


Dokumentacja projektu Tasker		
Prowadzący: mgr Sebastian Witek		
 Wyższa Szkoła Przedsiębiorczości i Administracji	Imię i nazwisko, nr albumu Tomasz Marzęda, 20993; Bartosz Siembor, 21003; Jakub Stawski, 21363 Marcin Samolej, 21001	Kierunek: Informatyka PUW
	Przedmiot: Zarządzanie przedsięwzięciami informatycznymi - laboratorium 24/25	Semestr: 5 Rok akademicki: 2024/2025

## Plan i Opis Realizacji Projektu Aplikacji Tasker

### 1. Wprowadzenie

Aplikacja Tasker ma na celu dostarczenie narzędzia do zarządzania zadaniami, które umożliwi skuteczne planowanie, monitorowanie i realizację zadań.

Projekt realizowany z wykorzystaniem metodyki Scrum, przy wsparciu narzędzi takich jak Linear do zarządzania zadaniami oraz GitHub do wersjonowania kodu. Mokupy przygotowane w Figma.

#### Funkcjonalności aplikacji:

- **Lista zadań:** Użytkownicy będą mogli dodawać zadania do listy, określać ich tytuły, opisy oraz terminy wykonania. Zadania przeterminowane będą oznaczane kolorem czerwonym, a wykonane zadania - kolorem zielonym.
- **Sortowanie i filtrowanie:** Umożliwi sortowanie zadań według daty lub tytułu, z opcją przywracania domyślnego porządku.
- **Przypomnienia:** Użytkownicy będą mogli ustawiać przypomnienia dla zadań, które będą zapisywane i wyświetlane w aplikacji.
- **Zarządzanie listami:** Będzie możliwe tworzenie i edycja list zadań poprzez intuicyjny interfejs.
- **Profil użytkownika:** Sekcja profilu umożliwi podgląd danych użytkownika i zarządzanie planami subskrypcji.
- **Rozszerzenia z wykorzystaniem AI:** W przyszłości aplikacja zostanie rozbudowana o sztuczną inteligencję, która będzie organizować dzień użytkownika na podstawie jego poleceń.
- **Interfejs mobilny:** Aplikacja będzie działać jako Progressive Web App (PWA), co umożliwi instalację jej jako aplikacji na komputerze lub telefonie.

## 2. Skład Zespołu Projektowego

- **Project Manager | Tomasz Marzęda (PM):** odpowiedzialny za planowanie projektu, zarządzanie komunikacją w zespole oraz zapewnienie zgodności realizacji z założonymi wymaganiami. PM zajmuje się także koordynacją działań w ramach metodyki Scrum, w tym organizowaniem sprint planningów, daily stand-upów oraz retrospektyw.
- **Deweloper 1 | Bartosz Siembor:** specjalista w frontendzie, którego głównym zadaniem jest przygotowanie i implementacja interfejsu użytkownika zgodnie z najlepszymi praktykami UX/UI. Jego praca obejmuje tworzenie responsywnych widoków, integrację z backendem oraz optymalizację wydajności w aplikacji.
- **Deweloper 2 | Jakub Stawski:** specjalista w backendzie, odpowiedzialny za projektowanie i implementację logiki biznesowej aplikacji, w tym realizację API, integrację z bazą danych oraz zarządzanie środowiskami produkcyjnymi i testowymi. Deweloper ten zajmuje się także zapewnieniem bezpieczeństwa danych.
- **Tester | Marcin Samolej:** zajmuje się testowaniem aplikacji na każdym etapie jej rozwoju. W zakres jego pracy wchodzi testy jednostkowe, funkcjonalne, integracyjne oraz regresyjne. Tester zgłasza błędy i monitoruje ich usuwanie, dbając o wysoką jakość produktu. Ze względu na ograniczone zasoby ludzkie, w projekcie zajmował się również przygotowaniem diagramów UML.

## 3. User Stories

### 1. Opis US: Tworzenie nowego zadania

- **Jako:** Zalogowany użytkownik
- **Chciałbym:** Mieć możliwość tworzenia nowych zadań poprzez wprowadzenie tytułu, opisu oraz terminu realizacji
- **Aby:** Organizować swoje zadania w sposób efektywny
- **Dodatkowe Informacje:**
  - Jeśli pola są puste, wyświetlamy komunikat "Wypełnij wszystkie wymagane pola"
  - Po zapisaniu zadania, użytkownik widzi je na liście w dashboardzie
  - Zadanie zapisuje się w bazie danych i synchronizuje z backendem
- **Makieta:** Formularz z polami do wprowadzenia tytułu, opisu i daty
- **Wycena:**
  - Analiza: 3h
  - Makieta: 2h
  - PM: 1h

