An Android Application for Keeping Up with the Latest Headlines

**1)Introduction**

**1.1)Overview:**

Such an application could be designed to provide users with quick and easy access to the latest news and headlines from a variety of sources. Some of the key features that could be included in such an app are:

1. News Categories: The app could provide different news categories such as national, international, politics, business, technology, entertainment, sports, and more, to help users quickly filter the news according to their interests.
2. Personalized News Feed: Users could create a personalized news feed by selecting the categories and sources they want to follow. The app could then use machine learning algorithms to curate news stories that are relevant to the user's interests.
3. Push Notifications: The app could send push notifications to users to keep them informed about breaking news or important events.
4. Social Sharing: The app could allow users to share interesting news stories with their friends and followers on social media platforms such as Twitter, Facebook, and Instagram.
5. Offline Reading: The app could allow users to save articles for offline reading, so they can catch up on the news even when they don't have an internet connection.
6. Search Functionality: The app could include a search functionality to allow users to quickly find news stories based on keywords or topics they are interested in.
7. User Interface: The app could be designed with a user-friendly interface that is easy to navigate and read, with features such as dark mode for easier reading at night.

Overall, an Android application for keeping up with the latest headlines could be a useful tool for staying informed about the news and events happening around the world.

**1.2)Purpose:**

The purpose of an Android application for keeping up with the latest headlines is to provide users with a convenient way to stay informed about the latest news and events happening around the world.

With the rapid pace of news and information sharing, it can be challenging for individuals to keep up with the latest developments in their areas of interest. This is where a dedicated news app can be helpful, as it can aggregate news stories from multiple sources and present them in a clear and concise manner.

By providing users with access to the latest news stories and breaking news alerts, a news app can help them stay informed about important events in real-time. This can be especially important for individuals who need to keep up with the news for work or other purposes.

In addition, a news app can also offer personalized content recommendations based on the user's interests and reading history. This can help users discover new topics and sources that they might not have otherwise come across.

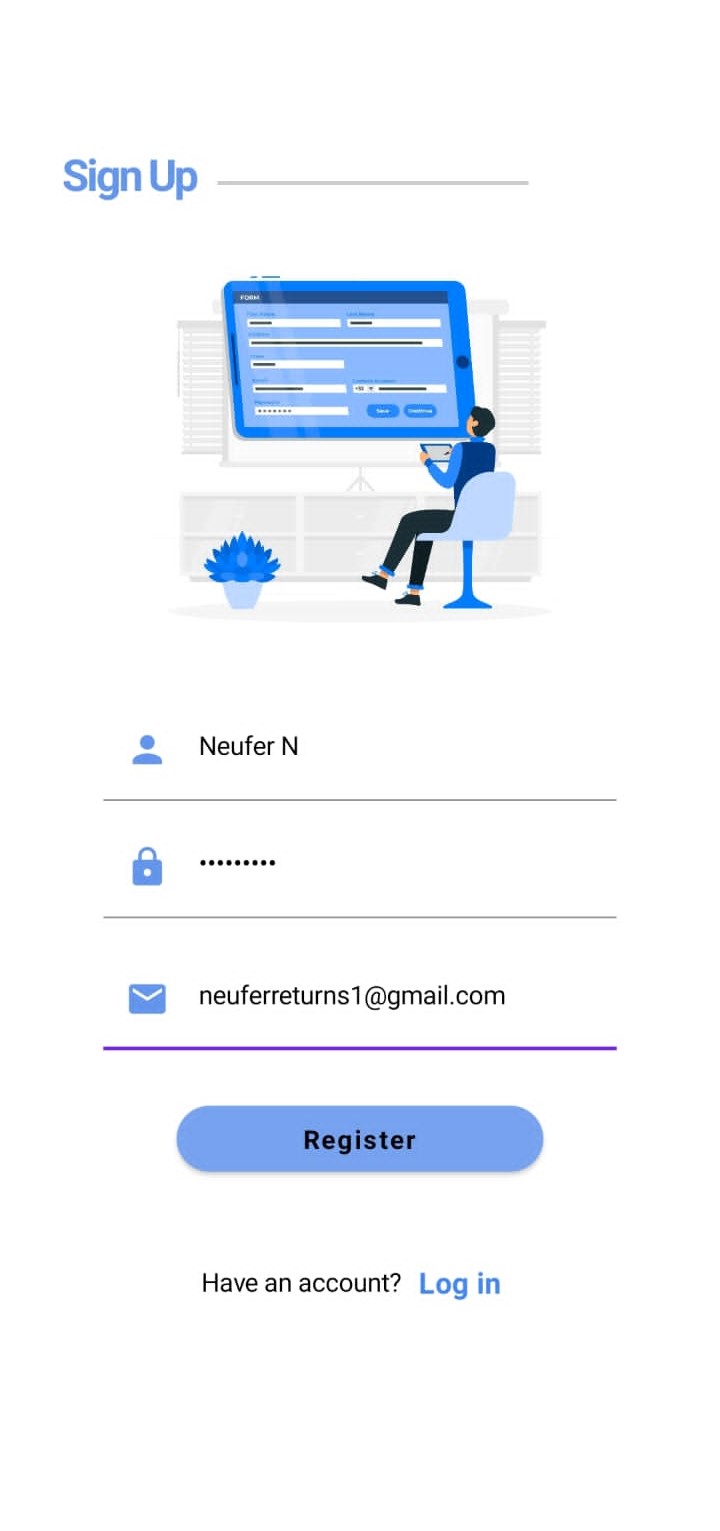
Overall, the purpose of an Android application for keeping up with the latest headlines is to make it easier for users to stay informed about the news that matters to them, without having to sift through multiple sources or spend a lot of time searching for relevant stories.

