INTRODUCTION:

In today's fast-paced world, keeping up with the latest news and events is crucial. An Android application can help you stay informed by providing quick and easy access to the latest headlines from around the world.

Such an application can feature an intuitive user interface that displays the latest news articles from various sources, including international, national, and local news. Users can customize their newsfeed by selecting the topics they are interested in, such as politics, business, sports, entertainment, science, and technology.

The application can also feature a search function that allows users to search for specific topics, keywords, or sources. Moreover, users can save articles for offline reading or share them on social media platforms such as Facebook, Twitter, and WhatsApp.

To ensure the accuracy and reliability of the news, the application can integrate with reputable news sources, such as CNN, BBC, Reuters, and The New York Times. Additionally, the application can provide push notifications for breaking news stories and important events.

Overall, an Android application for keeping up with the latest headlines can help users stay informed and up-to-date with the latest news and events from around the world.

PURPOSE:

The purpose of an Android application for keeping up with the latest headlines is to provide users with a quick and easy way to stay informed about the latest news and events from around the world. The application aims to provide users with a personalized newsfeed that reflects their interests and preferences, making it easier for them to consume news that is relevant to them.

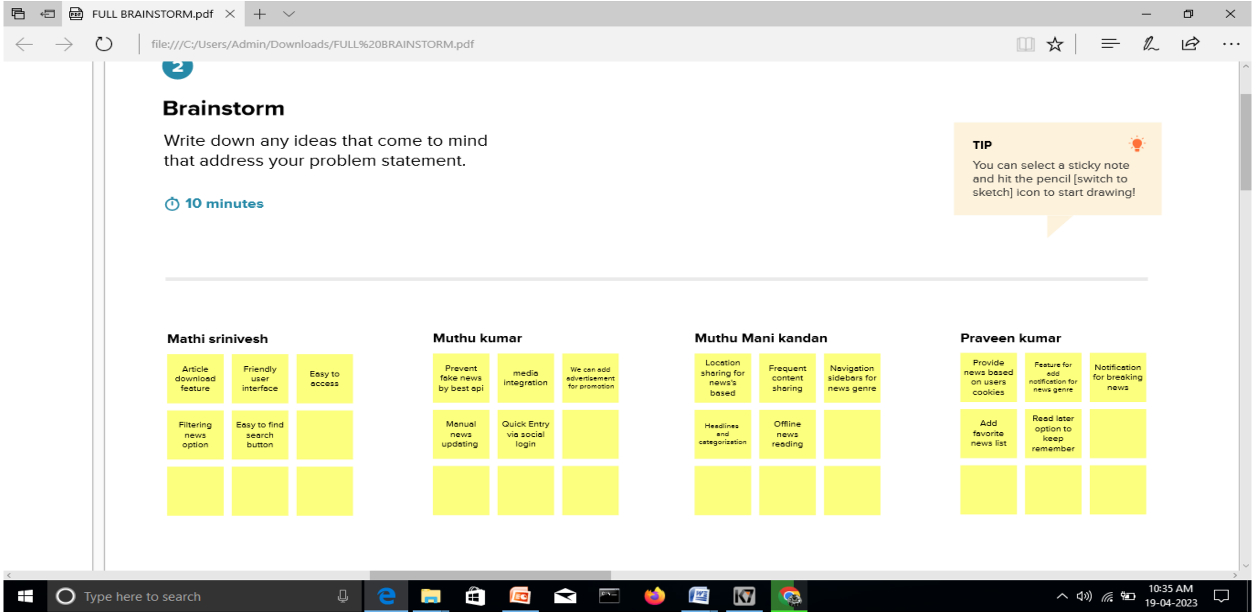
The application also aims to provide users with a seamless experience, allowing them to access the latest headlines with just a few clicks. Users can customize their newsfeed by selecting the topics they are interested in, and the application will filter out irrelevant news articles.