### 2. Opis US: Edytowanie zadania

- **Jako:** Zalogowany użytkownik
- **Chciałbym:** Mieć możliwość edytowania istniejących zadań
- **Aby:** Móc aktualizować szczegóły zadań, gdy wymagają zmian
- **Dodatkowe Informacje:**
  - Edytowanie dostępne tylko dla właściciela zadania
  - Po zapisaniu zmian, dane są natychmiast aktualizowane na liście zadań
  - Wyświetlamy Toast "Zadanie zostało zaktualizowane"
- **Makieta:** Formularz edycji z predefiniowanymi wartościami istniejącego zadania
- **Wycena:**
  - Analiza: 2h

- Makieta: 1h
- PM: 1h

### 3. Opis US: Usuwanie zadania

- **Jako:** Zalogowany użytkownik
- **Chciałbym:** Mieć możliwość usuwania zadań
- **Aby:** Móc usuwać zadania, które nie są już potrzebne
- **Dodatkowe Informacje:**
  - Wyświetlamy potwierdzenie przed usunięciem ("Czy na pewno chcesz usunąć zadanie?")
  - Po usunięciu, zadanie znika z listy zadań i z bazy danych
  - Wyświetlamy Toast "Zadanie zostało usunięte"
- **Makieta:** Dialog potwierdzający usunięcie
- **Wycena:**
  - Analiza: 1h
  - Makieta: 1h
  - PM: 1h

### 4. Opis US: Notyfikacje o zadaniach

- **Jako:** Zalogowany użytkownik
- **Chciałbym:** Otrzymywać notyfikacje o zbliżających się terminach zadań
- **Aby:** Być na bieżąco z terminami realizacji zadań
- **Dodatkowe Informacje:**
  - Notyfikacja pojawia się na 24 godziny przed terminem zadania
  - Jeśli zadanie zostanie zakończone, notyfikacja nie jest wyświetlana
  - Notyfikacja zawiera nazwę zadania, termin oraz przycisk "Przejdź do zadania"
- **Makieta:** Push notification z opcją "Przejdź do zadania"
- **Wycena:**
  - Analiza: 2h
  - Makieta: 1h
  - PM: 1h

## 4. Scenariusze Testowe

### 1. Tworzenie nowego zadania

- **Cel testu:** Sprawdzenie poprawności działania funkcji tworzenia nowego zadania.
- **Warunki wstępne:** Użytkownik jest zalogowany i znajduje się na stronie głównej aplikacji.
- **Kroki testowe:**
  - Użytkownik klika przycisk "Dodaj zadanie".
  - Użytkownik wprowadza tytuł, opis i termin realizacji zadania.
  - Użytkownik klika przycisk "Zapisz".
- **Dane testowe:**
  - Tytuł: "Zakupy"
  - Opis: "Kupić mleko i chleb."
  - Termin: "2025-01-15"
- **Oczekiwany wynik:** Zadanie zostaje dodane do listy zadań i jest widoczne w dashboardzie.
- **Rodzaje:**

- Pozytywne: Wszystkie pola wypełnione poprawnie.
- Negatywne: Pozostawienie pustych pól, wpisanie nieprawidłowej daty.
- Graniczne: Wpisanie maksymalnej długości tytułu (255 znaków).

## 2. Edycja zadania

- **Cel testu:** Weryfikacja poprawności działania funkcji edytowania zadania.
- **Warunki wstępne:** Zadanie istnieje na liście zadań użytkownika.
- **Kroki testowe:**
  - Użytkownik klika ikonę edycji przy wybranym zadaniu.
  - Użytkownik zmienia treść pola "Opis".
  - Użytkownik klika przycisk "Zapisz".
- **Dane testowe:**
  - Nowy opis: "Dodać jajka do listy zakupów."
- **Oczekiwany wynik:** Zadanie zostaje zaktualizowane na liście.
- **Rodzaje:**
  - Pozytywne: Zmiana opisu na poprawny tekst.
  - Negatywne: Wprowadzenie pustego opisu.

## 3. Usunięcie zadania

- **Cel testu:** Sprawdzenie możliwości usunięcia zadania.
- **Warunki wstępne:** Zadanie istnieje na liście zadań użytkownika.
- **Kroki testowe:**
  - Użytkownik klika ikonę kosza przy wybranym zadaniu.
  - Użytkownik potwierdza chęć usunięcia w wyświetlonym dialogu.
- **Dane testowe:**
  - ID zadania: 12345
- **Oczekiwany wynik:** Zadanie zostaje usunięte z listy i bazy danych.
- **Rodzaje:**
  - Pozytywne: Usunięcie istniejącego zadania.
  - Negatywne: Próba usunięcia zadania, które nie istnieje.

