Politechnika Poznańska
Wydział Informatyki i Telekomunikacji
Elektronika i Telekomunikacja

Projekt z przedmiotu

Zaawansowane Programowanie Terminali Mobilnych

Maciej Niedźwiecki
nr albumu: 147973

Prowadzący:
dr inż. Paweł Sroka

Poznań, 27.01.2025 r.

Spis treści:

1. Cel projektu.

Celem projektu było stworzenie funkcjonalnej aplikacji mobilnej na platformę iOS, wykorzystując język Swift oraz framework SwiftUI.
Aplikacja została zaprojektowana jako intuicyjny i estetyczny menedżer zadań, umożliwiający użytkownikom organizowanie codziennych obowiązków. Kluczową funkcjonalnością było dodawanie, edytowanie i usuwanie zadań, a także oznaczanie ich jako ukończone.

Rys. 1. Grafika ilustrująca wykorzystane technologie.

2. Struktura plików.

1. Plik **SimplyDoApp**

Plik odpowiedzialny za inicjalizację aplikacji jest punktem startowym jej działania, gdzie definiowane są podstawowe ustawienia oraz struktura aplikacji. Główny widok aplikacji określony jest za pomocą WindowGroup, który wskazuje na ContentView jako interfejs wyświetlany użytkownikowi po uruchomieniu. Aplikacja wykorzystuje model ToDoItem, który zarządza danymi i ich przechowywaniem, korzystając z mechanizmu SwiftData w celu zapewnienia efektywnego zarządzania oraz trwałości danych.

2. Plik **ToDoItem**

Plik odpowiedzialny za reprezentację pojedynczego zadania definiuje klasę ToDoItem, która jest oznaczona adnotacją @Model, umożliwiającą integrację z mechanizmem SwiftData.

Klasa zawiera następujące pola:

- title (String): Przechowuje nazwę zadania.
- timestamp (Date): Określa datę i czas utworzenia zadania.
- isCompleted (Bool): Wskazuje status ukończenia zadania.

Konstruktor klasy inicjalizuje domyślne wartości dla wszystkich pól, co znacząco ułatwia proces tworzenia nowych obiektów zadania, zapewniając spójność i prostotę zarządzania danymi w aplikacji.

3. Plik **ContentView**

Plik odpowiedzialny za wyświetlanie listy zadań oraz interakcje użytkownika zapewnia dynamiczny i intuicyjny interfejs, umożliwiający zarządzanie zadaniami. Deklaracja @Query private var items umożliwia pobieranie danych z modelu ToDoItem za pomocą integracji z SwiftData.

Funkcjonalności obejmują:

- Wyświetlanie listy zadań z możliwością filtrowania według statusu (showCompleted) oraz wyszukiwania na podstawie wprowadzonego tekstu (searchText).
- Zmianę statusu zadania poprzez kliknięcie w ikonę „checkmark.circle.fill", co oznacza jego ukończenie lub przywrócenie jako nieukończone.
- Przycisk „Add Item", który otwiera widok tworzenia nowego zadania za pomocą CreateToDoView.
- Przycisk edycji, który umożliwia modyfikację istniejącego zadania poprzez przejście do widoku UpdateToDoView.

4. Plik **CreateView**

Plik odpowiedzialny za tworzenie nowego zadania, obsługuje proces dodawania elementu do listy zadań. Deklaracja @State private var item przechowuje dane nowego zadania podczas jego tworzenia, umożliwiając tymczasowe zarządzanie stanem formularza.

Formularz składa się z następujących elementów:

- TextField: Pole tekstowe do wprowadzania nazwy zadania.
- DatePicker: Kontrolka pozwalająca na wybór daty i godziny utworzenia zadania.
- Przycisk „Create":
- Dodaje nowe zadanie do kontekstu danych za pomocą context.insert(item).
- Zamyka widok tworzenia zadania, wykorzystując metodę dismiss().