Moreover, the application aims to provide users with accurate and reliable news from reputable sources, ensuring that they are well-informed about the latest developments. The integration with reputable news sources helps users to trust the news they receive, which is particularly important in today's era of fake news.

Additionally, the application aims to provide users with the flexibility to save articles for offline reading or share them on social media platforms. This feature allows users to access news at their convenience and share important news stories with their friends and family.

Overall, the purpose of an Android application for keeping up with the latest headlines is to provide users with an easy and convenient way to stay informed about the latest news and events from around the world.

Brain Storm:



Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP
You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Mathi srinivesh

- Article download feature
- Friendly user interface
- Easy to access
- Filtering news option
- Easy to find search button

Muthu kumar

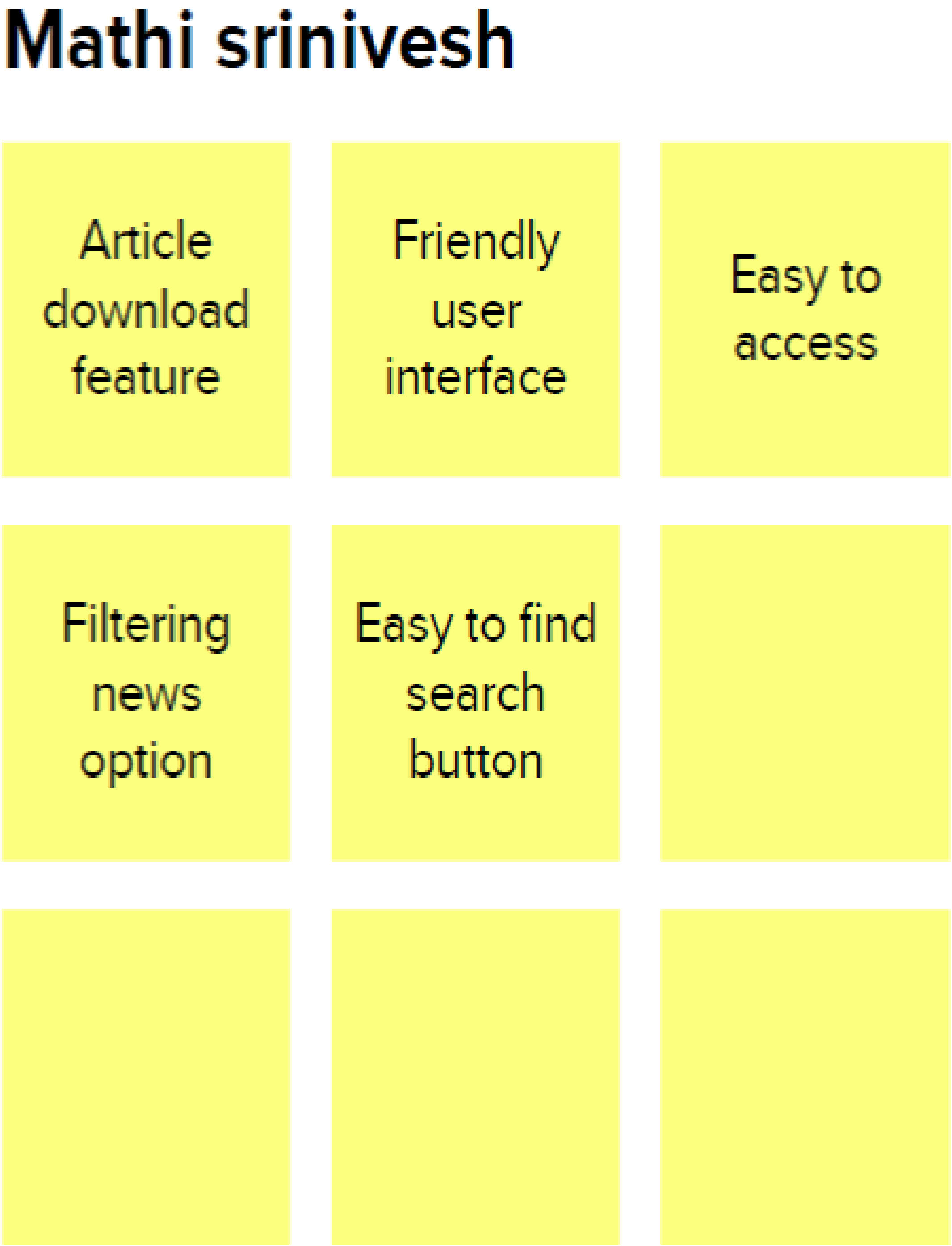
- Prevent fake news by best api
- media integration
- We can add advertisement for promotion
- Manual news updating
- Quick Entry via social login

