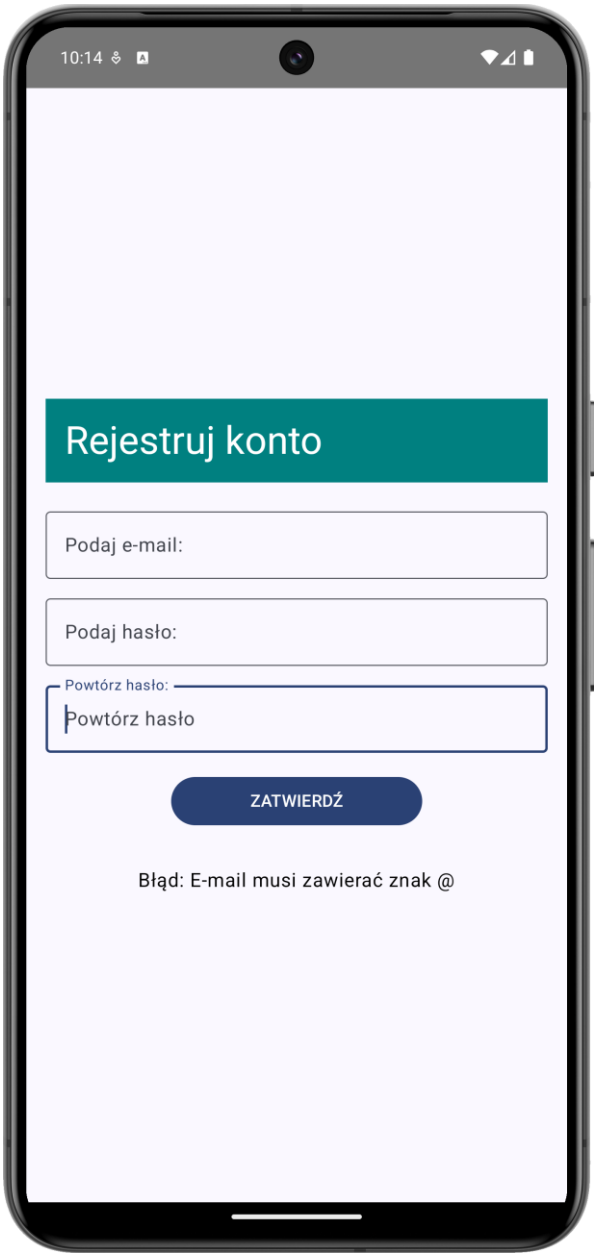
Tworzenie Aplikacji Rejestracji Konta w Android Studio z Jetpack Compose

Spis treści:

Wstęp

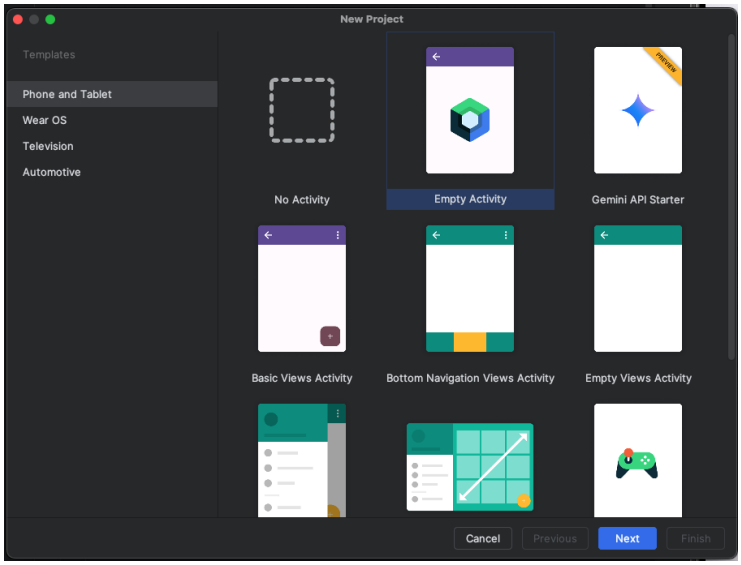
Na dzisiejszej lekcji nauczysz się, jak stworzyć aplikację rejestracji konta w Android Studio przy użyciu Jetpack Compose. Aplikacja umożliwi użytkownikom wprowadzenie adresu e-mail, hasła oraz potwierdzenie hasła, a następnie sprawdzi poprawność tych danych.