5. Plik **UpdateToDoView**

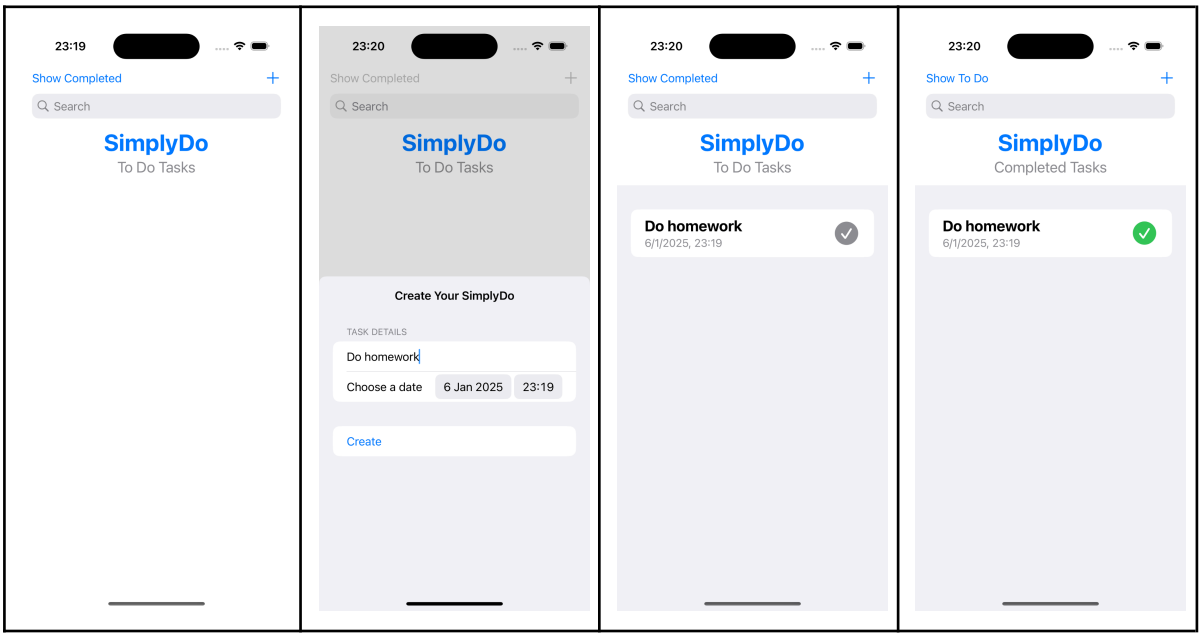
Plik odpowiedzialny za edycję szczegółów istniejącego zadania, umożliwia użytkownikowi modyfikację wybranego elementu listy zadań. Deklaracja @Bindable var item zapewnia referencję do edytowanego zadania, umożliwiając bezpośrednią aktualizację jego właściwości.

Formularz zawiera:

- TextField: Pole tekstowe pozwalające na edycję tytułu zadania.
- DatePicker: Kontrolka umożliwiająca zmianę daty przypisanej do zadania.
- Przycisk „Update":

