19
3.3.2 Opis modelu bazodanowego	19
3.3.3 Architektura aplikacji	20
3.3.4 Opis usług dostępnych w aplikacji	22
3.4 Technologia wykorzystana po stronie przeglądarki (Frontend)	23
3.4.1 Opis stosu technologicznego	23
3.4.2 Architektura aplikacji	24
4. Omówienie aplikacji oraz wykorzystanie opisanych technik zarządzania czasem	26
4.1 Opis funkcjonalności dostępnych dla użytkownika	26
4.1.1 Dodawanie zadań oraz podzadań	26
4.1.2 Czasomierz Pomodoro	27

4.1.3	Przypisywanie priorytetów za pomocą macierzy Eisenhowera.....	28
4.2	Prezentacja działania aplikacji w praktyce	28
4.2.1	Student zarządzania	29
4.2.2	Java Developer.....	29
4.2.3	Pracownik korporacji	30
4.2.4	Zalety i wady testowanej aplikacji.....	31
4.2.5	Porównanie utworzonej aplikacji z istniejącymi narzędziami.....	32
	Podsumowanie	33
	Bibliografia	34
	Spis tabel i ilustracji.....	35

Wstęp

Jeśli miałbym wybrać zasób, którego odpowiednie wykorzystywanie jest według mnie, ale i też wielu innych ludzi najważniejsze, wybrałbym czas. Każdy człowiek go posiada, każdy człowiek musi go używać i nim zarządzać, a jednocześnie każdy człowiek powinien zdawać sobie sprawę z tego, iż czas jest dobrem skończonym. Sekundy, minuty i godziny alokowane na jakąkolwiek czynność, nigdy już nie powrócą i nie pozwolą wykorzystać się ponownie. Oznacza to, że na każdym kroku ludzie decydują o tym na co poświęcą kolejne odcinki swojego życia. Obecnie jako ludzie mamy wiele możliwości pożytkowania czasu. Można uczyć się nowych umiejętności, można spędzać czas na relaksie czy podróżach. Niektórzy decydują się wymieniać go na inne zasoby w postaci pracy zarobkowej, a jeszcze inni najzwyczajniej w świecie go marnują. Poniższa praca omawia zagadnienia związane z efektywnym wykorzystywaniem tego zasobu i przytacza kilka prac badawczych na temat ćwiczeń w tym zakresie oraz ich skuteczności.

Jako że obecne technologie dostarczają możliwość tworzenia narzędzi mogących usprawniać nasze działania, praca przedstawia proces implementacji popularnych technik zarządzania czasem w aplikacji internetowej zbudowanej w technologiach Java oraz Angular.

W pierwszym rozdziale tejsze pracy zostały omówione najpopularniejsze techniki zarządzania czasem pod kątem teoretycznym. Jest to Technika Pomodoro służąca do usprawniania naszych działań, aby wykonywanie zadań było szybsze i bardziej wydajne. Opis techniki Pomodoro został zaczerpnięty z książki „*Pomodoro Technique*” autorstwa Francesco Cirillo. Macierz Eisenhowera, która powinna uporządkować zadania do wykonania tak aby skupiać się jedynie na najważniejszych, a przełożyć w czasie lub usunąć mniej istotne czynności jest drugim sposobem zarządzania czasem. Została ona opisana przy wsparciu pracy „*Eisenhower matrix Saaty AHP = Strong actions prioritization? Theoretical literature and lessons drawn from empirical evidences*”, której autorami są Alfred Homère Ngandam Mfoundoum, Mesmin Tchindajng, Jean Valery Mefire Mfoundoum, Isabelle Makeout. Prawo Parkinsona służące do skrócenia czasu wykonywanego zadania bez uszczerbku na jakości wykonanej pracy zostało opisane korzystając z pracy C.Northcote Parkinson „*Parkinson's law*”. Na koniec *Zasada Pareto* opisana na podstawie opracowania autorstwa Rosie Dunford, Quanrong Su, Ekraj Tamang oraz Abigail Wintour, „*The Pareto Principle*”, wspierająca decydowanie o tym, ile czasu powinniśmy poświęcać na poszczególne zadania. W pierwszym rozdziale znajduje się również omówienie skuteczności

treningów zarządzania czasem na podstawie pracy „*skuteczność treningów zarządzania czasem - przegląd badań*” autorstwa Dr Katarzyny Grunt-Mejer.

Rozdział drugi skupia się na implementacji powyższych narzędzi w prostej aplikacji internetowej. Składa się ona z części serwera, czyli backendowej, która zbudowana jest przy użyciu języka Java wraz z frameworkiem Spring oraz części użytkownika, czyli frontendowej wykorzystującej technologię Angular. Część backendowa została opracowana na podstawie dokumentacji technicznej języka Java, oraz dokumentacji technicznej frameworka Spring, natomiast część frontendowa wynika z dokumentacji technicznej frameworka Angular. Aplikacja posiada własną bazę danych oraz została zaprojektowana w taki sposób, aby korzystanie z niej było szybkie i wygodne dla użytkownika.

Ostatni rozdział pracy omawia działanie aplikacji. Zawiera szczegółowy opis funkcjonalności w niej zawartych, instrukcję użytkowania oraz przykładowy sposób wykorzystania aplikacji w celu usprawnienia swojej pracy.

1. Teoria zarządzania czasem

1.1 Wprowadzenie

Efektywne zarządzanie czasem jest w dobie dzisiejszych praktyk bardzo popularne. W dużej mierze polega na optymalnym wykorzystaniu go, szczególnie przy realizacji wybranych celów. Takim celem może być wykonanie konkretnego zadania, jak na przykład rozliczenie w pracy zawodowej czy różne długofalowe działania, z których można wymienić naukę języka, podniesienie sprawności fizycznej lub solidny odpoczynek. Lothar J. Seiwert w swojej książce *Zarządzanie czasem. Bądź Panem własnego czasu* zdefiniował je jako: „konsekwentne i zorientowane na cel stosowanie w praktyce sprawdzonych technik pracy w taki sposób, że kierowanie samym sobą i swoim otoczeniem odbywa się bez trudu, a otrzymany do dyspozycji czas jest wykorzystany sensownie i optymalnie”. Wymienione przez niego zalety dobrego korzystania z zasobu jakim jest czas to:

- realizacja tych samych zadań, przy użyciu mniejszej ilości energii
- zwiększenie poziomu organizacji własnej pracy
- zwiększenie wyników w pracy
- zmniejszenie chaosu oraz stresu
- zwiększenie poziomu zadowolenia z pracy
- zwiększenie poziomu motywacji pracownika
- zwiększenie ilości czasu na zadania „wyższego rzędu”
- zmniejszenie presji w pracy oraz nacisk na wydajność
- zmniejszenie ilości błędów popełnianych w trakcie pracy
- przyspieszenie realizacji celów

Ważnym pojęciem w kontekście zarządzania czasem jest prokrastynacja (z łac. Procrastinatio – odraczanie, odwlekanie na potem), a więc oddalanie w czasie czynności, które należy wykonać, aby osiągnąć postanowione cele. Człowiek interesujący się optymalizacją czasu powinien poznać również definicję dystraktora, czyli według serwisu SJP.pl czynnika rozpraszaającego uwagę, przeszkadzającego w skupieniu. Oba te zagadnienia pełnią istotną rolę w maksymalizacji efektywności, ponieważ są równie niepożądane i należy się ich wystrzegać.

Według L. J. Seiwertha na proces zarządzania czasem składa się pięć poszczególnych faz. Pierwszym etapem jest wyznaczanie celów. Ten etap ma za zadanie zidentyfikować działania, które należy wykonać, aby zadanie zostało uznane za zrealizowane. Następnie w kolejności należy podejść do fazy planowania, w której uprzednio zidentyfikowane zadania zostaną rozmieszczone w czasie. Kolejnym krokiem jest podejmowanie decyzji. Ta faza ma za zadanie posegregować działania pod kątem ich kolejności, niezbędności oraz rodzaju. Przedostatnim etapem jest realizacja, czyli faza, w której wcześniej zaplanowane zadania wcielane są w życie. Na koniec należy pamiętać o monitorowaniu, którego celem jest sprawdzenie poprawności wykonanych zadań oraz wszystkich parametrów z tym związanych jak czas realizacji, zaplanowane wykorzystanie zasobów czy ilość zrealizowanych zadań.

2. Popularne techniki oraz metody planowania zadań

2.1 Opis Technik zarządzania czasem

2.1.1 Technika Pomodoro

Technika Pomodoro powstała w późnych latach 80 ubiegłego wieku. Francesco Cirillo – jej autor zaczynał wtedy karierę studenta na uniwersytecie i mierzył się z problemami w nauce do egzaminów. Nie potrafił się skupić oraz zorganizować. Postanowił więc spróbować uczyć się przez krótki czas. Za cel postawił 10 minut, a jako strażnika swojego zaangażowania uznał kuchenny czasomierz w kształcie pomidora. Metoda okazała się być skuteczną. Dzięki niej ulepszył swoje podejście do nauki, a w późniejszym czasie również do pracy zawodowej. Z biegiem czasu doskonalił swoją metodę, a w końcu zaczął się nią dzielić.

Zdaniem autora stosowanie metody Pomodoro niesie ze sobą wiele zalet. W swojej pracy wymienił między innymi ograniczenie dystraktorów oraz ulepszenie swojej umiejętności do skupiania się. Zauważa również wzrost motywacji do wykonywania zadań jak i jej dłuższe utrzymywanie, a nawet podwyższenie uważności na podejmowane decyzje. Metoda ma za zadanie wzmocnić determinację do podejmowania złożonych wyzwań, ulepszać proces nauki oraz wydajność pracy zawodowej.

U podstaw techniki Pomodoro leżą trzy główne założenia, na które składa się przynoszące ulgę inne spojrzenie na czas, lepsze wykorzystywanie zasobów umysłu oraz prostota zapewniająca użytkownikowi maksymalne skupienie się na zadaniach do wykonania. Sama metoda natomiast powinna opierać się na dziennym procesie służącym do ciągłego doskonalenia swoich umiejętności efektywnego wykonywania zadań. Ważne jest planowanie na początku dnia, które powinno zdecydować jakie działania mają zostać podjęte. Równie istotne są śledzenie prac, aby zbierać informacje na temat wykonywanych zadań, zapisywanie zebranych informacji w celu tworzenia swojego rodzaju archiwum oraz wyciąganie wniosków z zebranych danych. Na sam koniec dnia pracy zawsze powinno się wizualizować zebrane informacje w sposób dostarczający głębsze zrozumienie procesu, które prowadzić ma do jego udoskonalenia.