Muthu Mani kandan

- Location sharing for news's based
- Frequent content sharing
- Navigation sidebars for news genre
- Headlines and categorization
- Offline news reading

Praveen kumar

- Provide news based on users cookies
- Feature for add notification for news genre
- Notification for breaking news
- Add favorite news list
- Read later option to keep remember



Mathi srinivesh

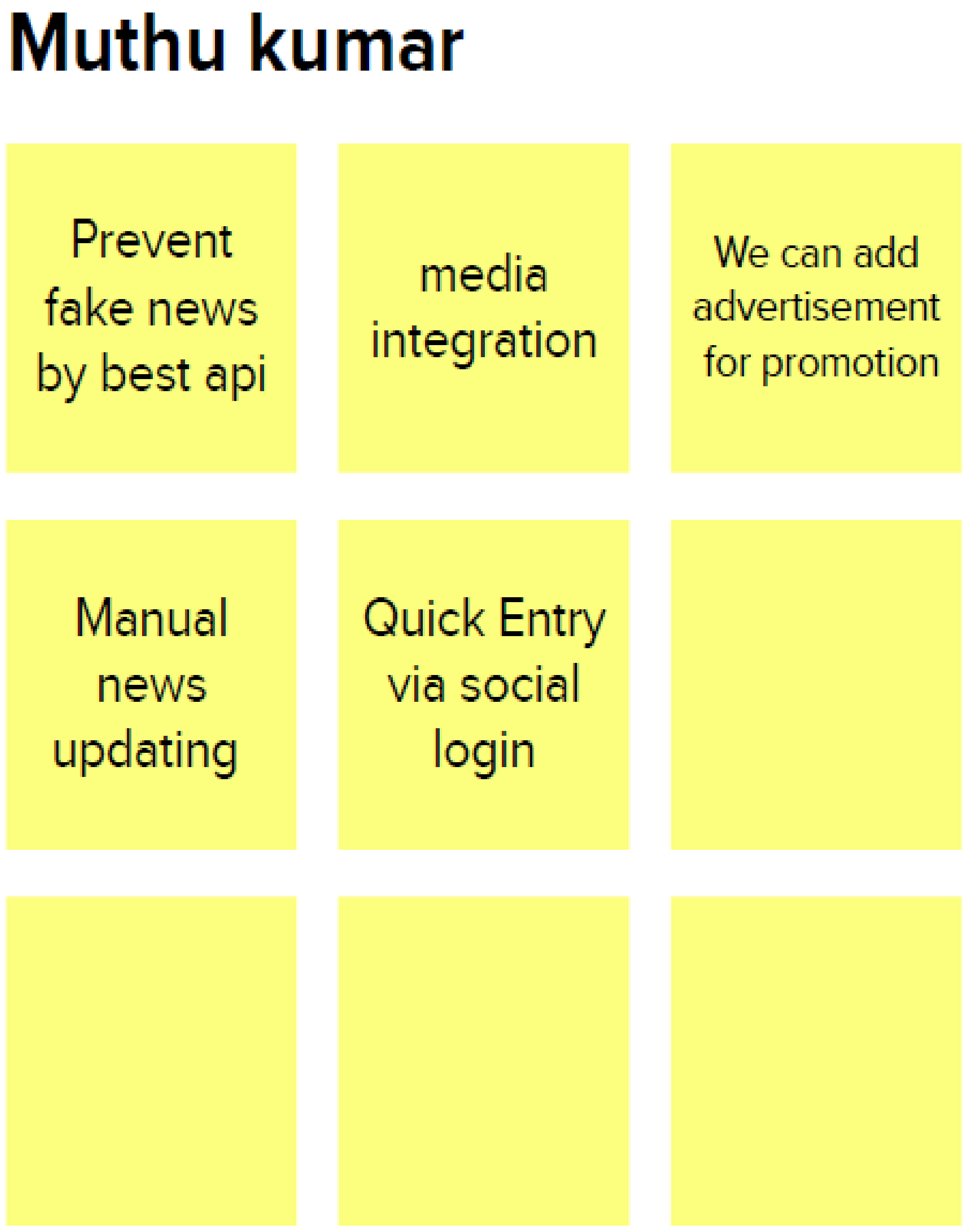
Article download feature

Friendly user interface

Easy to access

Filtering news option

Easy to find search button



Muthu kumar

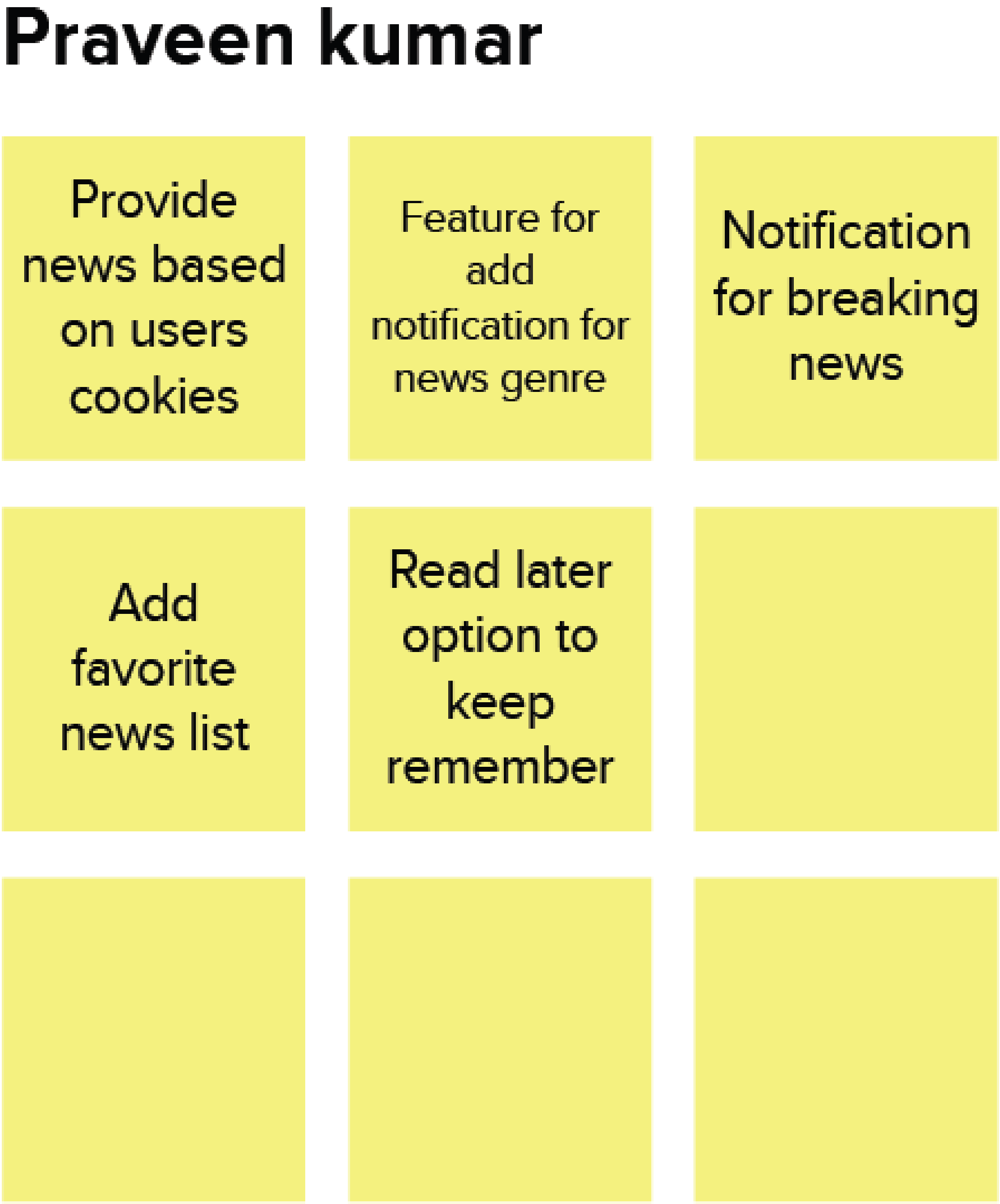
Prevent fake news by best api

media integration