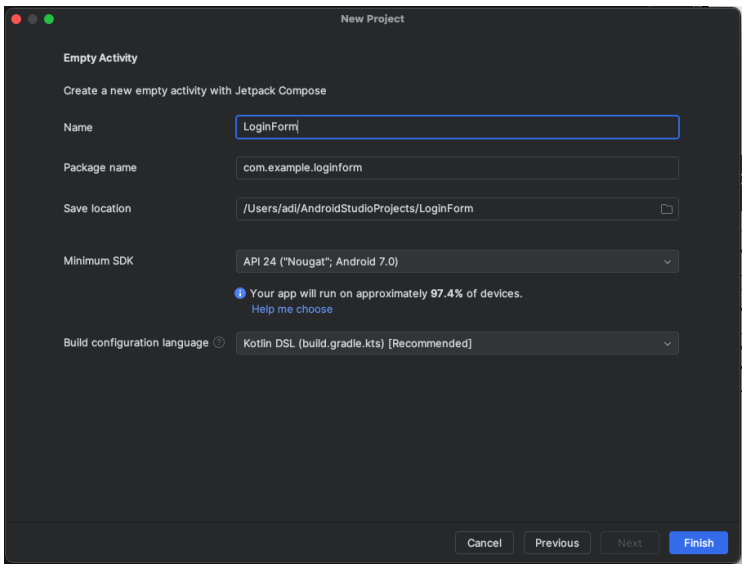


Krok 1: Tworzenie Nowego Projektu

1. Otwórz **Android Studio** i wybierz **"New Project"**.
2. Wybierz **"Empty Compose Activity"** i kliknij **"Next"**.
3. Nadaj nazwę aplikacji, np. RegistrationApp.
4. Upewnij się, że wybrane jest **Kotlin** jako język programowania.
5. Kliknij **"Finish"**, aby utworzyć nowy projekt.

Dlaczego to robimy? Tworzymy nowy projekt, który będzie bazą do rozwijania naszej aplikacji.





Krok 2: Przygotowanie pliku do pracy

1. Z pliku MainActivity.kt wyrzuć wszystkie zbędne linijki kodu.

```kotlin
package com.example.my

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.runtime.Composable
import androidx.compose.ui.tooling.preview.Preview
import com.example.my.ui.theme.MyTheme

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            MyTheme {
                // Call your screen function here
            }
        }
    }
}

@Preview(showBackground = true)
@Composable
fun DefaultPreview() {
    MyTheme {
        // Call your screen function here
    }
}
```

Dlaczego to robimy? Przygotowujemy kod aplikacji do pracy

Krok 3: Dodanie Funkcji Budującej Layout

1. Dodaj funkcję RegistrationScreen()
2. Uzupełnij funkcje “onCreate()” i “DefaultPreview()” o nowo dodaną funkcję

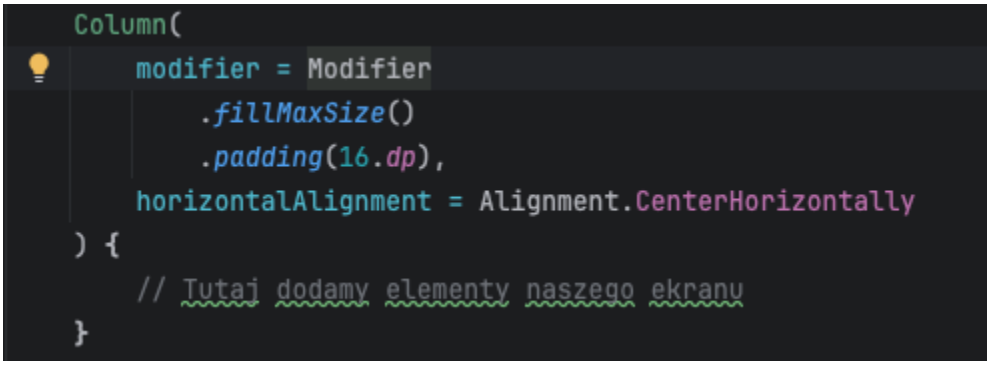
```
@Composable
fun RegistrationScreen() {
    // Tutaj zbudujemy nasz layout
}
```

```
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            MyTheme {
                RegistrationScreen() // Wywołanie naszej funkcji
            }
        }
    }
}

@Preview(showBackground = true)
@Composable
fun DefaultPreview() {
    MyTheme {
        RegistrationScreen() // Wywołanie naszej funkcji
    }
}
```