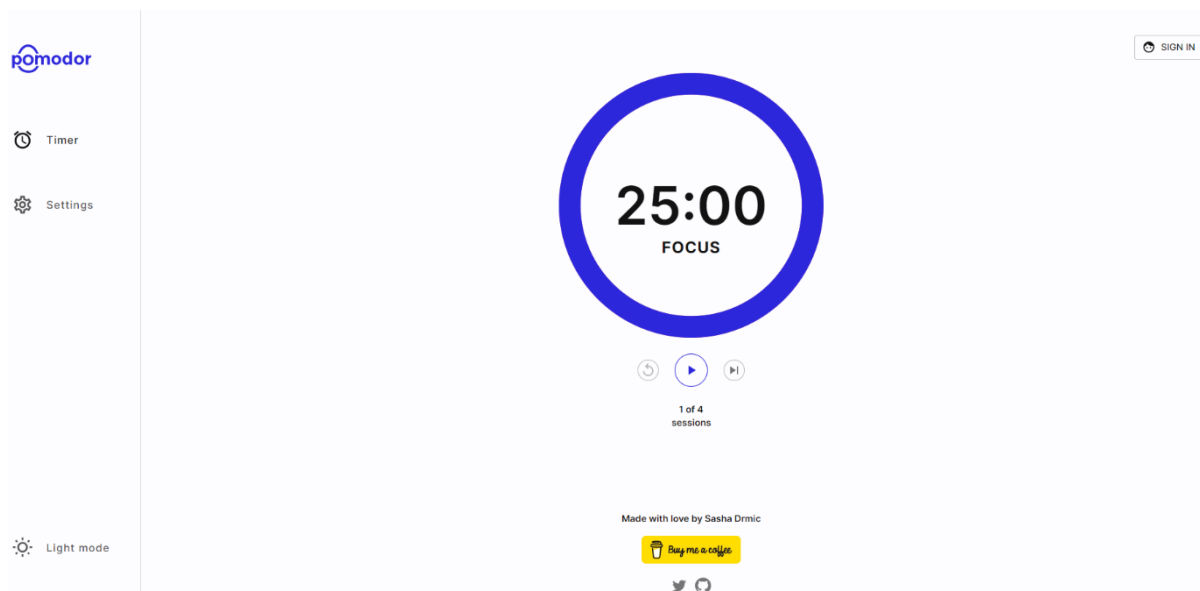
Zdaniem autora jego technika wymaga od siedmiu do dwudziestu dni konsekwentnego użytkowania, aby się jej nauczyć. Wpierw należy się jednak odpowiednio przygotować.

Adepci muszą posiadać narzędzie do pomiaru czasu, które w dzisiejszych czasach może być zarówno analogowym czasomierzem w kształcie pomidora jak i zwykłym smartfonem z zainstalowaną aplikacją. Niezbędna jest lista zadań na każdy dzień, którą należy przygotowywać o poranku. Powinna ona zawierać nagłówek z miejscem, datą jak i autorem listy, szczegółowe cele do osiągnięcia danego dnia, uporządkowane pod kątem ich priorytetowości, oraz na koniec sekcję zawierającą potencjalne niezaplanowane zjawiska mogące wpłynąć na wykonanie planu danego dnia. Poza listą zadań potrzebna jest również lista raportów z ich wykonywania. Powinna być ona wypełniana na koniec dnia i zawierać datę, opis przebiegu prac jak i liczbę cykli Pomodoro potrzebnych na wykonanie poszczególnych wyzwań.

Długość jednego cyklu Pomodoro powinna być stała, ale może się różnić ze względu na okoliczności oraz osoby ją wyznaczające. Przyjmuje się, że czas umożliwiający myślenie w skupieniu to od dwudziestu do czterdziestu pięciu minut, natomiast autor twierdzi, iż idealny cykl pracy zajmuje zazwyczaj trzydzieści minut, po których następuje krótka chwila przerwy zależna od poziomu zmęczenia danymi czynnościami. Same przerwy nie powinny skracać się przez zewnętrzną presję, gdyż będzie to negatywnie oddziaływać na efektywność wykonywanych czynności. Autor zaleca, aby odpoczynek nie trwał dłużej niż trzydzieści minut, ponieważ może to wybijać osobę pracującą z rytmu, oraz szczególnie dla osób początkujących radzi zrobić dłuższą przerwę po czterech wykonanych cyklach.

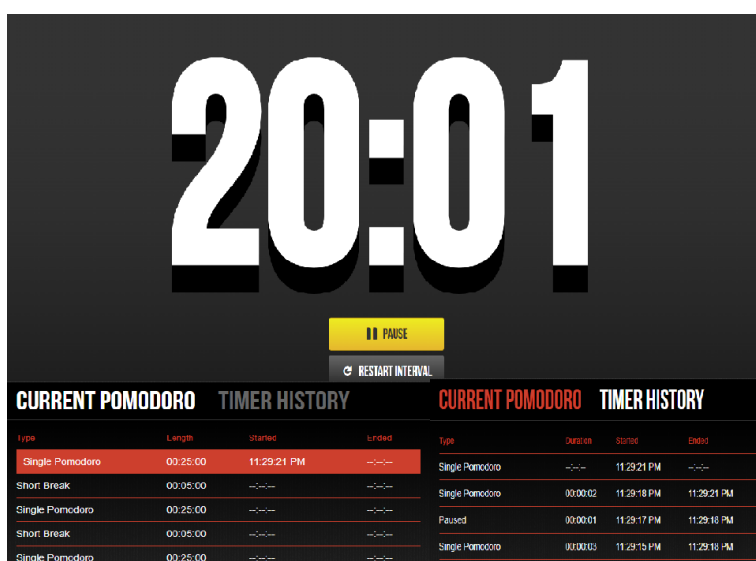
Aplikacja implementująca metodę Pomodoro powinna zawierać listę zadań służącą do spisywania wszystkich zagadnień wymagających wykonania. Oprócz tego niezbędny jest czasomierz z funkcją dostosowywania czasu pracy, czasu krótkiego odpoczynku, jak i czasu dłuższego odpoczynku. Ostatnim ważnym aspektem jest możliwość oznaczania wykonanych zadań oraz zapisywania statystyk zamkniętego zadania. Statystyki to ilość potrzebnych cykli Pomodoro na jego wykonanie oraz opcjonalne notatki.

Przykładem aplikacji implementującej metodę Pomodoro jest aplikacja internetowa pomodor.app zaprezentowana na rysunku 1. Posiada ona czytelny interfejs z czasomierzem oraz umożliwia konfigurację długości poszczególnych cykli.



Rysunek 1 Przykład zastosowania techniki Pomodoro w aplikacji internetowej; źródło: <https://pomodor.app/timer>; data odczytu: 27.07.2024

Kolejną aplikacją implementującą technikę Pomodoro jest marinaratimer. Podobnie jak poprzednia oferuje czytelny interfejs graficzny, ale nie pozwala zdefiniować długości poszczególnych cykli Pomodoro. Jako plus należy uznać możliwość sprawdzenia historii wykonywanych cykli jak i możliwość udostępnienia używanego czasomierza za pomocą odnośnika. Jej interfejs został pokazany na rysunku 2



Rysunek 2 Przykład zastosowania techniki Pomodoro w aplikacji internetowej; źródło: <https://www.marinaratimer.com/p5snj>; data odczytu: 27.07.2024

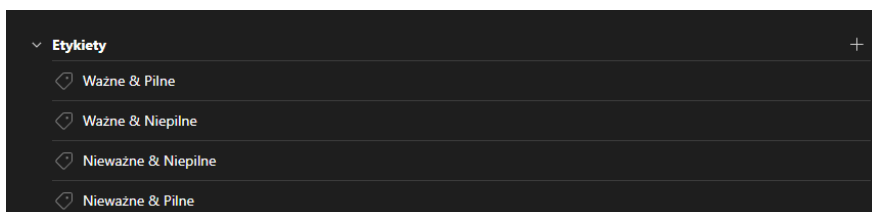
2.1.2 Macierz Eisenhowera

Macierz Eisenhowera opracował trzydziesty czwarty prezydent USA – Dwight Eisenhower. Podczas jego rządów mierzył się z politycznymi oraz ekonomicznymi problemami rangi państwowej. Musiał przez to wybierać odpowiednie zadania, którymi miał się zajmować w pierwszej kolejności. Pewnego razu stwierdził „Mam dwa typy problemów: pilne oraz ważne. Pilne nie są ważne, a ważne nigdy nie są pilne.” Od tamtej pory jego metoda dzielenia zadań do wykonania zyskiwała na popularności.

Macierz składa się z czterech obszarów. Pierwszym z nich jest ten zawierający zadania ważne oraz jednocześnie pilne. Muszą one zostać wykonane w pierwszej kolejności, najszybciej jak to możliwe. Kolejna grupa to zadania ważne, lecz nie pilne. Cechują się one tym, że powinny zostać wykonane, ale ich realizacja powinna zostać zaplanowana na czas późniejszy. Następny obszar obejmuje kwestie mało ważne, ale pilne. Oznacza to, iż nie wymagają one zaawansowanej wiedzy i mogą zostać oddelegowane do kogoś kto akurat ma na nie czas. Ostatnią grupą są zadania mało ważne i nie wymagające natychmiastowego wykonania. Powinny być one po prostu porzucone. Jeśli jakieś działanie miałoby się znaleźć w tej części macierzy, to zdecydowanie nie powinno się marnować na nie czasu.

Aplikacja implementująca funkcjonalność macierzy Eisenhowera powinna umożliwiać stworzenie listy zadań oraz dopasowanie każdego z nich do odpowiedniego sektora macierzy. Widok funkcjonalności powinien być przejrzysty, a zmiana właściwości poszczególnych zadań powinna być umożliwiona poprzez wygodne przenoszenie bloków zadań kursorem.

Przykładem aplikacji pozwalającej na skorzystanie z macierzy jest todoist.com. Umożliwia ona stworzenie etykiet, którymi będą oznaczane tworzone zadania. Użytkownik sam może ustalić nazwy oraz kolory etykiet, co sprawia, że każde nowo dodane zadanie może zostać przypisane do odpowiedniej grupy z macierzy Eisenhowera. Niestety aplikacja nie oferuje czytelnego widoku macierzy, a jedynie segregację zadań pod kątem przypisanych etykiet.