## 4. Notyfikacje

- **Cel testu:** Weryfikacja poprawności działania notyfikacji o zbliżających się terminach.
- **Warunki wstępne:** Zadanie z terminem za mniej niż 24 godziny istnieje w bazie danych.
- **Kroki testowe:**
  - Użytkownik loguje się do aplikacji.
  - Aplikacja wyświetla notyfikację o zadaniu.
- **Dane testowe:**
  - Termin: "2025-01-15 12:00"
  - Nazwa zadania: "Spotkanie z klientem."
- **Oczekiwany wynik:** Notyfikacja wyświetla poprawne dane o zadaniu.
- **Rodzaje:**
  - Pozytywne: Zadanie z prawidłowym terminem.
  - Negatywne: Zadanie z przeszłym terminem.

## 5. Harmonogram Realizacji (3 miesiące, 6 sprintów)

Każdy sprint trwa 2 tygodnie i składa się z planowania, realizacji zadań, przeglądu oraz retrospektywy. Szczegółowy opis każdego sprintu znajduje się poniżej.

## **Sprint 1: Przygotowanie i Podstawowa Architektura**

- **Warsztaty projektowe:** Spotkanie z zespołem projektowym oraz interesariuszami w celu omówienia celów aplikacji, wymagań biznesowych i technicznych. Wynikiem warsztatów będzie lista wymagań funkcjonalnych oraz backlog projektu.
- **Inicjalizacja repozytorium na GitHubie:** Utworzenie struktury katalogów, plików konfiguracyjnych oraz podstawowego systemu CI/CD do automatyzacji procesów.
- **Przygotowanie struktury projektu:** Rozdzielenie kodu na moduł frontendowy (React) i backendowy (Node.js). Utworzenie podstawowych komponentów, klas oraz schematów w bazie danych.
- **Commity:** Inicjalizacja projektu, konfiguracja narzędzi CI/CD, podstawowe workflowy na GitHubie, pierwsze szkielety aplikacji.

## **Sprint 2: Implementacja Kluczowych Funkcji Zarządzania Zadaniem**

- **Tworzenie użytkownika, logowanie, rejestracja:** Implementacja modułu uwierzytelniania z użyciem JWT. Funkcje obejmują rejestrację nowych użytkowników, logowanie oraz zarządzanie sesjami.
- **Ekran listy zadań (frontend + backend):** Zaprojektowanie interfejsu użytkownika do przeglądania zadań oraz stworzenie odpowiednich endpointów API do pobierania danych z backendu.
- **Integracja z bazą danych:** Konfiguracja PostgreSQL, przygotowanie tabel i indeksów, a także stworzenie mechanizmu migracji danych.
- **Commity:** Dodanie modułów logowania, tworzenie podstawowych API do obsługi użytkowników, przygotowanie struktury bazy danych.

## **Sprint 3: Rozbudowa Funkcjonalności**

- **Tworzenie, edytowanie i usuwanie zadań:** Dodanie funkcji CRUD (Create, Read, Update, Delete) dla zadań. Wdrożenie walidacji danych oraz mechanizmów zabezpieczających przed nieautoryzowanym dostępem.
- **Przypisywanie zadań do użytkowników:** Stworzenie systemu przypisania zadań z uwzględnieniem różnych poziomów uprawnień.
- **Notyfikacje:** Implementacja modułu powiadomień, w tym e-mailowych oraz push, informujących użytkowników o zmianach w zadaniach.
- **Commity:** Implementacja CRUD dla zadań, dodanie systemu przypisywania zadań, rozwój logiki notyfikacji w backendzie.

## **Sprint 4: Testowanie i Optymalizacja**

- **Testy jednostkowe i integracyjne:** Przygotowanie zestawu testów pokrywających główne funkcjonalności aplikacji. Testy będą realizowane z wykorzystaniem takich narzędzi jak Jest oraz Postman.
- **Optymalizacja wydajności aplikacji:** Analiza wydajności aplikacji i optymalizacja wąskich gardeł, takich jak czas ładowania danych czy wydajność zapytań do bazy danych.
- **Usuwanie błędów zgłoszonych przez testera:** Praca nad poprawkami oraz ponowna weryfikacja ich skuteczności.
- **Commity:** Dodanie testów jednostkowych, optymalizacja zapytań do bazy, poprawa wydajności interfejsu użytkownika.