Dlaczego to robimy? Funkcja ta będzie odpoweidać za definiowanie wyglądu i działania naszej aplikacji

Krok 4: Dodanie Elementów Interfejsu Użytkownika

1. W funkcji RegistrationScreen dodaj podstawowy układ za pomocą Column



Dlaczego to robimy? Column będzie naszą linią do trzymania wszystkich elementów, które dodamy

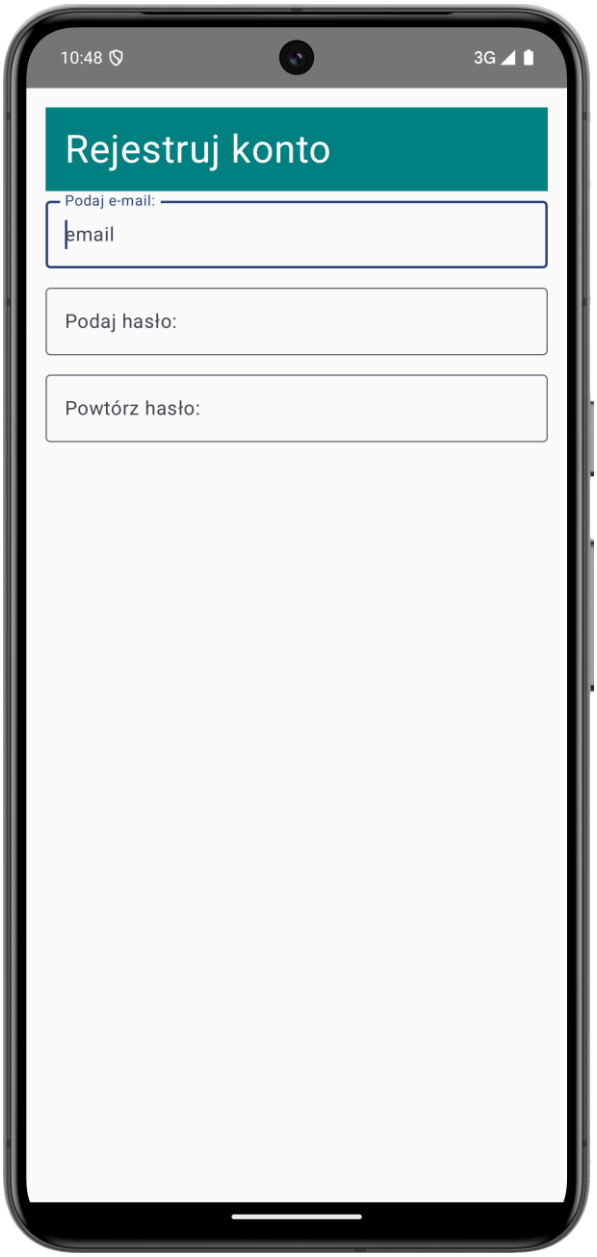
Krok 4: Dodawanie tekstu nagłówka "Rejestruj konto"