Rysunek 3 Przykład zastosowania macierzy Eisenhowera w aplikacji internetowej; źródło: <https://app.todoist.com/app/filters-labels>; data odczytu: 27.07.2024

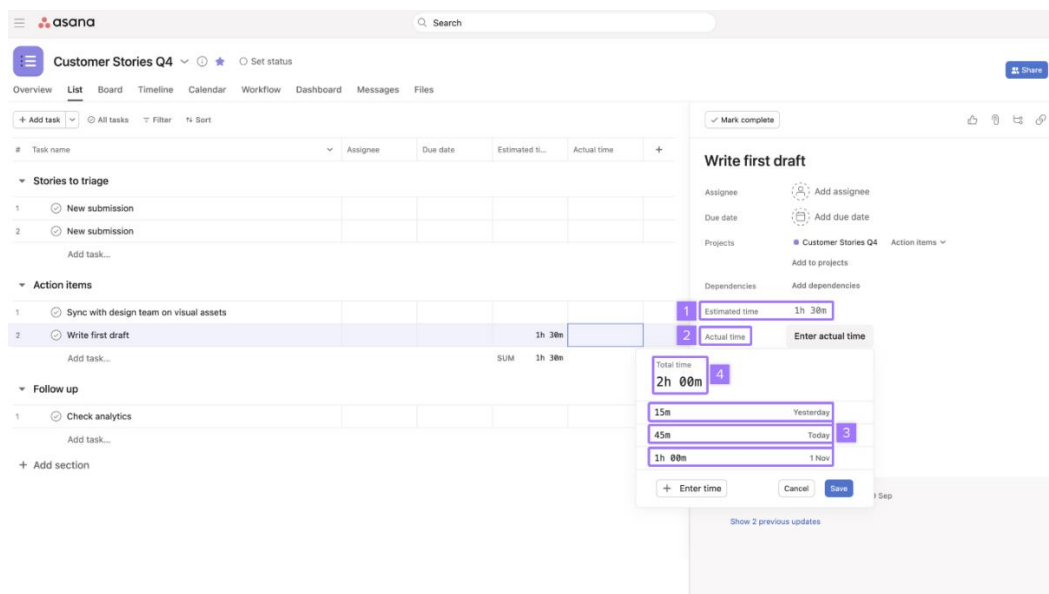
2.1.3 Prawo Parkinsona

Treść prawa Parkinsona to: „praca rozszerza się, aby wypełnić czas przeznaczony na jej wykonanie”. Autor podaje przykłady związane z rozszerzaniem kadry urzędniczej, która pomimo zwiększającej się liczby nie odnosiła większych sukcesów na polu efektywności. Sama zasada może być jednak wykorzystywana w procesie planowania zadań do wykonania. Osoba mająca na uwadze, że istnieje możliwość, aby identyczne zadanie zostało wykonane w godzinę zamiast dwóch godzin będzie efektywniej planować swoje działania.

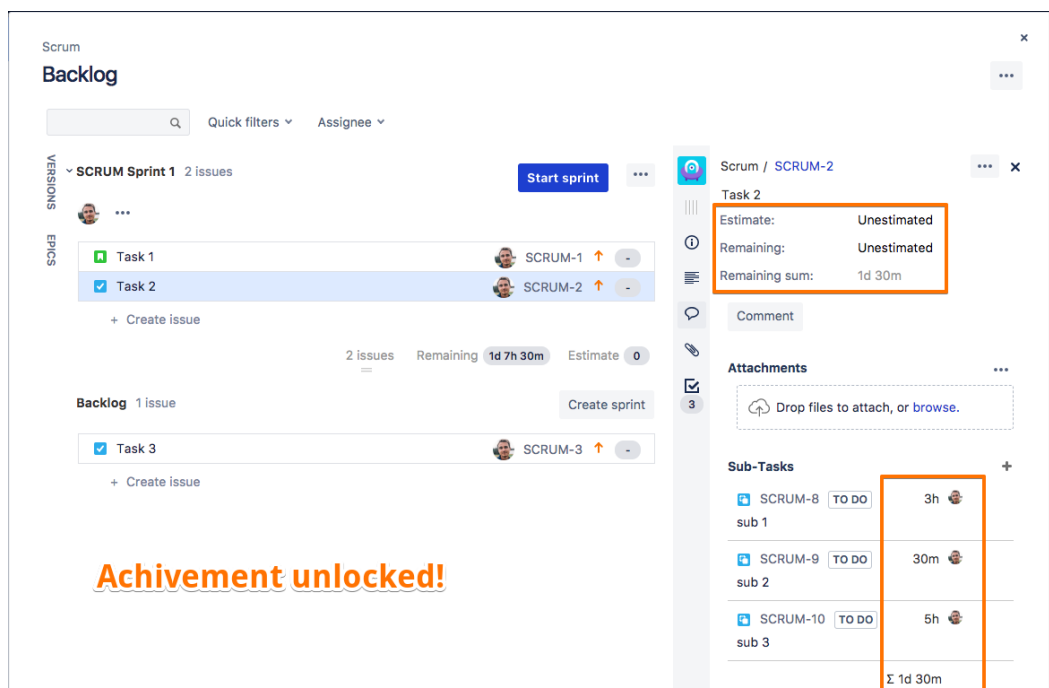
Aby przezwyciężyć działanie prawa Parkinsona należy w pierwszej kolejności stworzyć listę zadań do wykonania. Pozwoli ona skrupulatnie określić co powinno zostać zrobione, kogo należy zaangażować do pomocy oraz jakie są priorytety. Ostatnim krokiem jest dodanie terminu wykonania do każdego zadania z listy. Termin jest niezwykle ważny, ponieważ właśnie jego brak sprawi, iż realizacja będzie rozciągać się w czasie.

Aplikacja implementująca prawo Parkinsona powinna wspomagać proces tworzenia listy zadań. Każde zadanie na liście ma przewidywany czas poświęcony na jego wykonanie. Użytkownik musi mieć świadomość o istnieniu tendencji do rozszerzania pracy na cały dostępny czas, a więc podczas próby wprowadzenia każdego zadania musi otrzymywać subtelny informację przypominającą o tym, iż powinien się upewnić czy wprowadzony czas, aby na pewno nie jest zbyt długi.

Przykładem aplikacji, w której można użyć prawa Parkinsona jest każdy program umożliwiający przypisanie czasu na wykonanie zaplanowanego zadania. Wymienić można rozbudowane systemy jak Asana lub Jira, w których podczas tworzenia zadań istnieje możliwość przypisania do nich estymowanego czasu na ich wykonanie.



Rysunek 4 Przykład zastosowania prawa Parkinsona w aplikacji Asana; źródło: <https://help.asana.com/hc/en-us/articles/14101461379867-Time-Tracking>; data odczytu: 20.08.2024



Rysunek 5 Przykład zastosowania prawa Parkinsona w aplikacji Jira; źródło: <https://community.atlassian.com/t5/Jira-questions/Is-it-still-possible-to-add-a-estimation-field-to-a-subtask/qaq-p/1397610>; data odczytu: 20.08.2024

2.1.4 Zasada Pareto

Zasada Pareto mówi o tym, iż 80% efektów jest produkowana poprzez 20% nakładów pracy. Zjawisko to zostało pierwszy raz zauważone przez Włoskiego ekonomistę i socjologa Vilfredo Pareto. Jego obserwacja dotyczyła tego, że jedynie 20% populacji posiadało 80% nieruchomości we Włoszech. Na jej podstawie stworzył później model matematyczny opisujący dysproporcję między nakładami, a efektami. Utworzone prawo znajduje zastosowanie nie tylko w budownictwie, ale także w zarządzaniu oraz biznesie. Na przestrzeni lat okazało się, że zazwyczaj większość skutków bierze się z mniejszości wykonywanej pracy. Jest to więc dobre narzędzie do tego, aby wykonywane zadania były jak najbardziej wydajne pod kątem efektów oraz poświęconego czasu.

Model matematyczny zasady Pareto można przedstawić w postaci funkcji. Jeśli X jest zmienną losową reprezentującą pewną cechę (np. dochód, zysk), a x to konkretna wartość tej cechy, to możemy obliczyć prawdopodobieństwo, że zmienna losowa X przyjmie wartość mniejszą lub równą x , czyli $F(x)$. W przypadku rozkładu Pareto, funkcja dystrybucyjna $F(x)$ wygląda następująco:

$$F(x) = 1 - \left(\frac{x_m}{x}\right)^k$$

gdzie:

- x to konkretna wartość, dla której obliczamy dystrybucję
- x_m to minimalna wartość, jaką może przyjąć zmienna X (minimalna wartość Pareto)
- k to parametr kształtu charakteryzujący rozkład Pareto.

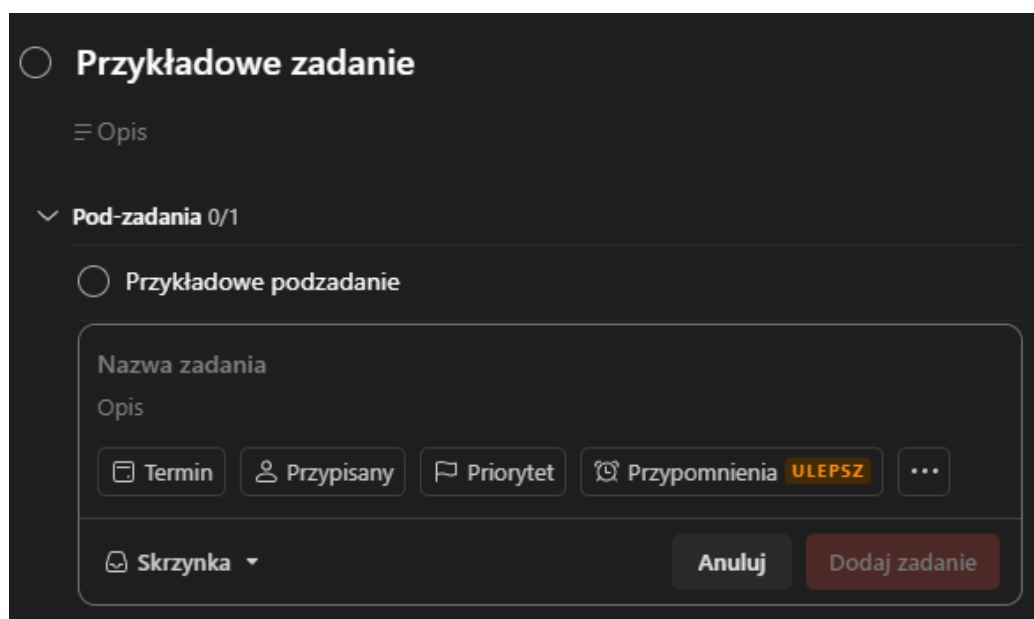
Korzystając z tego wzoru, można obliczyć, ile procent danej cechy przypada na określoną wartość lub przedział wartości.