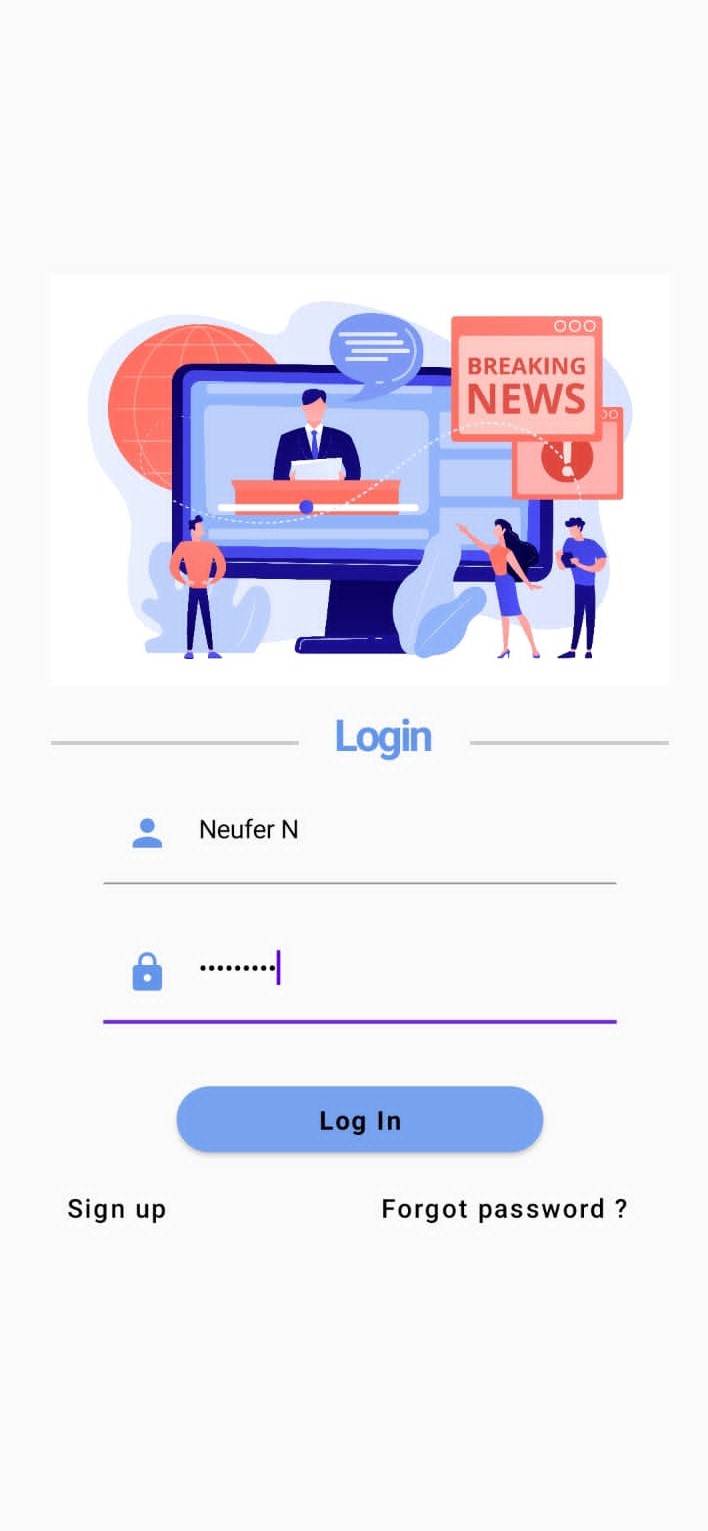
2.Problem Definition & Design Thinking:-