1. Wewnątrz Column dodaj tekst

Krok 5: Dodanie Pól Edycyjnych i Przycisku Zatwierdź

Tytuł: Dodawanie Pol Edycyjnych dla E-maila i Hasła

1. Dodaj pola tekstowe (OutlinedTextField) dla e-maila, hasła i powtórzenia hasła wewnątrz Column.



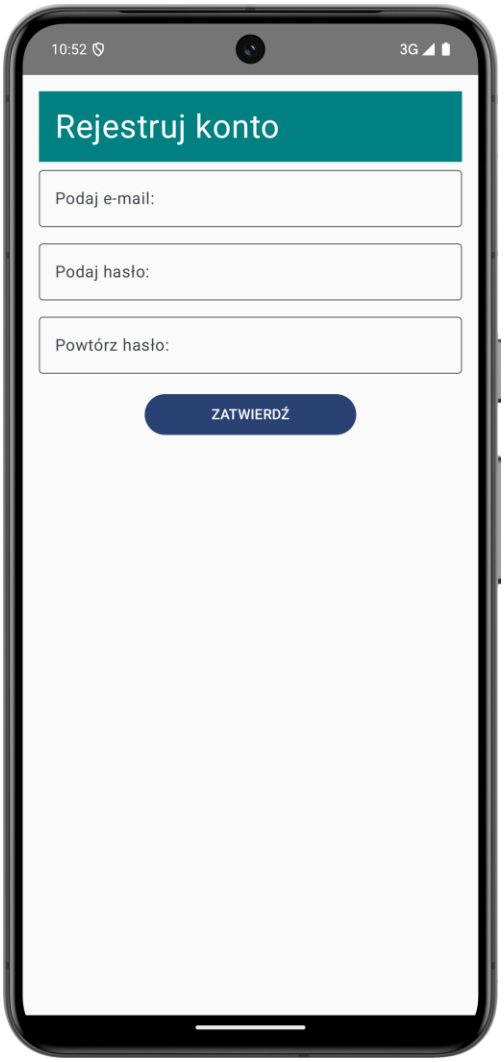
```
// Email input field
var email by remember { mutableStateOf( value: "") }

OutlinedTextField(
    value = email,
    onValueChange = { email = it },
    label = { Text( text: "Podaj e-mail:") },
    placeholder = { Text( text: "email") },
    modifier = Modifier
        .fillMaxWidth()
        .padding(bottom = 8.dp),
    singleLine = true
)

// Password input field
var password by remember { mutableStateOf( value: "") }

OutlinedTextField(
    value = password,
    onValueChange = { password = it },
    label = { Text( text: "Podaj hasło:") },
    placeholder = { Text( text: "Hasło") },
    visualTransformation = PasswordVisualTransformation(),
    modifier = Modifier
        .fillMaxWidth()
        .padding(bottom = 8.dp),
    singleLine = true
)
```

Dlaczego to robimy? Umożliwiamy użytkownikom wprowadzenie niezbędnych informacji w celu rejestracji.

Krok 6: Dodanie Guzika oraz Walidacji E-maila i Haseł

Tytuł: Sprawdzanie Poprawności Wprowadzonych Danych

1. Dodaj przysik do Column
2. Dodaj kod walidacji w funkcji obsługi kliknięcia przycisku

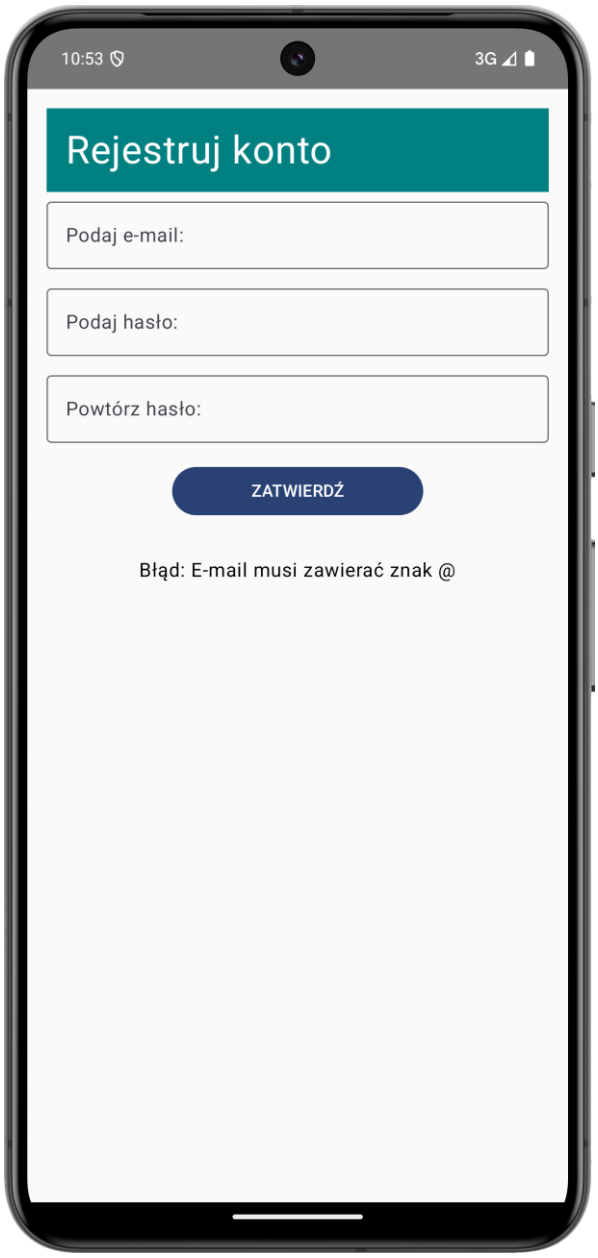

10:52
3G
Rejestruj konto
Podaj e-mail:
Podaj hasło:
Powtórz hasło:
ZATWIERDŹ

```
// State variable for message
var message by remember { mutableStateOf( value: "") }

Button(
    onClick = {
        when {
            !email.contains( other: "@") -> {
                message = "Błąd: E-mail musi zawierać znak @"
            }
            password != confirmPassword -> {
                message = "Błąd: Hasła nie są takie same"
            }
            email.isEmpty() || password.isEmpty() -> {
                message = "Błąd: Wszystkie pola muszą być wypełnione"
            }
            else -> {
                message = "Witaj $email"
            }
        }
    },
    modifier = Modifier
        .fillMaxWidth( fraction: 0.5f)
        .padding(bottom = 16.dp)
) {
    Text(text = "ZATWIERDŹ", color = Color.White)
}
```