Opracowanie R. Dunford przywołuje przykład najbogatszych ludzi z listy Forbes. Na podstawie 1226 najbardziej zamożnych ludzi zostało wyliczone, iż 20% najbardziej majątnych posiada 56,72% kapitału członków całej listy. Jest to mniej niż 80% jednak zasada Pareto została zachowana, ponieważ znaczna część nakładów, a w tym przypadku majątku, pochodzi od mniejszości członków badanej grupy. Drugi przykład opisuje listę państw wraz z ich produktem krajowym brutto z 2011 roku. Badacze podają, że 20% najbogatszych państw posiada 91,62% wszystkich pieniędzy na świecie. Wnioskiem wynikającym z badania

realnych danych jest, że pomimo iż relacja nie kreśli się dokładnie na poziomie 80:20 to jednak widać założone przez zasadę Pareto proporcje nakładów do efektów.

Aplikacja do zarządzania czasem wcielająca w życie zasadę Pareto powinna usprawniać proces tworzenia listy zadań na dwa sposoby. Pierwszym jest przypominanie o skupianiu się jedynie na najważniejszych aspektach danego zadania. Drugim natomiast jest umożliwienie użytkownikowi stworzenia listy zadań, która będzie zawierała podzadania. Po uzupełnieniu listy czynności niezbędnych do wykonania danego zadania będzie można łatwo określić priorytetowe 20% działań i odpowiednio nimi zarządzić, a pozostałe 80% oddelegować lub odłożyć ich wykonanie w czasie.

Przykładowa aplikacja umożliwiająca zastosowanie zasady Pareto to wspomniana już wcześniej todoist.com. Pozwala ona na tworzenie listy zadań wraz z podzadaniami. Każdemu podzadaniu można przypisać odpowiedni priorytet co pozwala na zaznaczenie najważniejszych 20% zadań do wykonania.

The image shows a dark-themed user interface for creating a task in the Todoist application. At the top, there is a title "Przykładowe zadanie" (Example task) with a toggle switch. Below it is a section for "Pod-zadania 0/1" (Sub-tasks 0/1). The main form area contains a "Przykładowe podzadanie" (Example sub-task) section. This section has input fields for "Nazwa zadania" (Task name) and "Opis" (Description). Below these are several interactive buttons: "Termin" (Due date), "Przypisany" (Assigned to), "Priorytet" (Priority), "Przypomnienia" (Reminders), and a button labeled "ULEPSZ" (Improve) in orange. At the bottom of the form, there is a "Skrzynka" (Inbox) dropdown menu, an "Anuluj" (Cancel) button, and a "Dodaj zadanie" (Add task) button in a dark red color.

Rysunek 6 Przykład zastosowania zasady Pareto w aplikacji internetowej; źródło: <https://app.todoist.com/app/inbox>; data odczytu: 20.08.2024

2.2 Skuteczność treningów zarządzania czasem

Rozpatrując zagadnienia mające poprawić efektywność zarządzania czasem należy również zwrócić uwagę na poziom skuteczności treningów związanych z tą dziedziną rozwoju osobistego. Obecnie można znaleźć duże ilości badań naukowych dotyczących tego obszaru, a nawet opracowania wyciągające wnioski z poszczególnych badań. Jest to niezwykle pomocne, ponieważ temat zarządzania czasem staje się coraz bardziej popularny

zarówno wśród firm jak i osób fizycznych. Logicznym podejściem jest więc uprzednia próba sprawdzenia czy teoretyczny trening bądź szkolenie ma szanse odnieść jakiegokolwiek pozytywne efekty po jego przeprowadzeniu. Postawa ta może oszczędzić potencjalnie wydane pieniądze, pomóc zdecydować się na podjęcie konkretnych kroków oraz w przypadku, jeśli szkolenia zarządzania czasem nie przynoszą pozytywnych efektów, oszczędzić czas nie uczęszczając na bezsensowne treningi.

Autorka opracowania „*skuteczność treningów zarządzania czasem – przegląd badań*” wskazuje na pozytywne efekty w dziedzinach nie związanych bezpośrednio z czasem pracy. Mówi o pozytywnym oddziaływaniu szkoleń na dobrostan psychiczny uczestników, mieli oni poczucie większej kontroli nad swoimi wyborami, czuli się bardziej skuteczni podczas codziennych działań, ale również wzrosła ich efektywność i zwiększyła się częstotliwość wypełniania postawionych sobie codziennych obowiązków. Uczestnikami zazwyczaj byli zarówno studenci jak i pracownicy zawodowi. Jedno z przytoczonych badań przeprowadzone przez Fritz, Fu Lam i Spreitzera, w 2011 roku na 200 osobach przebywających w warunkach zawodowych wykazało, że umiejętności takie jak unikanie dystraktorów czy robienie przerw o odmiennym charakterze niż bieżąca praca, są skuteczne w poprawie stanu vitalności uczestnika jak i wzrostu chęci do wykonania danej czynności.

Kolejnym aspektem poruszonym przez dr Katarzynę Grunt-Mejer jest faktyczne oddziaływanie treningów na ich uczestników. Zostały wyróżnione dwa obszary oddziaływań. Pierwszym jest efekt behawioralny szkoleń, a więc mówiący o skuteczności działania, natomiast drugim jest efekt psychologiczny, czyli ten, który odpowiada za polepszenie dobrostanu psychicznego.

Pozytywny wpływ pod kątem behawioralnym został zauważony w większości przytaczanych badań. Choć niektóre z rozpatrywanych opracowań posiadały braki w postaci zbyt małej lub zbyt mało reprezentatywnej grupy badanych, to inne starały się symulować odpowiednie warunki dla testów. Analiza wykonana przez Greena i Skinnera w 2005 roku została przeprowadzona na 233 uczestnikach i obejmowała 19 treningów od 4 do 22 osób na szkolenie. Treningi te były organizowane w różnych instytucjach co prowadzić miało do zwiększenia liczby reprezentowanych przez uczestników zawodów. Podstawą teoretyczną szkoleń były zasady zarządzania czasem IV generacji, zaprezentowane przez Coveya, Merrilla i Merrill z 1994 roku. Efektywność została zbadana od kilku tygodni do kilku miesięcy po ukończeniu treningu. Większość uczestników zgłosiła poprawę o 20% w

kluczowych aspektach jak planowaniu, priorytetyzacji czy asertywności. Niektóre badania napotykały problemy w postaci występujących zwolnień pracowników. Zwolnienia te mogły wpływać na efektywność pozostałych pracujących, którzy chcieli uniknąć losu byłych współuczestników. Kolejne problemy pojawiały się w przypadku zbyt dużego natężenia pracy wśród badanych. Treningi przyniosły wtedy najmniejszy efekt, gdyż pracownicy byli przeładowani obowiązkami. Można więc stwierdzić, iż w obszarze behawioralnym treningi zarządzania czasem odnoszą pozytywne skutki. Pracownicy są wydajniejsi oraz zauważają u siebie mniej sytuacji związanych z prokrastynacją. Ćwiczenia muszą jednak być przeprowadzone prawidłowo, a otoczenie uczestników szkoleń musi sprzyjać zdobywaniu nowej wiedzy oraz zapewniać poczucie bezpieczeństwa w miejscu pracy.

Obszar efektów psychologicznych zdaje się być jednoznaczny. Badanie przeprowadzone przez Häfner i Stock w 2010 roku na grupie 71 pracowników podzieliło uczestników na grupę kontrolną oraz grupę, która odbyła jednodniowy trening. Zostały odnotowane pozytywne skutki w grupie treningowej. Pracownicy byli mniej zestresowani oraz postrzegali własną kontrolę nad czasem jako większą. Większa część badanych korzystała również z poznanych na treningu technik co najmniej raz w okresie 6 tygodni od ukończenia treningu. Rozpatrywane badania wskazują więc, że nawet krótki jednodniowy trening zarządzania czasem zapewnia pracownikom poprawę samopoczucia i zmniejsza stres, z którym muszą się mierzyć. Uczestnicy szkoleń mieli przeświadczenie, że po odbyciu treningu posiadają większą kontrolę nad swoim czasem, nawet w przypadku, kiedy statystyki wykonywanych zadań nie wskazywały na jakąkolwiek poprawę pod kątem wydajności. Zdaje się to być jednak efekt krótkotrwały, ponieważ badania wykazały, iż w przypadku braku efektów behawioralnych, choć efekt psychologiczny występował, to po 4 do 5 miesiącach nie było po nim śladu i badana grupa uczestników nie różniła się od grupy kontrolnej.

Podsumowując treningi zarządzania czasem wydają się mieć raczej pozytywny wpływ na uczestników. Rozpatrywane przez dr Katarzynę Grunt-Mejer badania sugerują, że w znacznej mierze pozytywne efekty zarówno behawioralne jak i psychologiczne były widoczne. Przeprowadzanie szkoleń pod tym kątem nie jest więc stratą pieniędzy oraz czasu, a może realnie wpłynąć na samopoczucie oraz efektywność pracowników. Należy jednak pamiętać o tym, iż żadne szkolenie nie zapewni pozytywnych skutków w przypadku, kiedy pracownicy muszą się mierzyć z nadmiernym obciążeniem obowiązkami lub stresem w miejscu pracy.

3. Specyfikacja techniczna wykonanej aplikacji

3.1 System kontroli wersji

Podczas pracy nad aplikacją niezbędny jest system kontroli wersji. Służy on do pracy nad kodem w bezpieczny sposób. Gwarantuje również przejrzystość historii zmian wprowadzanych do projektu oraz możliwość pracy kilku programistów jednocześnie. Kolejną korzyścią wynikającą z używania systemu kontroli wersji jest proste udostępnianie kodu źródłowego aplikacji osobom postronnym przy jednoczesnym zarządzaniu uprawnieniami sprawiającymi, iż zmiany w kodzie będą wykonywane jedynie przez osoby upoważnione.