We can add advertisement for promotion

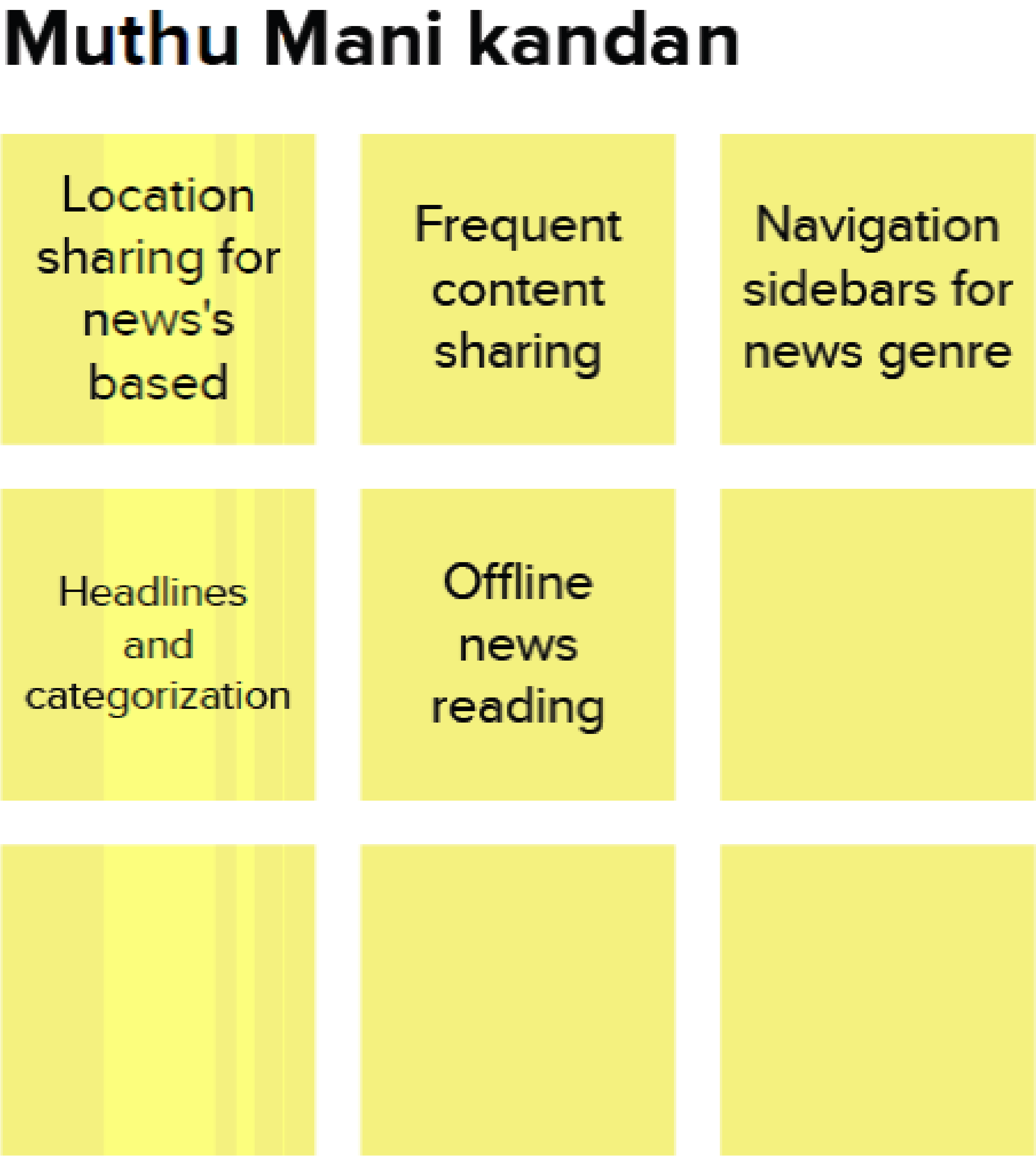
Manual news updating

Quick Entry via social login



Praveen kumar

Provide news based on users cookies	Feature for add notification for news genre	Notification for breaking news
Add favorite news list	Read later option to keep remember	