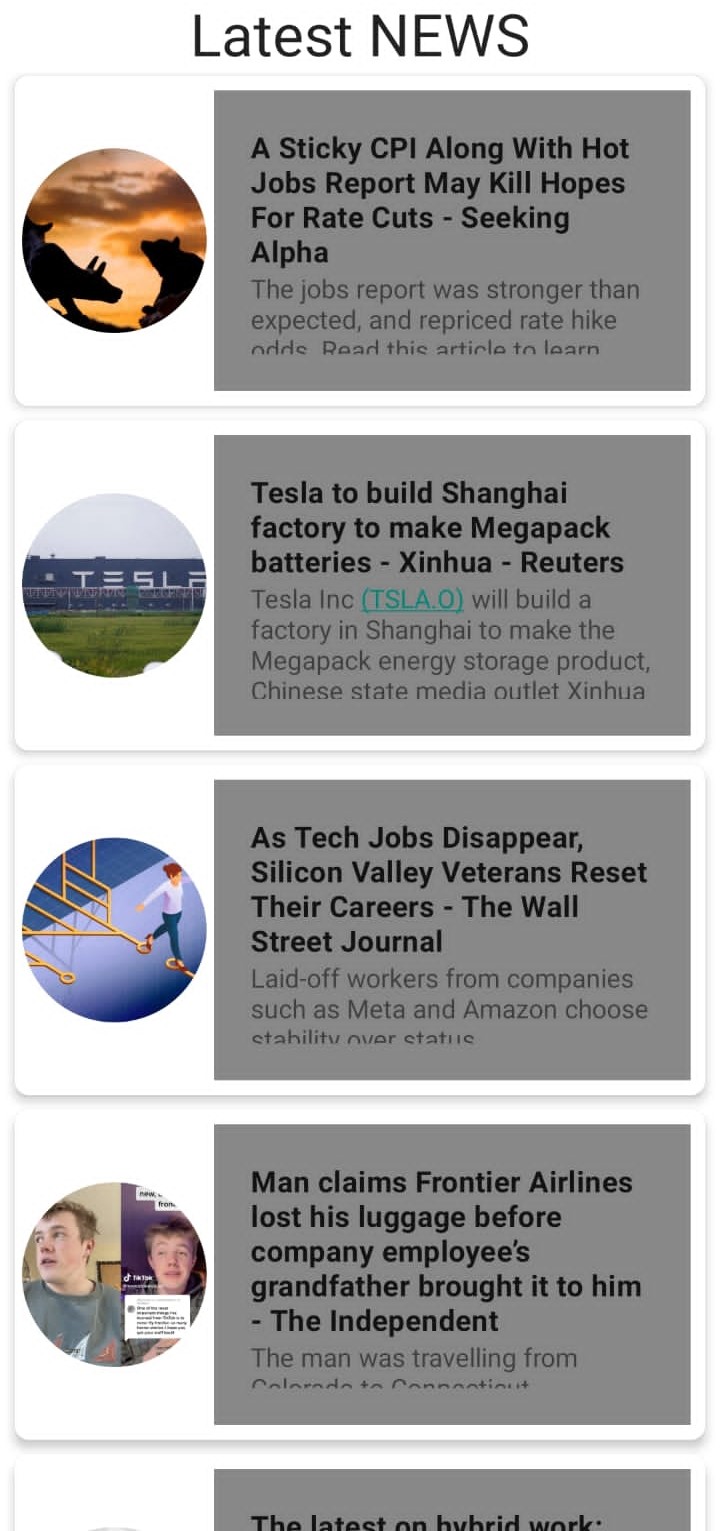
2.1Empathy Map:

2.2 Brainstorming Map:-

3.Result:







**Advantages:-**

There are several advantages to using an Android application for keeping up with the latest headlines:

1. Convenient Access: With a news app, users can easily access the latest news stories and breaking news alerts from anywhere, at any time, right from their mobile devices.
2. Personalization: A news app can offer personalized content recommendations based on the user's interests and reading history. This can help users discover new topics and sources that they might not have otherwise come across.
3. Multiple Sources: A news app can aggregate news stories from multiple sources, providing users with a more comprehensive view of the news than they might get from a single news outlet.
4. Quick and Easy Reading: News apps can present news stories in a clear and concise manner, making it easy for users to quickly skim through headlines and read articles of interest.
5. Push Notifications: Many news apps send push notifications to users to keep them informed about breaking news or important events. This can help users stay up-to-date on the latest developments in real-time.
6. Save for Later: Many news apps allow users to save articles for offline reading, so they can catch up on the news even when they don't have an internet connection.
7. Ad-Free Options: Some news apps offer ad-free options, which can provide a more streamlined and distraction-free reading experience.

