

An Android Application for keeping up with the latest headlines

Abstract

In the digital age, where information flows at an unprecedented rate, accessing timely, relevant news is more crucial than ever. With the increasing reliance on smartphones, mobile applications have become a primary source for people to stay informed. This project introduces an Android-based mobile application specifically designed to provide users with the latest news headlines and trending stories in a convenient, streamlined format. By integrating multiple news sources through APIs, the app delivers a comprehensive feed that encompasses a broad range of categories, including politics, technology, sports, business, health, and entertainment.

One of the app's primary goals is personalization—allowing users to customize their news feed based on their interests and preferences. With features like real-time notifications, bookmarks, and the ability to search for specific topics, the app ensures a tailored and engaging news experience. Additionally, the app's accessible interface is designed to provide an optimal user experience, offering dark mode and multi-language support. This news app is not only a tool for staying informed but also a platform that empowers users to connect with the stories that impact their lives.

Key features

- Customizable Notifications – Can customize the notifications
- Guides and Article – Helps the user to get relevant articles

Introduction

The media landscape has evolved dramatically over the past decade, with an increasing shift from traditional news outlets to digital platforms. The emergence of mobile technology has accelerated this change, making news more accessible than ever. As smartphones become a central device in people's lives, they offer the perfect medium to consume information quickly and efficiently. However, traditional news sources often lack the personalization, interactivity, and real-time delivery that users now demand. To meet these modern needs, this project proposes an Android application that aggregates news from a variety of sources, organizes it into intuitive categories, and allows users to personalize their experience.

This news app is designed to address several key issues with current news consumption methods. Firstly, the overwhelming amount of information available can make it difficult for users to find content that is both relevant and trustworthy. By consolidating news from reputable sources, the app aims to provide a streamlined solution, helping users avoid misinformation while staying up-to-date with credible news. Secondly, users often have varied interests and specific preferences, such as

wanting news focused on a particular region, topic, or even language. The app's personalization features allow users to select topics and set up alerts, ensuring a relevant and customized feed.

Project description

The project is an Android app for staying up-to-date on the latest in AI ethics. It offers a personalized news feed, curated expert insights, push notifications for breaking news, and a library of resources.

Users can bookmark articles, access visual trend analyses, and engage with an interactive glossary. Privacy-focused, the app ensures minimal data collection and operates locally where possible. Designed for both enthusiasts and professionals, this app fosters informed engagement with evolving AI ethics issues through easy access to reliable, relevant, and expertly curated content.

System requirements

- OS: Windows 10+, macOS 10.14+, or recent Linux versions.
- RAM: 8 GB minimum (16 GB recommended).
- Processor: Quad-core (Intel i5+ or equivalent).
- Storage: 10 GB of free space (SSD preferred).

Tools Used

- Android Studio: IDE for development
- Kotlin/Java: Programming languages.
- Firebase: For notifications and user data storage.
- News API: Source for news data.

Program:

LoginActivity.kt

```
imageView = Icons.Default.Lock,
        contentDescription = "lockIcon",
        tint = Color(0xFF6495ED)
    )
},
placeholder = { Text(text = "password", color = Color.Black) },
visualTransformation = PasswordVisualTransformation(),
colors = TextFieldDefaults.textFieldColors(backgroundColor = Color.Transparent)
```

)

```
Spacer(modifier = Modifier.height(12.dp))
```

```
if (error.isNotEmpty()) {
```

```
    Text(
```

```
        text = error,
```

```
        color = MaterialTheme.colors.error,
```

```
        modifier = Modifier.padding(vertical = 16.dp)
```

```
    )
```

```
}
```

```
Button(
```

```
    onClick = {
```

```
        if (username.isNotEmpty() && password.isNotEmpty()) {
```

```
            val user = databaseHelper.getUserByUsername(username)
```

```
            if (user != null && user.password == password) {
```

```
                error = "Successfully log in"
```

```
                context.startActivity(
```

```
                    Intent(
```

```
                        context,
```

```
                        MainPage::class.java
```

```
                    )
```

```
                )
```

```
                //onLoginSuccess()
```

```
            } else {
```

```
                error = "Invalid username or password"
```

```
            }
```

```
        } else {
```

```
            error = "Please fill all fields"
```

```

    }
},
shape = RoundedCornerShape(20.dp),
colors = ButtonDefaults.buttonColors(backgroundColor = Color(0xFF77a2ef)),
modifier = Modifier.width(200.dp)
.padding(top = 16.dp)
) {
    Text(text = "Log In", fontWeight = FontWeight.Bold)
}
Row(modifier = Modifier.fillMaxWidth()) {
    TextButton(onClick = {
        context.startActivity(
            Intent(
                context,
                RegistrationActivity::class.java
            ))
    }) {
        Text(text = "Sign up",
            color = Color.Black
        )
    }

    Spacer(modifier = Modifier.width(100.dp))

    TextButton(onClick = { /* Do something! */ })
    {
        Text(text = "Forgot password ?",
            color = Color.Black
        )
    }
}