Muthu Mani kandan

Location sharing for news's based

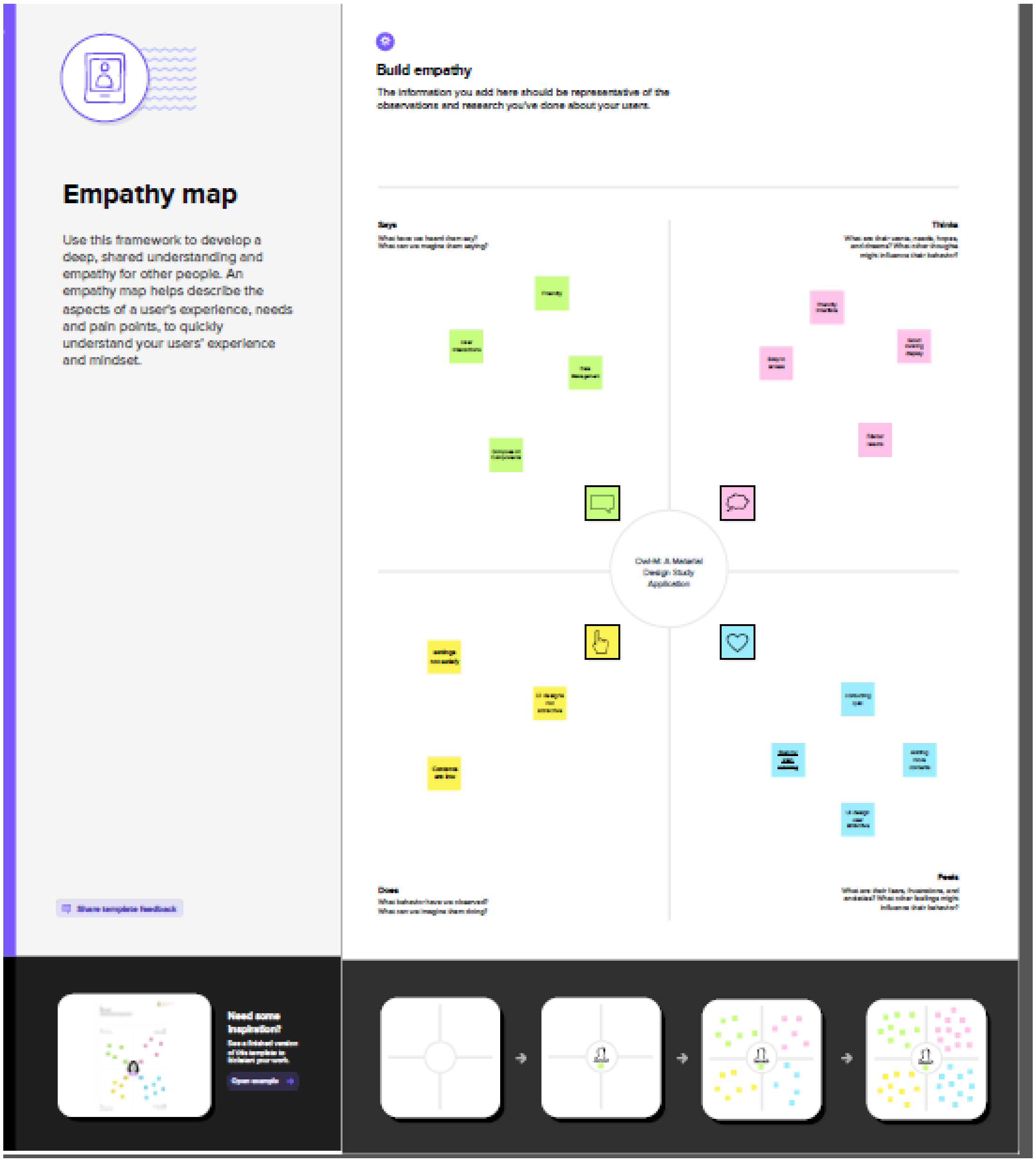
Frequent content sharing

Navigation sidebars for news genre

Headlines and categorization

Offline news reading

Empathy Map:



Empathy map

Use this framework to develop a deep, shared understanding and empathy for other people. An empathy map helps describe the aspects of a user's experience, needs and pain points, to quickly understand your users' experience and mindset.