Dlaczego to robimy? Sprawdzamy, czy dane wprowadzone przez użytkownika są poprawne i wyświetlamy odpowiednie komunikaty.

Krok 7: Wyświetlanie Komunikatów o Błędach i Sukcesie

1. Dodaj element Text do wyświetlania komunikatów o błędach lub powodzeniu.




```
// Display message
if (message.isNotEmpty()) {
    Text(
        text = message,
        color = Color.Black,
        modifier = Modifier.padding(top = 16.dp),
        style = MaterialTheme.typography.bodyLarge
    )
}
```

Dlaczego to robimy? Ważne jest, aby użytkownik wiedział, czy wprowadzone przez niego dane są poprawne czy też błędne.

Zadanie na ocenę

Na podstawie istniejącej aplikacji rejestracji wykonaj poniższe zadania:

Część 1: Dodanie opcji logowania

10. Rozszerz istniejący formularz, dodając do aplikacji ekran logowania.
    - **Po uruchomieniu aplikacji** użytkownik powinien zobaczyć dwa przyciski na głównym ekranie: **"Rejestracja"** i **"Logowanie"**.
    - Po wybraniu opcji "Rejestracja", użytkownik powinien zobaczyć istniejący formularz rejestracji, który już masz w aplikacji.
    - Po wybraniu opcji "Logowanie", użytkownik powinien zobaczyć nowy ekran logowania.
11. **Ekran logowania** powinien zawierać:
    - Pole do wpisania adresu e-mail.
    - Pole do wpisania hasła (z ukrytym tekstem).
    - Przycisk "ZALOGUJ SIĘ".
    - Po kliknięciu "ZALOGUJ SIĘ", sprawdź czy dane logowania zgadzają się z danymi podanymi podczas rejestracji.
    - Wyświetl odpowiedni komunikat: "Zalogowano pomyślnie" lub "Nieprawidłowy e-mail lub hasło".

Część 2: Dodanie zabezpieczeń i dodatkowych walidacji

12. Dodaj dodatkowe walidacje do formularza rejestracji:
    - Hasło powinno mieć co najmniej 8 znaków, zawierać przynajmniej jedną cyfrę oraz jeden znak specjalny (np. !@#$%^&*).
    - Adres e-mail powinien mieć poprawny format (sprawdź, czy jest zgodny z wzorcem np. xyz@example.com).
    - Wyświetl odpowiednie komunikaty, jeśli hasło lub e-mail nie spełniają wymagań.

Część 3: Stylizacja i czytelność

13. Popraw wygląd interfejsu:
    - Upewnij się, że przyciski mają zaokrąglone rogi.
    - Zastosuj inne kolory tła dla ekranów "Rejestracja" i "Logowanie", aby je odróżnić (np. niebieski dla rejestracji, zielony dla logowania).
    - Wykorzystaj MaterialTheme dla czcionek i kolorów, aby aplikacja wyglądała bardziej profesjonalnie.

Część 4: Opcjonalne zadania zaawansowane

14. Przechowuj dane rejestracyjne (e-mail i hasło) w pamięci urządzenia, używając SharedPreferences, tak aby po ponownym uruchomieniu aplikacji dane były nadal dostępne.
15. Dodaj możliwość przełączania widoczności hasła na ekranie rejestracji i logowania za pomocą ikonki "oko" obok pola hasła.

Punktacja:

- **Część 1**: 50 punktów (25 za dodanie ekranu logowania, 25 za poprawne przełączanie ekranów i sprawdzanie danych logowania)
- **Część 2**: 30 punktów (10 za walidację e-maila, 20 za walidację hasła)
- **Część 3**: 10 punktów (za poprawną stylizację i czytelność)
- **Część 4 (opcjonalne)**: 10 punktów za implementację SharedPreferences + 5 punktów za dodanie ikony widoczności hasła.