# An Android Application for Keeping Up with the Latest Headlines

# 1. Project Overview

This project focuses on developing an Android application designed to provide users with real-time access to the latest news headlines. By utilizing the Salesforce platform, we aim to integrate and streamline data storage, enhance app scalability, and manage user preferences. The app's purpose is to gather top headlines from various news sources and provide a seamless, customizable user experience.

# 2. Objectives

The primary objectives of this project are:

* To create a user-friendly Android application to aggregate and display the latest news headlines.
* To leverage Salesforce’s capabilities for managing user data, preferences, and personalized news feed customization.
* To ensure real-time updates and efficient data handling to avoid lags or outdated information.
* To explore the use of Salesforce’s key features and concepts in the context of an Android app for news aggregation.





**3. Functionalities:**

* Display News Articles: Present a scrollable list of news articles, each featuring a title, image, and brief description. Enable users to tap on an article to view detailed information.
* Dynamic UI with Jetpack Compose: Utilize composable functions to create a modular and maintainable UI structure. Implement state management to ensure the UI reflects changes in data seamlessly.
* Image Loading: Integrate Coil for asynchronous image loading, enhancing performance and user experience.
* Networking with Retrofit: Fetch news articles from a remote API, handling JSON data efficiently. Implement error handling and loading states to improve user interaction.
* User Interaction: Allow users to scroll through articles and navigate to detailed views, improving engagement with content.

**4. Salesforce Key Features and Concepts Utilized**

1. **User Data and Preference Management**: Salesforce handles user preferences and data storage, enabling customized news feeds based on user-selected topics and trends.
2. **Real-Time Data Updates**: Leveraging Salesforce’s ability to manage and provide real-time data access ensures the app delivers up-to-date news headlines without delays.
3. **API Integration for Data Retrieval and Updates**: Salesforce APIs are integrated to allow seamless retrieval and updates of user data, supporting a dynamic user experience.
4. **OAuth 2.0 for Secure Access**: Implemented to secure Salesforce and user data, enhancing the privacy and security of the application.
5. **User Activity Tracking**: Salesforce keeps records of user activity, aiding in identifying popular news categories and tailoring content based on usage patterns.
6. **Real-Time Notifications**: Using Salesforce triggers, the app provides real-time notifications for breaking news, keeping users informed instantly.

# 5. Detailed Steps to Solution Design

1. **Requirement Analysis and Research**:
   * Analyzed user requirements for a news aggregator app, noting desired features such as personalized feeds and trending news.
2. **Salesforce Setup**:
   * Set up Salesforce to handle backend functionalities like user management, preference settings, and data storage.
3. **App Design and Development**:
   * Developed the app’s UI with Android Studio, focusing on simplicity and ease of navigation.
   * Integrated Salesforce APIs for retrieving and updating user data in real time.
4. **Feature Integration**:
   * Added personalized feeds by categorizing news sources, trending topics, and user-selected preferences.
5. **Security and Permissions**:
   * Implemented OAuth 2.0 for secure access to Salesforce and user data.

# 6. Testing and Validation

* **Unit Testing**: To check individual functionalities such as news updates and personalized settings.
* **Integration Testing**: Ensured smooth data flow between Salesforce and the Android app.
* **User Acceptance Testing (UAT)**: Feedback gathered from target users to assess usability, design, and functionality.
* **Performance Testing**: Evaluated app responsiveness and data loading times.

## 7. Key Scenarios Addressed by Salesforce in the Implementation Project

* **Personalized News Feed**: Using Salesforce data management to customize the news feed based on user preferences.
* **Real-Time Notifications**: Implementing real-time notifications for breaking news based on Salesforce triggers.

# 8. Program

**8.1. AndroidManifest.xml:**

<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

    xmlns:tools="http://schemas.android.com/tools">

    <uses-permission android:name="android.permission.INTERNET"/>

    <uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE"/>

    <application

        android:allowBackup="true"

        android:dataExtractionRules="@xml/data\_extraction\_rules"

        android:fullBackupContent="@xml/backup\_rules"

        android:icon="@drawable/news\_app\_icon"

        android:label="@string/app\_name"

        android:supportsRtl="true"

        android:theme="@style/Theme.NewsHeadlines"

        tools:targetApi="31">

        <activity

            android:name=".DisplayNews"

            android:exported="false"

            android:label="@string/title\_activity\_display\_news"

            android:theme="@style/Theme.NewsHeadlines" />

        <activity

            android:name=".RegistrationActivity"

            android:exported="false"

            android:label="@string/title\_activity\_registration"

            android:theme="@style/Theme.NewsHeadlines" />

        <activity

            android:name=".MainPage"

            android:exported="false"

            android:label="@string/title\_activity\_main\_page"

            android:theme="@style/Theme.NewsHeadlines" />

        <activity

            android:name=".LoginActivity"

            android:exported="true"

            android:label="@string/app\_name"