**Disadvantages:-**

While there are many advantages to using an Android application for keeping up with the latest headlines, there are also a few potential disadvantages to consider:

1. Overwhelming Amount of Information: With so much news available, some users may find it difficult to keep up with the sheer volume of information presented by a news app. This can lead to information overload and decision fatigue.
2. Potential for Bias: Depending on the sources used by the app, there is a risk that the news presented may be biased towards certain viewpoints or agendas. Users should be mindful of this and seek out multiple sources to get a more balanced view.
3. Misinformation: With the prevalence of fake news and misinformation on the internet, there is a risk that some news stories presented by the app may be inaccurate or misleading. Users should exercise critical thinking skills and fact-check any stories that seem suspicious.
4. Privacy Concerns: Some news apps may collect personal data from users, such as their reading history or location data. Users should review the app's privacy policy and be aware of what data is being collected and how it will be used.
5. Subscription Fees: While many news apps are free to use, some may require a subscription fee to access certain content or features. Users should be aware of any fees associated with using the app before downloading and using it.

**5.Application:-**

To create an Android application for keeping up with the latest headlines, here are some steps that can be taken:

1. Define the app's purpose and target audience: Decide what features the app will have, what types of news will be included, and who the target audience is.
2. Choose a news API: Select a reliable news API to use for the app. The API will provide the latest news headlines and stories, which can be displayed in the app.
3. Design the user interface: Design a user-friendly interface that allows users to easily browse news stories and access different features of the app.
4. Develop the app: Use a programming language like Java or Kotlin to develop the app, using Android Studio as the development environment.
5. Test the app: Test the app thoroughly to ensure that it functions properly and is free of bugs.
6. Publish the app: Publish the app on the Google Play Store, making it available to users for download.
7. Promote the app: Use marketing and advertising strategies to promote the app to the target audience, such as social media campaigns or influencer partnerships.
8. Monitor user feedback: Monitor user feedback and reviews to identify areas for improvement and make updates to the app as necessary.

**6.Conclusion:**

In conclusion, an Android application for keeping up with the latest headlines can provide a convenient and personalized way for users to stay informed about the news that matters to them. With features such as personalization, push notifications, and search functions, the app can help users stay up-to-date on breaking news and important events in real-time. While there are potential disadvantages to using a news app, such as bias or misinformation, these risks can be mitigated through careful selection of sources and critical thinking when evaluating news stories. Overall, by providing a reliable and engaging news experience, an Android application for keeping up with the latest headlines can help people stay informed and connected to the world around them.

**7.Future Scope:**

The future scope of an Android application for keeping up with the latest headlines is promising. As technology advances and user preferences change, there are several areas in which such an app could evolve and improve:

1. Artificial intelligence and machine learning: As AI and machine learning technologies improve, they can be used to provide even more personalized news recommendations based on user preferences and reading behavior.
2. Augmented and virtual reality: These emerging technologies can provide an immersive news experience, allowing users to explore news stories and events in a more interactive and engaging way.
3. Audio and voice recognition: With the rise of smart speakers and voice assistants, news apps can integrate with these devices to provide a more seamless and hands-free news experience.
4. Blockchain technology: Blockchain technology can be used to improve the authenticity and verifiability of news stories, providing users with more confidence in the accuracy of the information presented.
5. Social media integration: As social media continues to play an important role in news dissemination, news apps can integrate with social media platforms to provide a more holistic and comprehensive news experience.
6. Localization: As the world becomes more globalized, there is a growing demand for news that is tailored to specific localities and regions. News apps can incorporate localization features to provide users with news stories relevant to their geographic location.

**8.Appendix**

**Source code:**

**Link:** [**https://github.com/smartinternz02/NewsHeadlines**](https://github.com/smartinternz02/NewsHeadlines)

**Code:-**

**DISPLAY NEWS.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) }**

**)**

**}**

**LOGIN ACTIVITY.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(0xFF6495ED),**

**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(0xFF6495ED)**

**)**

**},**

**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(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:**

**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) }**

**)**

**}**

**}**

**MODEL.KT:**

**package com.example.newsheadlines**

**data class Movie(val name: String,**

**val imageUrl: String,**

**val desc: String,**

**val category: String)**

**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(0xFF6495ED),**

**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(0xFF6495ED)**

**)**

**},**

**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(0xFF6495ED)**

**)**

**},**

**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(0xFF6495ED)**

**)**

**},**

**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)**

**}**