Projekt time-planner jest dostępny w repozytorium na platformie GitHub pod adresem: <https://github.com/dominikbrzek/time-planner>. Zawiera on zarówno część serwera jak i część użytkownika wraz z wymaganą dokumentacją. Podczas pracy nad kodem aplikacji został przyjęty szereg zasad. Kod pisany jest w języku angielskim. Dokumentacja pisana jest w języku polskim. Komentarze do kodu są zbędne, ponieważ sam kod jest self-describing, a więc nie potrzebuje dodatkowych wyjaśnień.

3.2 Wymagania biznesowe

Aplikacja umożliwia korzystanie z wcześniej opisanych technik zarządzania czasem. Zawiera zatem:

Panel z czasomierzem wykorzystywanym w technice Pomodoro. Użytkownik może skonfigurować ilość oraz długości cykli, uruchomić bądź zatrzymać licznik czasu jak i zresetować mechanizm do pozycji początkowej.

Panel z listą umożliwiający wprowadzanie nowych zadań, modyfikowanie, usuwanie, przypisywanie priorytetów oraz estymowanie potrzebnego na ich wykonanie czasu. Podczas dodawania zadań wyświetlane są przypomnienia na temat prawa Parkinsona oraz zasady Pareto. Lista umożliwia również dodawanie podzadań.

Panel z macierzą Eisenhowera, która pozwala na wyświetlenie zadań z listy wraz z przypisanymi im priorytetami. Sama macierz umożliwia zmienianie priorytetów zadań poprzez przenoszenie kafelków za pomocą myszki komputerowej.

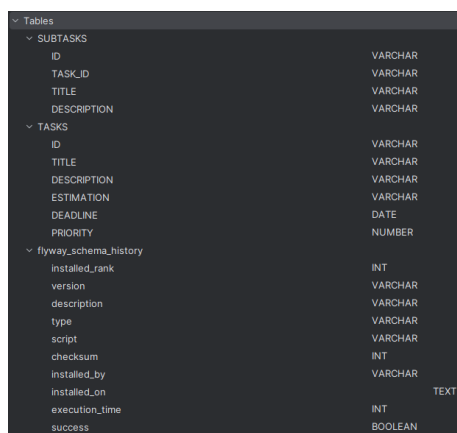
3.3 Technologia wykorzystana po stronie serwera (Backend)

3.3.1 Opis stosu technologicznego

Jako główny język programowania została użyta Java w wersji 22, a więc najnowszej na moment tworzenia opracowania. Jako framework służący do tworzenia aplikacji webowych został użyty Spring w wersji 6.1.8 oraz Spring Boot w wersji 3.3.0. Zastosowana baza danych to SQLite. Jest to baza relacyjna i pozwala na łatwe tworzenie prostych konstrukcji bazodanowych. Jej dodatkowym atutem jest kompaktowy rozmiar oraz brak potrzeby uruchamiania dodatkowego serwisu na serwerze. Wersjonowanie bazy danych zostało wsparte biblioteką Flyway w wersji 10.15.2, która w prosty sposób umożliwia odbudowanie schematu tabel. Sam kontakt z bazą danych został zaimplementowany przy użyciu Spring data JPA w wersji 3.3.0 oraz Hibernate w wersji 6.5.2. Pozwoliło to na szybkie stworzenie prostych zapytań bazodanowych, które były niezbędne do działania aplikacji.

3.3.2 Opis modelu bazodanowego

Jako że w aplikacji została zastosowana relacyjna baza danych, należało utworzyć potrzebne tabele wraz z relacjami. Nazwy tabel reprezentują zawarte w nich dane, a więc jest to tabela TASKS posiadająca w swoim obrębie informacje na temat zadań oraz tabela SUBTASKS zawierająca podzadania. Obie tabele posiadają klucze główne, a więc unikalne identyfikatory zawarte w kolumnach, które pozwalają na jednoznaczne zdefiniowanie wiersza w strukturze, dodatkowo tabela SUBTASKS posiada również klucz obcy z tabeli TASKS, reprezentujący relację jeden do wielu pomiędzy zadaniami oraz podzadaniami. Dodatkowa tabela flyway_schema_history jest utworzona poprzez bibliotekę flyway i zawiera wartości konfiguracyjne. Kolumny oraz ich typy pokazane na rysunku 7.

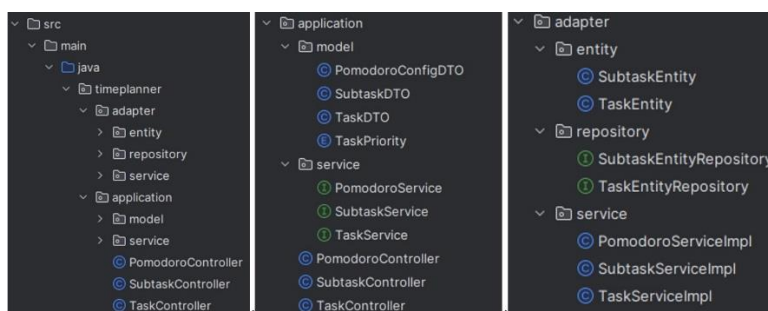


Tables	
▼ SUBTASKS	
ID	VARCHAR
TASK_ID	VARCHAR
TITLE	VARCHAR
DESCRIPTION	VARCHAR
▼ TASKS	
ID	VARCHAR
TITLE	VARCHAR
DESCRIPTION	VARCHAR
ESTIMATION	VARCHAR
DEADLINE	DATE
PRIORITY	NUMBER
▼ flyway_schema_history	
installed_rank	INT
version	VARCHAR
description	VARCHAR
type	VARCHAR
script	VARCHAR
checksum	INT
installed_by	VARCHAR
installed_on	TEXT
execution_time	INT
success	BOOLEAN

Rysunek 7 Przedstawienie nazw oraz typów kolumn w bazie danych aplikacji time-planer; źródło: zrzut ekranu z aplikacji IntelliJ Idea

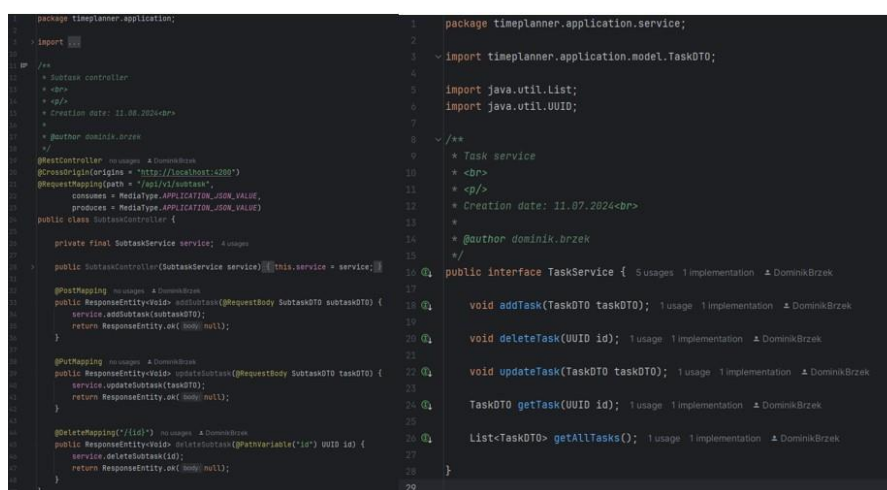
3.3.3 Architektura aplikacji

Nierozłączną częścią każdej aplikacji jest jej architektura, a więc układ plików czy też opis logicznego grupowania funkcji, które zarządzają obiektami. W projekcie została użyta architektura heksagonalna. Charakteryzuje się ona podziałem aplikacji na warstwy adaptera i portu. Sprzyja to oddzieleniu logiki biznesowej od technicznych szczegółów implementacyjnych. Podział na warstwę portu (reprezentowaną przez katalog application) oraz na warstwę adaptera (reprezentowaną przez katalog adapter) dobrze widać na zrzutach ekranu zawierających drzewo projektu.



Rysunek 8 Przedstawienie drzewa katalogów projektu aplikacji time-planer; źródło: zrzut ekranu z aplikacji IntelliJ Idea

Warstwa portu zawiera logikę biznesową aplikacji. Znajdują się w niej klasy służące do kontaktu aplikacji ze światem zewnętrznym nazywane Controllerami. Zawierają one definicje metod udostępniających zasoby serwera oraz serwisy niezbędne do kontaktu z warstwą adaptera. Serwisy te są interfejsami, których jedynym zadaniem jest zdefiniowanie metod, które są z kolei wykorzystywane w logice biznesowej oraz powinny zostać zaimplementowane w części adaptera. Szczegóły te pokazano na rysunku 9.



Rysunek 9 Przedstawienie plików zawartych w części portu w aplikacji time-planer; źródło: zrzut ekranu z aplikacji IntelliJ Idea

Część adaptera zawiera połączenie do bazy danych oraz wszystkie inne logiczne procesy i klasy niezbędne do działania aplikacji. Projekt zawiera 3 rodzaje takich klas, pierwszą z nich są klasy Entity reprezentujące strukturę bazy danych. Są one nieodłączną częścią pracy z biblioteką Spring i zawierają wartości zawarte w kolumnach odwzorowywanych przez nie tabel. Taka struktura pokazana została na rysunku 10.

```
package timeplanner.adapter.entity;

import ...

/**
 * Subtask entity
 * <br>
 * <p/>
 * Creation date: 11.08.2024<br>
 *
 * @author dominik.brzek
 */
@Entity
@Table(name = "SUBTASKS")
public class SubtaskEntity {

    @Id
    @Column(name = "ID")
    @DbTypeCode(Types.VARCHAR)
    @GeneratedValue(generator = "UUID")
    @GenericGenerator(name = "UUID", strategy = "org.hibernate.id.UUIDGenerator")
    private UUID id;

    @ManyToOne
    @JoinColumn(name = "TASK_ID")
    private TaskEntity task;

    @Column(name = "TITLE")
    private String title;

    @Column(name = "DESCRIPTION")
    private String description;
}
```

Rysunek 10 Przedstawienie pliku zawierającego klasę SubtaskEntity w aplikacji time-planer; źródło: zrzut ekranu z aplikacji IntelliJ Idea