            android:theme="@style/Theme.NewsHeadlines">

            <intent-filter>

                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />

            </intent-filter>

        </activity>

    </application>

</manifest>

**8.2. LoginActivity.kt:**

package com.example.newsheadlines

import android.content.Context

import android.content.Intent

import android.os.Bundle

import androidx.activity.ComponentActivity

import androidx.activity.compose.setContent

import androidx.compose.foundation.Image

import androidx.compose.foundation.background

import androidx.compose.foundation.layout.\*

import androidx.compose.foundation.shape.RoundedCornerShape

import androidx.compose.material.\*

import androidx.compose.material.icons.Icons

import androidx.compose.material.icons.filled.Lock

import androidx.compose.material.icons.filled.Person

import androidx.compose.runtime.\*

import androidx.compose.ui.Alignment

import androidx.compose.ui.Modifier

import androidx.compose.ui.graphics.Color

import androidx.compose.ui.res.painterResource

import androidx.compose.ui.text.font.FontWeight

import androidx.compose.ui.text.input.PasswordVisualTransformation

import androidx.compose.ui.tooling.preview.Preview

import androidx.compose.ui.unit.dp

import androidx.compose.ui.unit.sp

import androidx.core.content.ContextCompat

import androidx.core.content.ContextCompat.startActivity

import com.example.newsheadlines.ui.theme.NewsHeadlinesTheme

class LoginActivity : ComponentActivity() {

private lateinit var databaseHelper: UserDatabaseHelper

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

databaseHelper = UserDatabaseHelper(this)

setContent {

LoginScreen(this, databaseHelper)

}

}

}

@Composable

fun LoginScreen(context: Context, databaseHelper: UserDatabaseHelper) {

var username by remember { mutableStateOf("") }

var password by remember { mutableStateOf("") }

var error by remember { mutableStateOf("") }

Column(

Modifier

.fillMaxHeight()

.fillMaxWidth()

.padding(28.dp),

horizontalAlignment = Alignment.CenterHorizontally,

verticalArrangement = Arrangement.Center)

{

Image(

painter = painterResource(id = R.drawable.news),

contentDescription = "")

Spacer(modifier = Modifier.height(10.dp))

Row {

Divider(color = Color.LightGray, thickness = 2.dp, modifier = Modifier

.width(155.dp)

.padding(top = 20.dp, end = 20.dp))

Text(text = "Login",

color = Color(0xFF6485ED),

fontWeight = FontWeight.Bold,

fontSize = 24.sp,style = MaterialTheme.typography.h1)

Divider(color = Color.LightGray, thickness = 2.dp, modifier = Modifier

.width(155.dp)

.padding(top = 20.dp, start = 20.dp))

}

Spacer(modifier = Modifier.height(10.dp))

TextField(

value = username,

onValueChange = { username = it },

leadingIcon = {

Icon(

imageVector = Icons.Default.Person,

contentDescription = "personIcon",

tint = Color(0xFF6485ED)

)

},

placeholder = {

Text(

text = "username",

color = Color.Black

)

},

colors = TextFieldDefaults.textFieldColors(

backgroundColor = Color.Transparent

)

)

Spacer(modifier = Modifier.height(20.dp))

TextField(

value = password,

onValueChange = { password = it },

leadingIcon = {

Icon(

imageVector = Icons.Default.Lock,

contentDescription = "lockIcon",

tint = Color(0xFF6485ED)

)

},

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)

}

**8.3. MainPage.kt:**

package com.example.newsheadlines

import android.content.Context

import android.content.Intent

import android.content.Intent.FLAG\_ACTIVITY\_NEW\_TASK

import android.os.Bundle

import android.util.Log

import android.widget.TextView

import androidx.activity.ComponentActivity

import androidx.activity.compose.setContent

import androidx.activity.viewModels

import androidx.compose.foundation.Image

import androidx.compose.foundation.background

import androidx.compose.foundation.clickable

import androidx.compose.foundation.layout.\*

import androidx.compose.foundation.lazy.LazyColumn

import androidx.compose.foundation.lazy.itemsIndexed

import androidx.compose.foundation.selection.selectable

import androidx.compose.foundation.shape.RoundedCornerShape

import androidx.compose.material.Card

import androidx.compose.material.MaterialTheme

import androidx.compose.material.Surface

import androidx.compose.material.Text

import androidx.compose.runtime.\*

import androidx.compose.ui.Modifier

import androidx.compose.ui.graphics.Color

import androidx.compose.ui.text.font.FontWeight

import androidx.compose.ui.text.style.TextAlign

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 coil.size.Scale

import coil.transform.CircleCropTransformation

import com.example.example.Articles

import com.example.newsheadlines.ui.theme.NewsHeadlinesTheme

class MainPage : ComponentActivity() {

val mainViewModel by viewModels<MainViewModel>()

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContent {

NewsHeadlinesTheme {

// A surface container using the 'background' color from the theme

Surface(color = MaterialTheme.colors.background) {

Column() {

Text(text = "Latest NEWS", fontSize = 32.sp, modifier = Modifier.fillMaxWidth(), textAlign = TextAlign.Center)