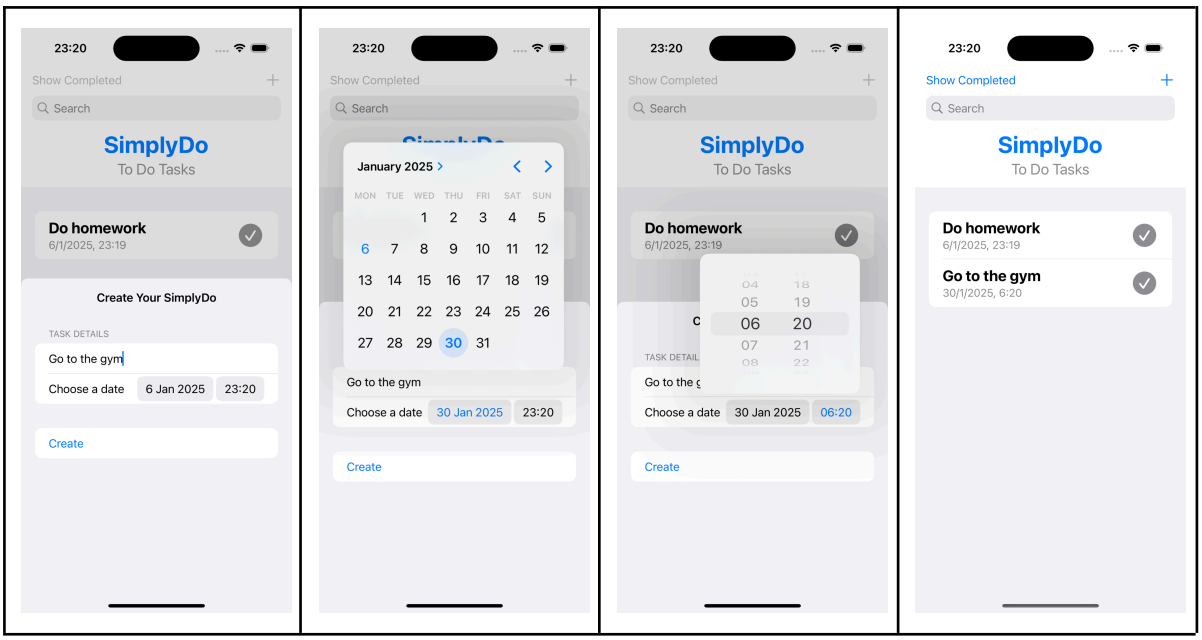
- Utrwala wprowadzone zmiany w zadaniu.
- Zamyka widok edycji za pomocą metody dismiss().

3. Test Aplikacji.



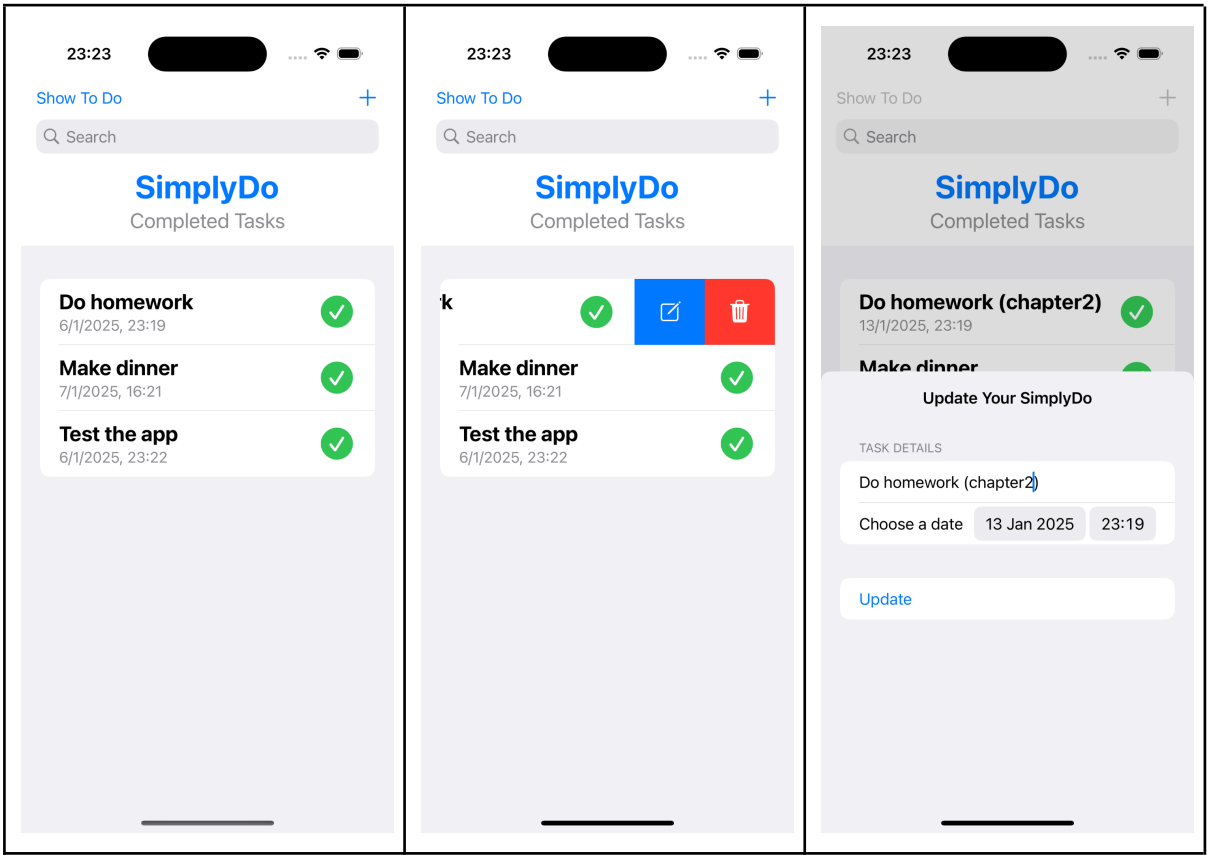
Rys. 2. Widok aplikacji podczas tworzenia zadania.