Kolejnym typem struktur w warstwie adaptera są interfejsy repository zawierające zbiór metod niezbędnych do kontaktu z bazą danych. Dzięki rozszerzeniu interfejsu JpaRepository z biblioteki spring-data-jpa programista może pominąć żmudne definiowanie standardowych metod. Implementacja interfejsów również może zostać pominięta dzięki bibliotece Hibernate, która między innymi samoistnie generuje wymagane serwisy z odpowiednimi zapytaniami bazodanowymi.

```
1 package timeplanner.adapter.repository;
2
3 import ...
4
5 /**
6  * Subtask entity repository
7  * <br>
8  * <p/>
9  * Creation date: 11.08.2024<br>
10 *
11 * @author dominik.brzek
12 */
13 public interface SubtaskEntityRepository extends JpaRepository<SubtaskEntity, UUID> {
14 }
15
16
17
18
```

Rysunek 11 Przedstawienie pliku zawierającego interfejs SubtaskEntityRepository w aplikacji time-planer; źródło: zrzut ekranu z aplikacji IntelliJ Idea

Ostatnim wykorzystanym typem struktur w omawianej warstwie są klasy Service. Implementują one wcześniej wspomniane interfejsy z warstwy portu. Oznacza to, że zawierają logikę niezbędną do otrzymania wymaganych przez część biznesową danych. Korzystają z interfejsów Repository, klas Entity oraz z innych klas pomocniczych.

```
15  /**
16   * Task service implementation
17   * <br>
18   * <p/>
19   * Creation date: 11.07.2024<br>
20   *
21   * @author dominik.brzek
22   */
23  @Service no usages 1 DominikBrzek
24  public class TaskServiceImpl implements TaskService {
25
26      private final TaskEntityRepository repository; 7 usages
27
28      public TaskServiceImpl(TaskEntityRepository repository) { no usages 1 DominikBrzek
29          this.repository = repository;
30      }
31
32      @Override 1 usage 1 DominikBrzek
33      public void addTask(TaskDTO taskDTO) {
34          TaskEntity entity = new TaskEntity();
35          updateEntity(entity, taskDTO);
36          repository.save(entity);
37      }
38
39      @Override 1 usage 1 DominikBrzek
40      public void deleteTask(UUID id) {
41          repository.deleteById(id);
42      }
43
44      @Override 1 usage 1 DominikBrzek
45      public void updateTask(TaskDTO taskDTO) {
46          TaskEntity entity = repository.findById(taskDTO.getId()).orElseThrow();
47          updateEntity(entity, taskDTO);
48          repository.save(entity);
49      }
50
51      @Override 1 usage 1 DominikBrzek
52      public TaskDTO getTask(UUID id) {
53          return mapFromEntity(repository.getReferenceById(id));
54      }
55  }
```

Rysunek 12 Przedstawienie pliku zawierającego klasę TaskServiceImpl w aplikacji time-planer; źródło: zrzut ekranu z aplikacji IntelliJ Idea

3.3.4 Opis usług dostępnych w aplikacji

Do poprawnego działania aplikacji serwerowej niezbędny jest sposób, którym będzie można się z nią komunikować jako klient. Służą do tego wspomniane już wcześniej struktury Controller zawarte w warstwie portu. Tworzą one REST API czyli aplikacyjny interfejs programistyczny, który udostępnia zasoby serwera za pomocą zdefiniowanych metod REST. W przedstawionej aplikacji są to: GET służąca do pobierania zasobów z serwera, POST umożliwiająca dodawanie nowych danych, PUT dająca możliwość modyfikowania już istniejących danych oraz DELETE, która jak sama nazwa wskazuje wykorzystywana jest do usuwania istniejących zasobów.

Opis metod udostępnianych przez aplikację serwerową został wygenerowany przy pomocy biblioteki Swagger i prezentuje się następująco:

task-controller		
GET	/api/v1/task	Pobierz wszystkie zadania
PUT	/api/v1/task	Zmodyfikuj istniejące zadanie
POST	/api/v1/task	Dodaj nowe zadanie
GET	/api/v1/task/{id}	Pobierz dane konkretnego zadania
DELETE	/api/v1/task/{id}	Usuń istniejące zadanie
GET	/api/v1/task/priorities	Pobierz priorytety dostępne dla zadań
subtask-controller		
PUT	/api/v1/subtask	Zmodyfikuj istniejące podzadanie
POST	/api/v1/subtask	Dodaj nowe podzadanie
DELETE	/api/v1/subtask/{id}	Usuń istniejące podzadanie
pomodoro-controller		
GET	/api/v1/pomodoro	Pobierz ustawienia czasomierza Pomodoro
PUT	/api/v1/pomodoro	Zmodyfikuj ustawienia czasomierza Pomodoro

Rysunek 13 Przedstawienie usług dostępnych w aplikacji time-planer; źródło: strona wygenerowana przez bibliotekę swagger dostępna po uruchomieniu aplikacji pod adresem <http://localhost:8080/swagger-ui/index.html#/>

3.4 Technologia wykorzystana po stronie przeglądarki (Frontend)

3.4.1 Opis stosu technologicznego

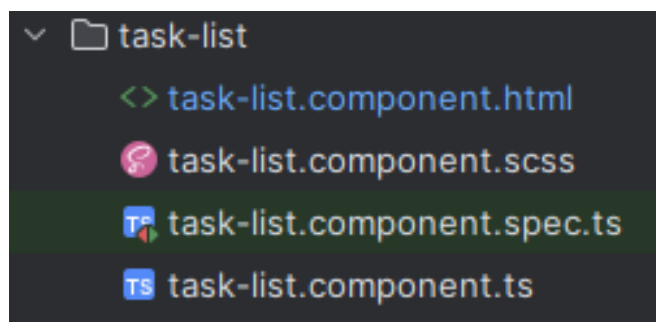
Podczas tworzenia aplikacji przeglądarkowej został wykorzystany zbiór technologii webowych oraz bibliotek. Podstawowymi językami programowania są trzy technologie. Pierwszą jest TypeScript, w wersji 5.4.2, który odpowiada za skrypty niezbędne do działania aplikacji. Drugą jest HTML tworzący strukturę strony, a trzecią CSS, który z kolei odpowiada za wygląd znaczników na stronie. Pozostałe technologie to Angular w wersji 17.3, a więc główna biblioteka wspierająca tworzenie aplikacji przeglądarkowej. Służy do generowania komponentów aplikacji, zarządza nimi oraz wspiera sam proces tworzenia oprogramowania. Kolejnymi, choć znacznie mniejszymi bibliotekami wspierającymi są rxjs w wersji 7.8.0, tslib w wersji 2.3.0 oraz zone.js w wersji 0.14.3. Ważną biblioteką jest bootstrap w wersji 4.3.1, która odpowiada za gotowe do wykorzystania komponenty

wizualne. Przyspiesza ona pracę w aplikacjach, które nie wymagają skomplikowanych struktur artystycznych w interfejsie użytkownika.

3.4.2 Architektura aplikacji

Aplikacja internetowa napisana przy użyciu biblioteki Angular składa się z komponentów oraz serwisów. Każdy komponent odpowiada za określoną część interfejsu użytkownika bądź posiada określone zadanie w procesie działania strony, natomiast serwisy są odpowiedzialne za logikę oraz komunikację z serwerem. Komponenty generowane są przy pomocy konsoli AngularCli, która dostarcza narzędzia dla developerów ułatwiające pracę nad projektem. Komponenty utworzone w aplikacji time-planer są reprezentowane przez strukturę katalogów w folderze app. Katalog assets zawiera obrazy, zdjęcia oraz grafiki wykorzystywane na stronie, natomiast pozostałe pliki są odpowiedzialne za konfigurację i ustawienia niezbędne do prawidłowej pracy aplikacji napisanej przy pomocy biblioteki Angular.

Każdy komponent składa się z czterech plików. Podstawowym jest plik HTML, który określa jakie znaczniki mają istnieć w ramach struktury komponentu.