Need some inspiration?

See a finished version of this template to kickstart your work.

Build empathy

The information you add here should be representative of the observations and research you've done about your users.

Says

What have we heard them say?
What can we imagine them saying?

Thinks

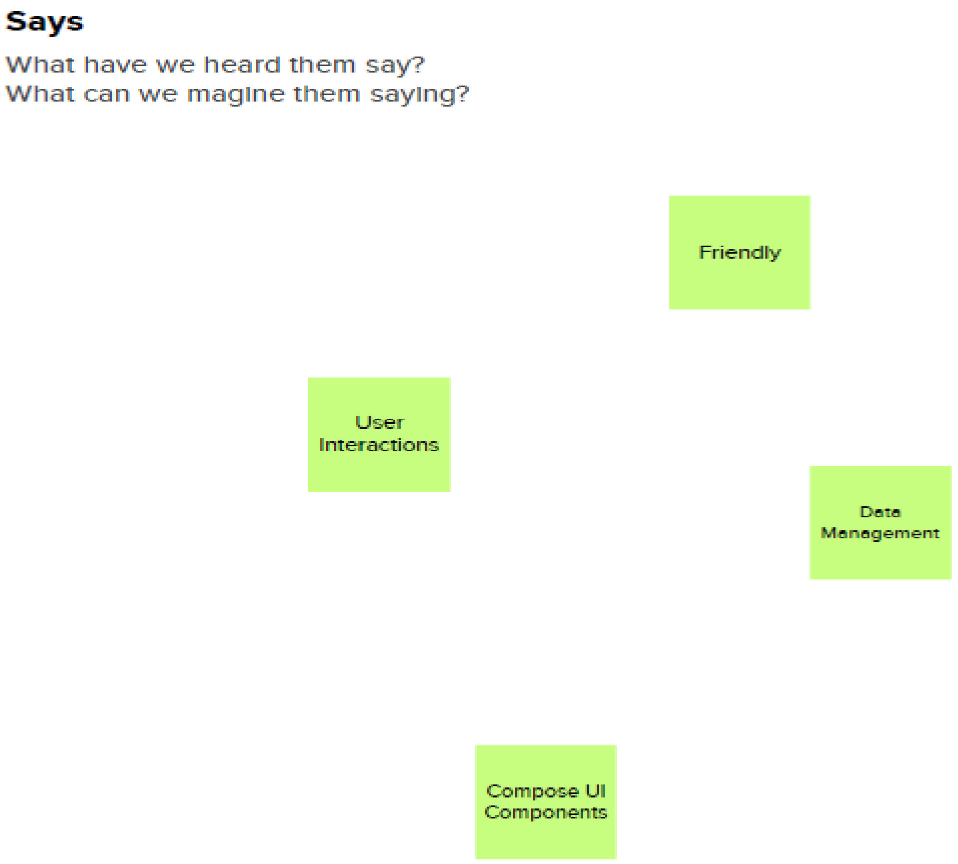
What are their wants, needs, hopes, and dreams? What other thoughts might influence their behavior?