```

```

    }
}
private fun startMainPage(context: Context) {
    val intent = Intent(context, MainPage::class.java)
    ContextCompat.startActivity(context, intent, null)
}

```

MainPage.kt

```

}))

        .clickable { onClick(index) }

        .height(180.dp), shape = RoundedCornerShape(8.dp), elevation = 4.dp
    ) {
        Surface(color = Color.White) {

            Row(
                Modifier
                    .padding(4.dp)
                    .fillMaxSize()

            )
            {
                Image(
                    painter = rememberImagePainter(
                        data = movie.urlToImage,
                        builder = {
                            scale(Scale.FILL)
                            placeholder(R.drawable.placeholder)
                            transformations(CircleCropTransformation())
                        }
                    ),

```

```

        contentDescription = movie.description,
        modifier = Modifier
            .fillMaxHeight()
            .weight(0.3f)
    Column(
        verticalArrangement = Arrangement.Center,
        modifier = Modifier
            .padding(4.dp)
            .fillMaxHeight()
            .weight(0.8f)
            .background(Color.Gray)
            .padding(20.dp)
            .selectable(true, true, null,
                onClick = {
                    Log.i("test123abc", "MovieItem: $index/n${movie.description}")
                    context.startActivity(
                        Intent(context, DisplayNews::class.java)
                            .setFlags(Intent.FLAG_ACTIVITY_NEW_TASK)
                            .putExtra("desk", movie.description.toString())
                            .putExtra("urlToImage", movie.urlToImage)
                            .putExtra("title", movie.title)
                    )
                })
    ) {

    Text(
        text = movie.title.toString(),
        style = MaterialTheme.typography.subtitle1,
        fontWeight = FontWeight.Bold
    )
}

```

```

        HtmlText(html = movie.description.toString())
    }
}
}
}
}
@Composable
fun HtmlText(html: String, modifier: Modifier = Modifier) {
    AndroidView(
        modifier = modifier
            .fillMaxSize()
            .size(33.dp),
        factory = { context -> TextView(context) },
        update = { it.text = HtmlCompat.fromHtml(html,
            HtmlCompat.FROM_HTML_MODE_COMPACT) }
    )
}
}

```

RegistrationActivity.kt

```

visualTransformation = PasswordVisualTransformation(),
        colors = TextFieldDefaults.textFieldColors(backgroundColor = Color.Transparent)
    )

    Spacer(modifier = Modifier.height(16.dp))

    TextField(
        value = email,

```

```

onValueChange = { email = it },
leadingIcon = {
    Icon(
        imageVector = Icons.Default.Email,
        contentDescription = "emailIcon",
        tint = Color(0xFF6495ED)
    )
},
placeholder = { Text(text = "email", color = Color.Black) },
colors = TextFieldDefaults.textFieldColors(backgroundColorColor.Transparent)
)

```

```

Spacer(modifier = Modifier.height(8.dp))

```

```

if (error.isNotEmpty()) {
    Text(
        text = error,
        color = MaterialTheme.colors.error,
        modifier = Modifier.padding(vertical = 16.dp)
    )
}

```

```

Button(
    onClick = {
        if (username.isNotEmpty() && password.isNotEmpty() && email.isNotEmpty()) {
            val user = User(
                id = null,
                firstName = username,
                lastName = null,
                email = email,

```



```

        password = password
    )
    databaseHelper.insertUser(user)
    error = "User registered successfully"
    // Start LoginActivity using the current context
    context.startActivity(
        Intent(
            context,
            LoginActivity::class.java
        )
    )

} else {
    error = "Please fill all fields"
}

},
shape = RoundedCornerShape(20.dp),
colors = ButtonDefaults.buttonColors(backgroundColor = Color(0xFF77a2ef)),
modifier = Modifier.width(200.dp)
    .padding(top = 16.dp)
) {
    Text(text = "Register", fontWeight = FontWeight.Bold)
}

Row(
    modifier = Modifier.padding(30.dp),
    verticalAlignment = Alignment.CenterVertically,
    horizontalArrangement = Arrangement.Center
) {