Początkowy widok aplikacji nie zawiera dodanych zadań. Aby rozpocząć, użytkownik może kliknąć ikonę „+”, która otwiera widok tworzenia nowego zadania.



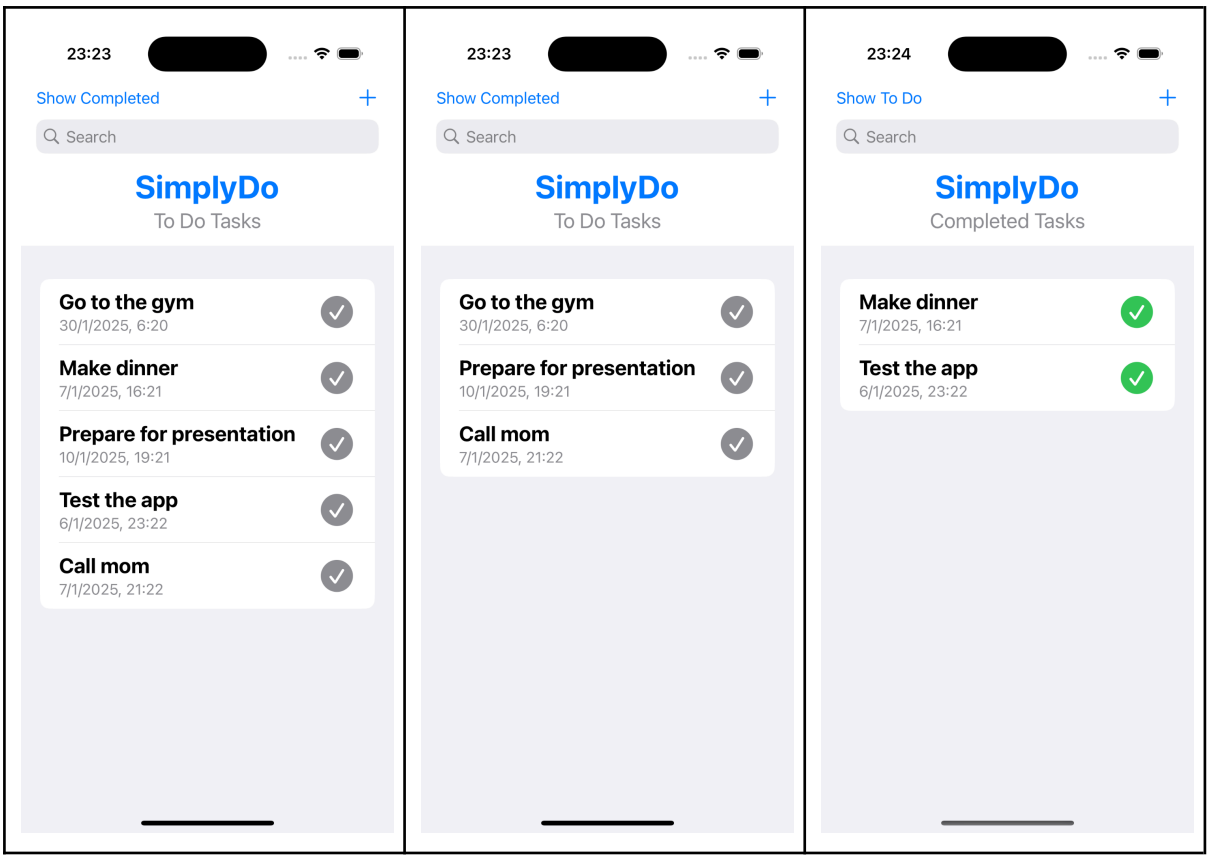
Rys. 3. Widok aplikacji podczas konfigurowania zadania.