Rysunek 14 Przedstawienie plików składowych komponentu task-list w aplikacji time-planer; źródło: zrzut ekranu z aplikacji webstorm


```

1 import sys
2
3 @component(Show usages & DominiBzsek
4   detector: 'app-task-list',
5   templateUrl: './task-list.component.html',
6   standalone: true,
7   imports: [NgForOf, NgIf, FormsModule],
8   styleUrls: ['./task-list.component.scss']
9 )
10
11 export class TaskListComponent implements OnInit {
12
13   tasks: Task[] = [];
14   priorities: string[] = [];
15   taskToAdd: Task = new Task();
16   subtaskToAdd: SubTask = new SubTask();
17
18   constructor(private taskService: TaskService) { no usages & DominiBzsek
19   }
20
21   ngOnInit(): void { no usages & DominiBzsek
22     this.getTasks();
23     this.getPriorities();
24   }
25
26   deleteTask(id: string): void { Show usages & DominiBzsek
27     this.taskService.deleteTask(id).subscribe(observerOf<void> { complete: console.info });
28     this.tasks = this.tasks.filter(task => task.id !== id);
29   }
30
31   toggleEdit(task: Task): void { Show usages & DominiBzsek
32     task.isEditing = !task.isEditing;
33   }
34
35   toggleSubtaskEdit(subtask: SubTask, taskId: string): void { Show usages & DominiBzsek
36     if (subtask.isEditing) {
37       this.subtaskToAdd = new SubTask();
38     } else {
39       this.subtaskToAdd.taskId = taskId;
40     }
41     subtask.isEditing = !subtask.isEditing;
42   }
43 }
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

25

4. Omówienie aplikacji oraz wykorzystanie opisanych technik zarządzania czasem

4.1 Opis funkcjonalności dostępnych dla użytkownika

4.1.1 Dodawanie zadań oraz podzadań

Aplikacja udostępnia funkcjonalność listy zadań. Każdy wiersz prezentuje Tytuł, opis, termin wykonania, priorytet, estymację oraz akcje, które użytkownik może wykonać na istniejącym zadaniu. Pierwsza z dostępnych czynności to edycja zadania, następnie jest usunięcie wiersza tabeli, a ostatnią z opcji jest dodanie podzadania. Wygląd listy został zaprezentowany na rysunku 16.

Lista zadań

Dodaj zadanie					
Tytuł	Opis	Termin wykonania	Priorytet	Estymacja	Akcje
Zadanie 1	Opis	24 August 2024, 21:52	Ważne i pilne	2h	<button>Edytuj</button> <button>Usuń</button> <button>Dodaj podzadanie</button>
Zadanie 2	Opis	25 August 2024, 21:54	Nieważne i niepilne	1h	<button>Edytuj</button> <button>Usuń</button> <button>Dodaj podzadanie</button>
Zadanie 4	Opis	26 August 2024, 21:54	Nieważne i niepilne	30m	<button>Edytuj</button> <button>Usuń</button> <button>Dodaj podzadanie</button>

Rysunek 16 Wygląd listy zadań aplikacji time-planer; źródło: zrzut ekranu z aplikacji time-planer

Aby dodać nowe zadanie, użytkownik musi użyć przycisku oznaczonego napisem: „Dodaj zadanie”. Wyświetla się wtedy seria znaczników typu input, w które można wpisać odpowiednie wartości. Po użyciu przycisku oznaczonego napisem: „Zapisz”, aplikacja doda podane przez użytkownika dane do bazy danych. Istnieje również możliwość anulowania akcji dodawania. Dodawanie nowego zadania zostało przedstawione na rysunku 17.

Lista zadań

Dodaj zadanie					
Tytuł	Opis	Termin wykonania	Priorytet	Estymacja	Akcje
<input type="text" value="Title"/>	<input type="text" value="Description"/>	<input type="text" value="mm/dd/yyyy --:-- --"/>	<input type="text" value=""/>	<input type="text" value="Estimation"/>	<button>Zapisz</button> <button>Anuluj</button>
Zadanie 1	Opis	24 August 2024, 21:52	Ważne i pilne	2h	<button>Edytuj</button> <button>Usuń</button> <button>Dodaj podzadanie</button>

Rysunek 17 Wygląd formularza dodającego nowe zadanie; źródło: zrzut ekranu z aplikacji time-planer

Dodawanie podzadania również zostało zrealizowane jako formularz pojawiający się po naciśnięciu przycisku – w tym przypadku należy użyć tego z napisem „Dodaj podzadanie” w wierszu zadania, które chcemy wzbogacić o podpunkt. Użytkownik może podać Tytuł oraz opis podzadania, anulować akcję dodawania przyciskiem z napisem „Anuluj”, oraz zapisać podzadanie tym z napisem „Zapisz podzadanie”. Formularz przedstawiony został na rysunku 18.

Lista zadań

[Dodaj zadanie](#)

Tytuł	Opis	Termin wykonania	Priorytet	Estymacja	Akcje
Zadanie 1	Opis	24 August 2024, 21:52	Wazne i pilne	2h	Edytuj Usuń Dodaj podzadanie

[Zapisz podzadanie](#)
[Anuluj](#)

Rysunek 18 Wygląd formularza dodającego podzadanie do istniejącego zadania; źródło: zrzut ekranu z aplikacji time-planer

4.1.2 Czasomierz Pomodoro

Aplikacja time-planer udostępnia ekran zawierający czasomierz Pomodoro. Użytkownik może skonfigurować długość przerwy, długość pracy oraz liczbę cykli Pomodoro za pomocą panelu przycisków nad licznikiem czasu. Każdy atrybut posiada przypisane do niego przyciski oznaczone znakami „-” zmniejszające, oraz „+” zwiększające jego wartość. Poniżej panelu konfiguracji znajduje się właściwy obszar czasomierza. Są to przyciski start, stop i reset oraz okrągła tarcza zegara w kolorze czerwonym zawierająca informację o pozostałym czasie, numerze aktualnego cyklu oraz napis mówiący użytkownikowi czy powinien zająć się pracą czy odpoczynkiem. Czasomierz Pomodoro został przedstawiony na rysunku 19.

Czasomierz Pomodoro

Długość przerwy

Długość pracy

Liczba cykli

[Start](#)
[Stop](#)
[Reset](#)

Praca: 1/5

20

Rysunek 19 Wygląd czasomierza Pomodoro w aplikacji time-planer; źródło: zrzut ekranu z aplikacji time-planer

4.1.3 Przypisywanie priorytetów za pomocą macierzy Eisenhowera

Aplikacja udostępnia możliwość grupowania zadań według ich priorytetów. Widok macierzy Eisenhowera pozwala na czytelne wyświetlenie listy zadań z podziałem na ich pilność wykonania. Macierz składa się z 4 identycznych obszarów zawierających pomniejsze listy zadań. Każda z list zawiera przypisane do odpowiedniego priorytetu zadania w formie kafelków. Same kafelki zawierają tytuły odpowiadających im zadań, a użytkownik może za pomocą kursora myszki wykonać operację „przeciągnij i upuść” aby przenieść zadanie do innego obszaru macierzy i tym samym zmienić priorytet. Wygląd macierzy został przedstawiony na rysunku 20.

Macierz Eisenhowera



Rysunek 20 Wygląd macierzy Eisenhowera w aplikacji time-planer; źródło: zrzut ekranu w aplikacji time-planer

4.2 Prezentacja działania aplikacji w praktyce

Aby przetestować działanie aplikacji należy jej użyć w realnych warunkach. Takowe mogą zostać wytworzone poprzez udostępnienie narzędzia kilku testerom. Ze względu na to, że aplikacja time-planer jest dostępna w przeglądarce i nie wspiera rozdzielczości dla smartfonów zostały wybrane osoby, które podczas swojej codziennej pracy mają dostęp do komputera. Pierwsza testująca osoba to student zarządzania, druga to programista Java, a trzecia to pracownik korporacyjny.

4.2.1 Student zarządzania

Studenci na co dzień stawiają czoła wielu wyzwaniom, w ich skład wchodzi uczęszczanie na wykłady, organizacja swojej pracy poza uczelnią, przygotowania do egzaminów czy tworzenie projektów na poszczególne zajęcia.

Lista zadań została użyta w pierwszej kolejności do opisanie wszystkich czynności, które należało wykonać podczas trwania testów. Były to przygotowania do poprawy kolokwium z ekonomii, które składały się z przeczytania rozdziałów z podręcznika, przeanalizowania notatek z wykładów oraz rozwiązanie próbnego testu. Kolejnym zadaniem było dokończenie rozdziału pracy licencjackiej obejmujące napisanie sekcji notabene metodologii badawczej, korekta wraz z redakcją wstępu oraz konsultacja zmian z promotorem. Ostatnim z zadań okazał się udział w seminarium online, które wymagało zarejestrowania się, włączenie seminarium o konkretnej godzinie oraz robienie notatek podczas trwania spotkania.

Macierz Eisenhowera posłużyła jako narzędzie do przyporządkowywania priorytetów dodanym wcześniej zadaniom. Przygotowania do kolokwium z ekonomii zostały umieszczone w sekcji „Pilne i Ważne”, natomiast seminarium online w obszarze „Niepilne i Nieważne”. Ostatnie z zadań, czyli napisanie rozdziału pracy licencjackiej zostało przypisane do priorytetu „Niepilne i Ważne”.

Licznik Pomodoro również znalazł zastosowanie w pracy z zadaniami studenta. Został on wykorzystany podczas pisania rozdziału pracy licencjackiej. Użytkownik ustawił 25 minut fazy roboczej oraz 5 minut odpoczynku, po czym odbył 4 sesje, w których opisał sekcję z metodologią oraz przeredagował wstęp swojego opracowania.

4.2.2 Java Developer

Programista Java często musi pracować nad wieloma projektami równocześnie, jego praca wymaga zarządzania wieloma zadaniami jak pisanie kodu aplikacji, uczestnictwo w spotkaniach oraz przeglądanie kodu współpracowników. Istotnym aspektem codzienności jest płynne przechodzenie między kolejnymi zadaniami.

Na początku każdego dnia programista tworzył listę zadań, które musiał zrealizować. Przykładowy dzień składał się z trzech obowiązkowych spotkań organizacyjnych oraz jednego opcjonalnego spotkania treningowego. Resztę dnia miał poświęcić na pisanie kodu

nowej funkcjonalności oraz poprawianie błędów w oddzielnej aplikacji i przejrzanie zmian wprowadzanych przez dwóch współpracowników.

Jako że wpisane spotkania były obowiązkowe, to w macierzy Eisenhowera trafiły do ćwiartki „Pilne i Ważne”. Opcjonalny trening został natomiast umieszczony w sekcji „Niepilne i Nieważne”. Reszcie zadań został przypisany priorytet „Niepilne i Ważne”.

Programista korzystał z licznika Pomodoro podczas pracy nad zadaniami związanymi z pisanem kodu oraz przeglądaniem pracy współpracowników. Wspierało to skupienie i pozwalało na umieszczenie przerw w trakcie pracy. Wspomniane zajęcia zajęły znaczną część dnia, a przez to długość cyklu pracy została ustawiona na trzydzieści, natomiast wielkość przerwy na pięć minut.

4.2.3 Pracownik korporacji

Pracownik korporacji musi stawiać czoła wielu wyzwaniom codzienności. Jest odpowiedzialny za kilka zespołów i musi organizować ich pracę. Oprócz tego uczestniczy w spotkaniach, przygotowuje raporty dla zarządu i jest zobligowany do realizowania projektów w swojej firmie.