```

```

        Text(text = "Have an account?")

        TextButton(onClick = {
            context.startActivity(
                Intent(
                    context,
                    LoginActivity::class.java
                )
            )
        }) {
            Text(text = "Log in",
                fontWeight = FontWeight.Bold,
                style = MaterialTheme.typography.subtitle1,
                color = Color(0xFF4285F4)
            )
        }
    }
}

private fun startLoginActivity(context: Context) {
    val intent = Intent(context, LoginActivity::class.java)
    ContextCompat.startActivity(context, intent, null)
}

```

DisplayNews.kt

```
package com.example.newsheadlines
```

```

import android.content.Intent
import android.os.Bundle
import android.util.Log

```

```

import android.widget.TextView
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.Image
import androidx.compose.foundation.background
import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.padding
import androidx.compose.material.MaterialTheme
import androidx.compose.material.Surface
import androidx.compose.material.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import androidx.compose.ui.viewinterop.AndroidView
import androidx.core.text.HtmlCompat
import coil.compose.rememberImagePainter
import com.example.newsheadlines.ui.theme.NewsHeadlinesTheme

class DisplayNews : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            NewsHeadlinesTheme {
                // A surface container using the 'background' color from the theme
                Surface(

```

```

        modifier = Modifier.fillMaxSize(),
        color = MaterialTheme.colors.background
    ) {

        var desk = getIntent().getStringExtra("desk")
        var title = getIntent().getStringExtra("title")
        var uriImage = getIntent().getStringExtra("urlToImage")
        Log.i("test123abc", "MovieItem: $desk")

        Column(Modifier.background(Color.Gray).padding(20.dp), horizontalAlignment
= Alignment.CenterHorizontally, verticalArrangement = Arrangement.Center) {
            Text(text = ""+title, fontSize = 32.sp)
            HtmlText(html = desk.toString())
            /* AsyncImage(
                model = "https://example.com/image.jpg",
                contentDescription = "Translated description of what the image contains"
            )*/
            Image(
                painter = rememberImagePainter(uriImage),
                contentDescription = "My content description",
            )
        }
        // Greeting(desk.toString())
    }
}

@Composable
fun Greeting(name: String) {

```

```

        // Text(text = "Hello $name!")
    }

    @Preview(showBackground = true)
    @Composable
    fun DefaultPreview() {
        NewsHeadlinesTheme {
            // Greeting("Android")
        }
    }

    @Composable
    fun HtmlText(html: String, modifier: Modifier = Modifier) {
        AndroidView(
            modifier = modifier,
            factory = { context -> TextView(context) },
            update = { it.text = HtmlCompat.fromHtml(html,
                HtmlCompat.FROM_HTML_MODE_COMPACT) }
        )
    }

```


Output:

2:59

10.05 KB/s 4G 78%



Login

 selva

 ...

Successfully log in

Log In

[Sign up](#)

[Forgot password ?](#)

Sign Up



username



password



email

Register

Have an account? [Log in](#)

'Collective punishment': Why Supreme Court declared 'bulldozer justice' illegal



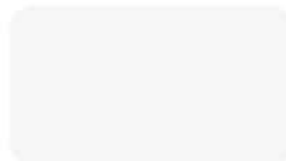
Can't shake off that cough? Why doctors say don't take it lightly



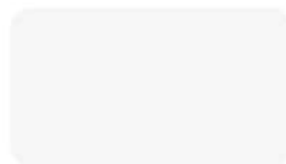
Watch: PM Modi's gesture as Nitish Kumar bows down to touch his feet



Zomato CEO's heartfelt message as rival Swiggy makes market debut



Rohit continues to train in Mumbai, no clarity on Australia departure



Challenges and Solutions

Challenges: Ensuring Real-Time Data Updates

Problem: Fetching and displaying up-to-the-minute news in real-time can be challenging, especially if there are delays in API responses or internet connectivity issues.

Solution: Implement caching with local storage. Using libraries like Retrofit for efficient API requests, combined with Room Database, the app can display cached data when offline and update it whenever an internet connection is available. This approach maintains smooth performance and reduces loading times.

Future Enhancements

- 1. AI-Driven Recommendations:** Use machine learning to personalize content based on reading history and provides necessary information.
- 2. Advanced Filtering:** Allow filtering by region, language, or sentiment.
- 3. Offline Mode:** Enable users to save articles for offline reading.
- 4. Multimedia Integration:** Add video summaries and podcasts for a richer experience.
- 5. Social Sharing and Comments:** Allow article sharing and community discussion within the app.

Conclusion

The Android News App provides a streamlined, customizable news experience that keeps users informed in real-time. Through efficient data handling, personalization, and performance optimization, the app meets modern user needs for on-the-go news. With future enhancements, it can further evolve into an intelligent application to make people to know about the news all over the world. The technical backbone of this app would rely on reliable APIs for consistent news delivery and a backend setup that supports caching and user preferences, thus optimizing performance and personalization.