MovieList(applicationContext, movieList = mainViewModel.movieListResponse)

mainViewModel.getMovieList()

}

}

}

}

}

}

@Composable

fun MovieList(context: Context, movieList: List<Articles>) {

var selectedIndex by remember { mutableStateOf(-1) }

LazyColumn {

itemsIndexed(items = movieList) {

index, item ->

MovieItem(context,movie = item, index, selectedIndex) { i ->

selectedIndex = i

}

}

}

}

@Composable

fun MovieItem(context: Context) {

val movie = Articles(

"Coco",

"",

" articl"

)

MovieItem(context,movie = movie, 0, 0) { i ->

Log.i("wertytest123abc", "MovieItem: "

+i)

}

}

@Composable

fun MovieItem(context: Context, movie: Articles, index: Int, selectedIndex: Int,

onClick: (Int) -> Unit)

{

val backgroundColor = if (index == selectedIndex) MaterialTheme.colors.primary else MaterialTheme.colors.background

Card(

modifier = Modifier

.padding(8.dp, 4.dp)

.fillMaxSize()

.selectable(true, true, null,

onClick = {

Log.i("test123abc", "MovieItem: $index/n$selectedIndex")

})

.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) }

)

}

}

**8.4. 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) }

)

}

**8.5. RegistrationActivity.kt:**

package com.example.newsheadlines

import android.content.Context

import android.content.Intent

import android.os.Bundle

import androidx.activity.ComponentActivity

import androidx.activity.compose.setContent

import androidx.compose.foundation.Image

import androidx.compose.foundation.background

import androidx.compose.foundation.layout.\*

import androidx.compose.foundation.shape.RoundedCornerShape

import androidx.compose.material.\*

import androidx.compose.material.icons.Icons

import androidx.compose.material.icons.filled.Email

import androidx.compose.material.icons.filled.Lock

import androidx.compose.material.icons.filled.Person

import androidx.compose.runtime.\*

import androidx.compose.ui.Alignment

import androidx.compose.ui.Modifier

import androidx.compose.ui.graphics.Color

import androidx.compose.ui.res.painterResource

import androidx.compose.ui.text.font.FontWeight

import androidx.compose.ui.text.input.PasswordVisualTransformation

import androidx.compose.ui.tooling.preview.Preview

import androidx.compose.ui.unit.dp

import androidx.compose.ui.unit.sp

import androidx.core.content.ContextCompat

import com.example.newsheadlines.ui.theme.NewsHeadlinesTheme

class RegistrationActivity : ComponentActivity() {

private lateinit var databaseHelper: UserDatabaseHelper

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

databaseHelper = UserDatabaseHelper(this)

setContent {

RegistrationScreen(this,databaseHelper)

}

}

}

@Composable

fun RegistrationScreen(context: Context, databaseHelper: UserDatabaseHelper) {

var username by remember { mutableStateOf("") }

var password by remember { mutableStateOf("") }

var email by remember { mutableStateOf("") }

var error by remember { mutableStateOf("") }

Column(

Modifier

.background(Color.White)

.fillMaxHeight()

.fillMaxWidth(),

horizontalAlignment = Alignment.CenterHorizontally,

verticalArrangement = Arrangement.Center)

{

Row {

Text(

text = "Sign Up",

color = Color(0xFF6485ED),

fontWeight = FontWeight.Bold,

fontSize = 24.sp, style = MaterialTheme.typography.h1

)

Divider(

color = Color.LightGray, thickness = 2.dp, modifier = Modifier

.width(250.dp)

.padding(top = 20.dp, start = 10.dp, end = 70.dp)

)

}

Image(

painter = painterResource(id = R.drawable.sign\_up),

contentDescription = "",

modifier = Modifier.height(270.dp)

)

TextField(

value = username,

onValueChange = { username = it },

leadingIcon = {

Icon(

imageVector = Icons.Default.Person,

contentDescription = "personIcon",

tint = Color(0xFF6485ED)

)

},

placeholder = {

Text(

text = "username",

color = Color.Black

)

},

colors = TextFieldDefaults.textFieldColors(

backgroundColor = Color.Transparent

)

)

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

TextField(

value = password,

onValueChange = { password = it },

leadingIcon = {

Icon(

imageVector = Icons.Default.Lock,

contentDescription = "lockIcon",

tint = Color(0xFF6485ED)

)

},

placeholder = { Text(text = "password", color = Color.Black) },

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(0xFF6485ED)

)

},

placeholder = { Text(text = "email", color = Color.Black) },

colors = TextFieldDefaults.textFieldColors(backgroundColor = Color.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)

}

# 9. Conclusion

The "Keeping Up with the Latest Headlines" project demonstrates the integration of Salesforce with Android app development, emphasizing the effective use of CRM capabilities for data management, security, and personalized content delivery. This project highlights Salesforce's versatility in supporting real-time data access and customized user experiences. By successfully integrating these Salesforce features, the app meets user demands for a reliable and up-to-date news aggregator.