Lista zadań podobnie jak w przypadku programisty była uzupełniana na początku każdego dnia. Przykładowy dzień korzystania z aplikacji wskazał, że funkcjonalności usuwania jak i edytowania wpisanych wcześniej zadań są niezbędne, ponieważ pracownik korporacji musi liczyć się z dużą ilością zmian zaplanowanych wydarzeń oraz spotkań. Pierwotnie ustalone spotkania z zarządem musiały zostać przeniesione na kolejny dzień. Zajęcia rozdysponowane między pracownikami musiały się zmienić ze względu na nieobecność jednego z podwładnych. Jedynym niezmiennym z uprzednio ustalonych zadań było tworzenie raportu na koniec dnia pracy.

W przypadku pracy korporacyjnej wszystkie zaplanowane zadania były wymagane, a ich priorytety były z góry ustalone przez przełożonych. Z tego względu macierz Eisenhowera nie miała dużego zastosowania.

Licznik Pomodoro również nie okazał się pożyteczny w przypadku pracy w dużym otwarto-powierzchniowym biurze wypełnionym ludźmi. Ze względu na dużą liczbę dystrakcji oraz liczne rozmowy ze współpracownikami, ustalenie sztywnego czasu na pracę oraz na przerwę nie mogło mieć miejsca.

4.2.4 Zalety i wady testowanej aplikacji

Testowanie aplikacji w realnych warunkach wykazało szereg wad oraz zalet. Wada wskazana przez studenta to niedostępność aplikacji na urządzeniach mobilnych. Ze względu na duże wykorzystanie telefonu komórkowego przez tego użytkownika, dodatkowa aplikacja nie wspierająca tej platformy była niewygodna. Programista zwrócił natomiast uwagę na ubogi interfejs graficzny, który mógłby być lepiej dopracowany. Uznał, że choć nie wpływa to na działanie aplikacji, to jednak jest to jedna z kluczowych kwestii programu, którego użytkownik ma używać przez większą część dnia. Pracownik korporacyjny uznał natomiast, że choć lista zadań była przydatna, to pozostałe funkcjonalności nie sprawdziły się w jego przypadku. Wadą odnotowaną przez każdego z uczestników testu był brak instrukcji obsługi aplikacji, co sprawiało, że przez pierwsze momenty jej używania była nieczytelna i nieintuicyjna. Najlepiej ocenianym aspektem narzędzia time-planer była lista zadań, która umożliwiała uporządkowanie swoich celów w ciągu dnia. Wraz z macierzą Eisenhowera sprawiały, że użytkownicy nie pomijali swoich obowiązków oraz w łatwy sposób mogli rezygnować z mniej ważnych czynności na rzecz tych pilnych i kluczowych. Zarówno programista, pracownik korporacyjny jak i student zgłosili, że dużym plusem testowanego narzędzia jest jego darmowy dostęp. Programista zwrócił uwagę na licznik Pomodoro, który jego zdaniem przyczynił się do większego odprężenia podczas pracy, ponieważ pojawiły się regularne przerwy. Omawiane zalety oraz wady zostały zbiorczo przedstawione w tabeli 1.

Tabela 1 Wykaz zalet oraz wad aplikacji time-planer

ZALETY	WADY
Darmowy dostęp do aplikacji	Brak instrukcji obsługi aplikacji
Większe poczucie uporządkowania	Niedostępność na urządzeniach mobilnych
Ułatwienie priorytetyzowania zadań	Ubogi interfejs graficzny
Brak pomijania zadań, które miały zostać wykonane	Trudność używania podczas pracy wymagającej elastyczności oraz szybkiego reagowania na zmiany
Większe skupienie podczas pracy	
Robienie przerw podczas pracy i związane z tym mniejsze poczucie zmęczenia	

4.2.5 Porównanie utworzonej aplikacji z istniejącymi narzędziami

Funkcjonalność utworzonej aplikacji można porównać z już istniejącymi narzędziami stosowanymi do zarządzania czasem. Jednym z takich narzędzi jest wspomniana już wcześniej aplikacja todoist. Posiada ona znacznie ładniejszy interfejs graficzny, który sprawia, że korzystanie z narzędzia jest przyjemniejsze. Ergonomia listy zadań zdecydowanie przewyższa możliwości aplikacji time-planer. Pozwala grupować zadania w projekty oraz pomniejsze grupy. Umożliwia również filtrowanie zadań ze względu na dodane do nich etykiety oraz udostępnia sortowanie zadań ze względu na ich terminy. Aplikacja todoist posiada widok kalendarza, który czytelnie wyświetla dodane uprzednio zadania w poszczególnych dniach. Użytkownik ma możliwość stworzenia konta, którym może się logować na różnych platformach i mieć na nich dostęp do swoich zadań. Nie zawiera natomiast licznika Pomodoro oraz interfejsu macierzy Eisenhowera co jest jedyną wadą w porównaniu z aplikacją time-planer. Obie aplikacje są darmowe, natomiast todoist daje możliwość skorzystania z wersji PRO, która oferuje dodatkowe funkcjonalności jak przypomnienia o zadaniach czy dodawanie do nich lokalizacji.

Dostępne w internecie liczniki Pomodoro nie odbiegają znacząco jakością od licznika dostępnego w aplikacji time-planer. Powodem tego jest prostota samego narzędzia oraz sztywne założenia jego funkcjonalności. Poszczególne narzędzia mogą różnić się od siebie jedynie wyglądem interfejsu lub wsparciem dla różnych platform, z których mogliby korzystać użytkownicy. Zaprezentowana wcześniej aplikacja o nazwie marinaratimer posiada dodatkową funkcjonalność w postaci alarmu podczas skończenia się cyklu pracy lub przerwy. Jest to zdecydowanie przydatny aspekt, ponieważ użytkownik skupiony na pracy może nie zauważyć, iż powinien zrobić sobie przerwę, jeżeli nie zostanie o tym poinformowany.

Podsumowanie

Jako że czas jest jednym z najważniejszych zasobów jakie człowiek może wykorzystywać należy dbać o to, aby robić to z należytą starannością. Opracowanie wykazało, że prawidłowe zarządzanie czasem podczas wykonywania codziennych zadań jest ważne i pozwala oszczędzić ten cenny zasób oraz wykonać więcej czynności w ciągu dnia. Omawiane metody wspierające optymalne wykorzystanie czasu wskazują na to, że istnieją narzędzia, które pozytywnie wpływają na organizowanie czasu. Dają użytkownikom możliwość większego skupienia podczas wykonywania zadań i pozwalają uniknąć zakłóceń przeszkadzających w działaniu. Przedstawiony proces tworzenia aplikacji udowodnił, że technologie informatyczne mogą mieć zastosowanie w kontekście zarządzania. Odpowiednio wykorzystane mogą tworzyć funkcjonalne narzędzia i wspomagać codzienną pracę ich użytkowników. Podsumowując kooperacja teorii zarządzania czasem z nowoczesnymi technologiami może prowadzić do zwiększenia wydajności użytkowników stosujących zasady omawiane w opracowaniu.

Bibliografia

- Alfred Homère NGANDAM MFONDOUM, Mesmin TCHINDJANG, Jean Valery MEFIRE MFONDOUM, & Isabelle MAKOUET. (n.d.). Eisenhower matrix * Saaty AHP = Strong actions prioritization? Theoretical.
- Cirillo, F. (2007). *The Pomodoro Technique (The Pomodoro)*.
- Covey S., M. R. (1994). *First Things First*. New York: Simon & Schuster.
- Dokumentacja frameworka Angular*. (2024, 09 01). Retrieved from <https://v17.angular.io/docs>
- Dokumentacja frameworka Spring*. (2024, 09 01). Retrieved from <https://docs.spring.io/spring-framework/reference/index.html>
- Dokumentacja języka Java*. (2024, 09 01). Retrieved from <https://docs.oracle.com/en/java/>
- Fritz Ch., F. L. (2011). It's the Little Things That Matter: An examination of Knowledge Workers' Energy Management. *Academy of Management*.
- Green P., S. D. (2005). Does time management training work: an evaluation. *International Journal of Training and Development*.
- Grunt-Mejer, d. K. (2012). *Skuteczność treningów zarządzania czasem - przegląd badań*.
- Häfner A., S. A. (2010). Time management training and perceived control of time at work. *Journal of psychology*.
- Parkinson, C. N. (n.d.). *PARKINSON'S LAW*.
- Rosie Dunford, Q. S. (2014). The Pareto Principle.
- Seiwert, L. J. (1998). *Zarządzanie czasem. Bądź Panem własnego czasu*.
- Słownik SJP*. (2024, 09 01). Retrieved from <https://sjp.pl/>

Spis tabel i ilustracji

Rysunek 1 Przykład zastosowania techniki Pomodoro w aplikacji internetowej; źródło: https://pomodor.app/timer ; data odczytu: 27.07.2024	10
Rysunek 2 Przykład zastosowania techniki Pomodoro w aplikacji internetowej; źródło: https://www.marinaratimer.com/p5snj ; data odczytu: 27.07.2024.....	10
Rysunek 3 Przykład zastosowania macierzy Eisenhowera w aplikacji internetowej; źródło: https://app.todoist.com/app/filters-labels ; data odczytu: 27.07.2024	11
Rysunek 4 Przykład zastosowania prawa Parkinsona w aplikacji Asana; źródło: https://help.asana.com/hc/en-us/articles/14101461379867-Time-Tracking ; data odczytu: 20.08.2024.....	13
Rysunek 5 Przykład zastosowania prawa Parkinsona w aplikacji Jira; źródło: https://community.atlassian.com/t5/Jira-questions/Is-it-still-possible-to-add-a-estimation-field-to-a-subtask/qaq-p/1397610 ; data odczytu: 20.08.2024	13
Rysunek 6 Przykład zastosowania zasady Pareto w aplikacji internetowej; źródło: https://app.todoist.com/app/inbox ; data odczytu: 20.08.2024	15
Rysunek 7 Przedstawienie nazw oraz typów kolumn w bazie danych aplikacji time-planer; źródło: zrzut ekranu z aplikacji IntelliJ Idea	19
Rysunek 8 Przedstawienie drzewa katalogów projektu aplikacji time-planer; źródło: zrzut ekranu z aplikacji IntelliJ Idea.....	20
Rysunek 9 Przedstawienie plików zawartych w części portu w aplikacji time-planer; źródło: zrzut ekranu z aplikacji IntelliJ Idea.....	20
Rysunek 10 Przedstawienie pliku zawierającego klasę SubtaskEntity w aplikacji time-planer; źródło: zrzut ekranu z aplikacji IntelliJ Idea	21
Rysunek 11 Przedstawienie pliku zawierającego interfejs SubtaskEntityRepository w aplikacji time-planer; źródło: zrzut ekranu z aplikacji IntelliJ Idea.....	21
Rysunek 12 Przedstawienie pliku zawierającego klasę TaskServiceImpl w aplikacji time-planer; źródło: zrzut ekranu z aplikacji IntelliJ Idea	22
Rysunek 13 Przedstawienie usług dostępnych w aplikacji time-planer; źródło: strona wygenerowana przez bibliotekę swagger dostępna po uruchomieniu aplikacji pod adresem http://localhost:8080/swagger-ui/index.html#/	23
Rysunek 14 Przedstawienie plików składowych komponentu task-list w aplikacji time-planer; źródło: zrzut ekranu z aplikacji webstorm	24

Rysunek 15 Zawartość plików task-list.ts oraz task-list.html w aplikacji time-planer; źródło: zrzut ekranu z aplikacji WebStorm.....	25
Rysunek 16 Wygląd listy zadań aplikacji time-planer; źródło: zrzut ekranu z aplikacji time-planer.....	26
Rysunek 17 Wygląd formularza dodającego nowe zadanie; źródło: zrzut ekranu z aplikacji time-planer	26
Rysunek 18 Wygląd formularza dodającego podzadanie do istniejącego zadania; źródło: zrzut ekranu z aplikacji time-planer.....	27
Rysunek 19 Wygląd czasomierza Pomodoro w aplikacji time-planer; źródło: zrzut ekranu z aplikacji time-planer	27
Rysunek 20 Wygląd macierzy Eisenhowera w aplikacji time-planer; źródło: zrzut ekranu w aplikacji time-planer	28