Podczas tworzenia zadania użytkownik ma możliwość jego konfiguracji. W formularzu może wprowadzić nazwę zadania, a także wybrać datę z wbudowanego kalendarza oraz godzinę jego wykonania. Po dodaniu zadania wszystkie uzupełnione informacje są widoczne na liście w głównym widoku aplikacji.



Rys. 4. Widok aplikacji podczas edycji zadania.

Dodane zadanie może również zostać edytowan. W tym celu, użytkownik wykonuje gest przesunięcia w lewo na wybranym zadaniu, a następnie wybiera niebieską ikonę z symbolem długopisu. Po wybraniu tej opcji użytkownik zostaje przeniesiony do widoku edycji, gdzie może zmienić wcześniej skonfigurowane szczegóły zadania.

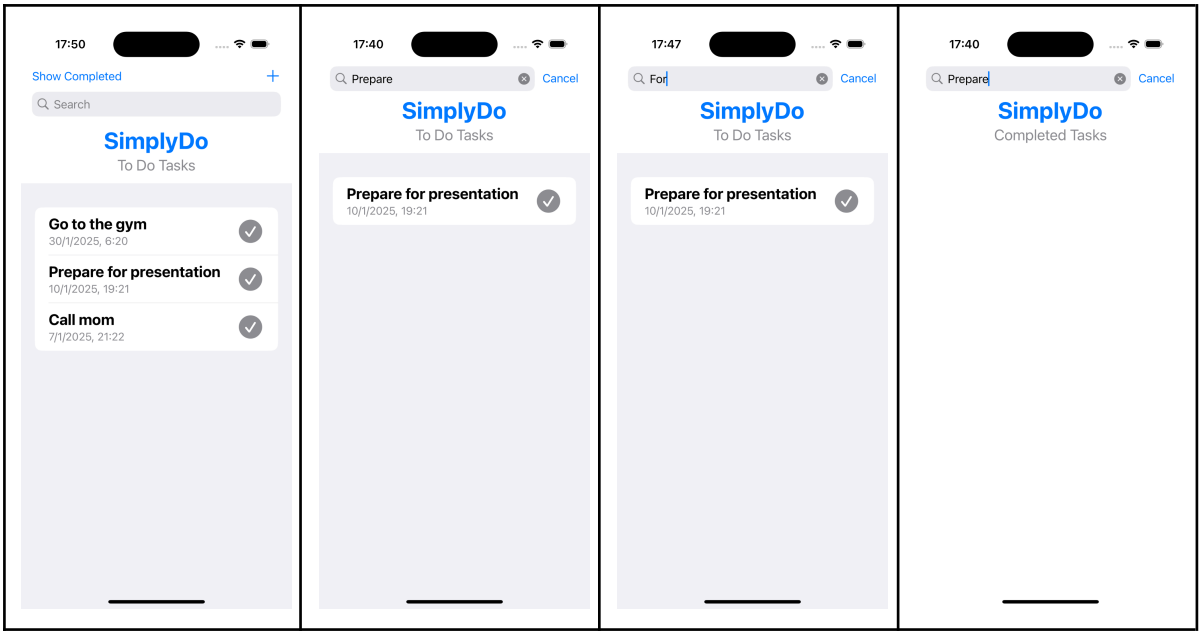


Rys. 5. Widok aplikacji podczas filtrowania widoku zadań.

Wszystkie utworzone zadania posiadają funkcję filtrowania, która pomaga w organizacji zadań. Każde zadanie ma przycisk filtrujący, który pozwala oznaczyć je jako ukończone. Po zaznaczeniu zadania przycisk zmienia kolor na zielony, symbolizując jego wykonanie.