## **Sprint 5: Przygotowanie do Wdrożenia**

- **Dokumentacja użytkownika:** Opracowanie szczegółowej instrukcji obsługi aplikacji, uwzględniającej wszystkie kluczowe funkcje.
- **Testy akceptacyjne:** Przeprowadzenie testów akceptacyjnych z udziałem interesariuszy, weryfikacja zgodności z wymaganiami biznesowymi.
- **Przygotowanie aplikacji do publikacji:** Finalizacja procesu buildowania aplikacji, konfiguracja serwerów produkcyjnych oraz testowych.
- **Commity:** Finalizacja dokumentacji, testy akceptacyjne, przygotowanie skryptów deploymentowych.

## **Sprint 6: Wdrożenie i Retrospektywa**

- **Wdrożenie aplikacji na serwery produkcyjne:** Publikacja aplikacji oraz testy funkcjonalne w środowisku produkcyjnym. Monitorowanie stabilności i wydajności systemu.
- **Retrospektywa projektu:** Analiza przebiegu projektu, identyfikacja sukcesów i obszarów do poprawy. Sporzenie raportu podsumowującego wyniki projektu.
- **Commity:** Finalny deployment, rozwiązywanie ostatnich zgłoszonych problemów, zamknięcie projektu na GitHubie.

## **5. Budżet Projektu**

- **Zasoby ludzkie:**
  - PM: 40 godzin / miesiąc, 100 PLN / godzina = 12 000 PLN
  - Deweloperzy: 160 godzin / miesiąc każdy, 80 PLN / godzina = 76 800 PLN
  - Tester: 80 godzin / miesiąc, 70 PLN / godzina = 16 800 PLN
- **Infrastruktura:**
  - Serwery: 2 000 PLN
  - Licencje na oprogramowanie: 1 000 PLN
- **Razem:** 108 600 PLN

## **6. Struktura Rozkładu Pracy (WBS)**

- 1. Przygotowanie**
  - Warsztaty projektowe
  - Konfiguracja narzędzi (Linear, GitHub)
- 2. Rozwój Funkcjonalności**
  - Moduł logowania i rejestracji
  - Moduł zarządzania zadaniami
  - System notyfikacji
- 3. Testowanie**
  - Testy jednostkowe
  - Testy integracyjne
  - Testy akceptacyjne
- 4. Wdrożenie**
  - Dokumentacja
  - Publikacja

## 7. Technologie użyte do przygotowania aplikacji

Aplikacja została przygotowana z wykorzystaniem następujących technologii:

- **Next.js:** Framework, który obsługuje zarówno frontend, jak i API aplikacji. Wykorzystano wersję z nowym systemem zarządzania podstronami (App Router).
- **PostgreSQL:** Baza danych używana do przechowywania informacji o użytkownikach i zadaniach.
- **Firebase Authentication:** Narzędzie do autoryzacji użytkowników, zapewniające bezpieczeństwo i weryfikację tokenów.
- **React:** Używany do budowy komponentów interfejsu użytkownika.
- **Progressive Web App (PWA):** Technologia umożliwiająca instalację aplikacji jako natywnej aplikacji na urządzeniach.
- **Struktura projektu:** Kod został podzielony na foldery API, komponentów, funkcji, hooków, bibliotek, narzędzi i testów, co ułatwia jego organizację i rozwój.
- **Mechanizmy hydracji i zarządzania layoutem:** Zastosowano rozwiązania takie jak `useEffect` i `isMounted` w React, aby zapobiegać problemom związanym z różnicami między renderowaniem po stronie serwera i klienta.
- **Logowanie zdarzeń (planowane):** Zastosowanie logów serwerowych jest przewidziane jako przyszłe ulepszenie.
- **Inicjalizacja Firebase:** Aplikacja wykorzystuje konfigurację Firebase do zarządzania autoryzacją użytkowników i ich działaniami.

Dzięki tym technologiom aplikacja oferuje stabilność, funkcjonalność oraz możliwość rozwoju w przyszłości.

## 8. Podsumowanie

Projekt aplikacji Tasker został szczegółowo zaplanowany zgodnie z metodyką Scrum. Uwzględniono wszystkie kluczowe aspekty, takie jak harmonogram, budżet, user stories oraz narzędzia. Realizacja projektu wymagała ścisłej współpracy zespołu i odpowiedniego zarządzania zasobami, co pozwoliło na dostarczenie wysokiej jakości produktu w zaplanowanym czasie.