Aplikacja umożliwia łatwe przełączanie widoku między ukończonymi a niewykonanymi zadaniami za pomocą dedykowanego przycisku „Show Completed” lub „Show To Do”. Dzięki temu użytkownik może szybko odnaleźć zadania w zależności od ich statusu.

Co istotne, zaznaczenie zadania jako ukończone nie usuwa go z listy, pozwalając użytkownikowi zachować pełną historię zadań. W celu trwałego usunięcia zadania należy wykonać gest przesunięcia w lewo i nacisnąć czerwony przycisk z ikoną kosza.



Rys. 6. Widok aplikacji podczas wyszukiwania tekstowego zadań.

Aplikacja oferuje również funkcję wyszukiwania zadań za pomocą słów kluczowych, co znacząco ułatwia zarządzanie większą liczbą wpisów. Aby skorzystać z tej opcji, wystarczy nacisnąć pasek wyszukiwania i wprowadzić poszukiwane frazy.

Co ważne, wyszukiwanie jest dynamiczne i działa w obrębie aktualnie wybranego filtru. Oznacza to, że jeśli użytkownik znajduje się w widoku ukończonych zadań, wyszukiwanie obejmuje jedynie te zadania. Analogicznie, w widoku zadań do wykonania wyszukiwane są tylko wpisy w tej kategorii.

4. Przyszłość aplikacji.

Aplikacja jest punktem wyjścia do przyszłego rozwoju i rozszerzenia funkcjonalności, koncentrując się na dostarczaniu intuicyjnych i praktycznych narzędzi do zarządzania zadaniami. Obecnie umożliwia użytkownikom dodawanie, edytowanie i organizowanie zadań, wspierając ich codzienne obowiązki.

Jednym z kierunków rozwoju może być **wprowadzenie ciemnego motywu**, który zwiększy wygodę użytkowania w warunkach słabego oświetlenia i dostosuje aplikację do preferencji użytkowników.

Kolejnym krokiem mogłoby być **dodanie powiadomień**, które przypominałyby użytkownikowi o zbliżających się terminach lub zadaniach do wykonania. Dzięki

możliwości personalizacji przypomnień aplikacja mogłaby lepiej wspierać użytkowników w zarządzaniu czasem.

Dodatkowo aplikację można wzbogacić o organizację zadań według **kategorii**, takich jak „Praca”, „Dom” czy „Nauka”. Pozwoliłoby to na szybszy dostęp do konkretnych grup zadań i łatwiejsze filtrowanie.

Rozwój aplikacji w tych kierunkach pozwoliłby na dostarczenie jeszcze bardziej wszechstronnego narzędzia, które spełniałoby potrzeby szerokiej grupy użytkowników.

5. Wnioski.

Stworzenie aplikacji To-Do na platformę iOS pozwoliło na zdobycie praktycznej wiedzy w zakresie programowania w języku Swift oraz wykorzystania frameworku SwiftUI. Projekt umożliwił zrozumienie znaczenia spójnego i intuicyjnego interfejsu użytkownika, a także zasad projektowania aplikacji zgodnych z ekosystemem Apple.

Praca nad implementacją głównych funkcjonalności, takich jak dodawanie, edytowanie oraz usuwanie zadań, pozwoliła na rozwinięcie umiejętności w zakresie zarządzania stanem aplikacji oraz tworzenia dynamicznych interfejsów. Integracja z mechanizmem SwiftData umożliwiła zrozumienie, jak efektywnie zarządzać danymi w aplikacji oraz zapewnić ich trwałość.

6. Bibliografia.

- Wykłady “Zaawansowane Programowanie Terminali Mobilnych”
  dr inż. Paweł Sroka
- Laboratoria z przedmiotu “Zaawansowane Programowanie Terminali Mobilnych”
  dr inż. Paweł Sroka
- Apple Developer - SwiftUI Tutorials *(link)*
- iOS Academy *(link)*
- tundsdev *(link)*
- ChatGPT
  pomoc w redakcji raportu, pisaniu kodu i code review (link)