Does

What behavior have we observed?
What can we imagine them doing?

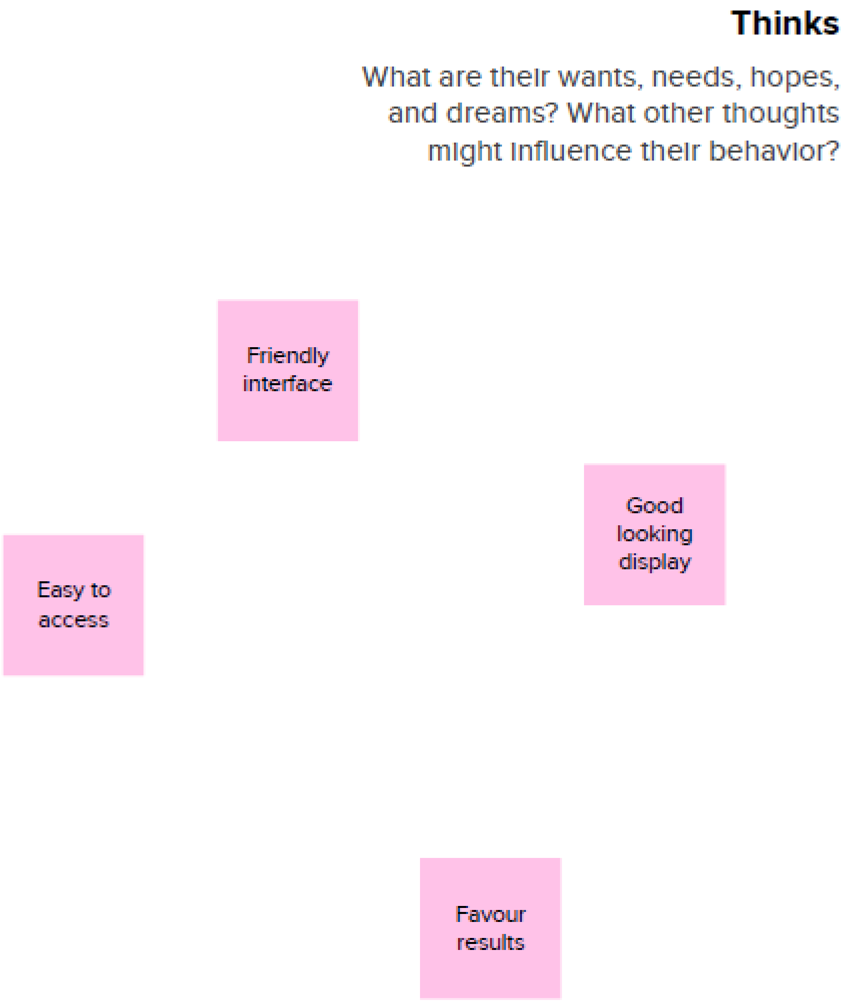
Feels

What are their fears, frustrations, and anxieties? What other feelings might influence their behavior?



Says

What have we heard them say?
What can we magine them saying?



Thinks

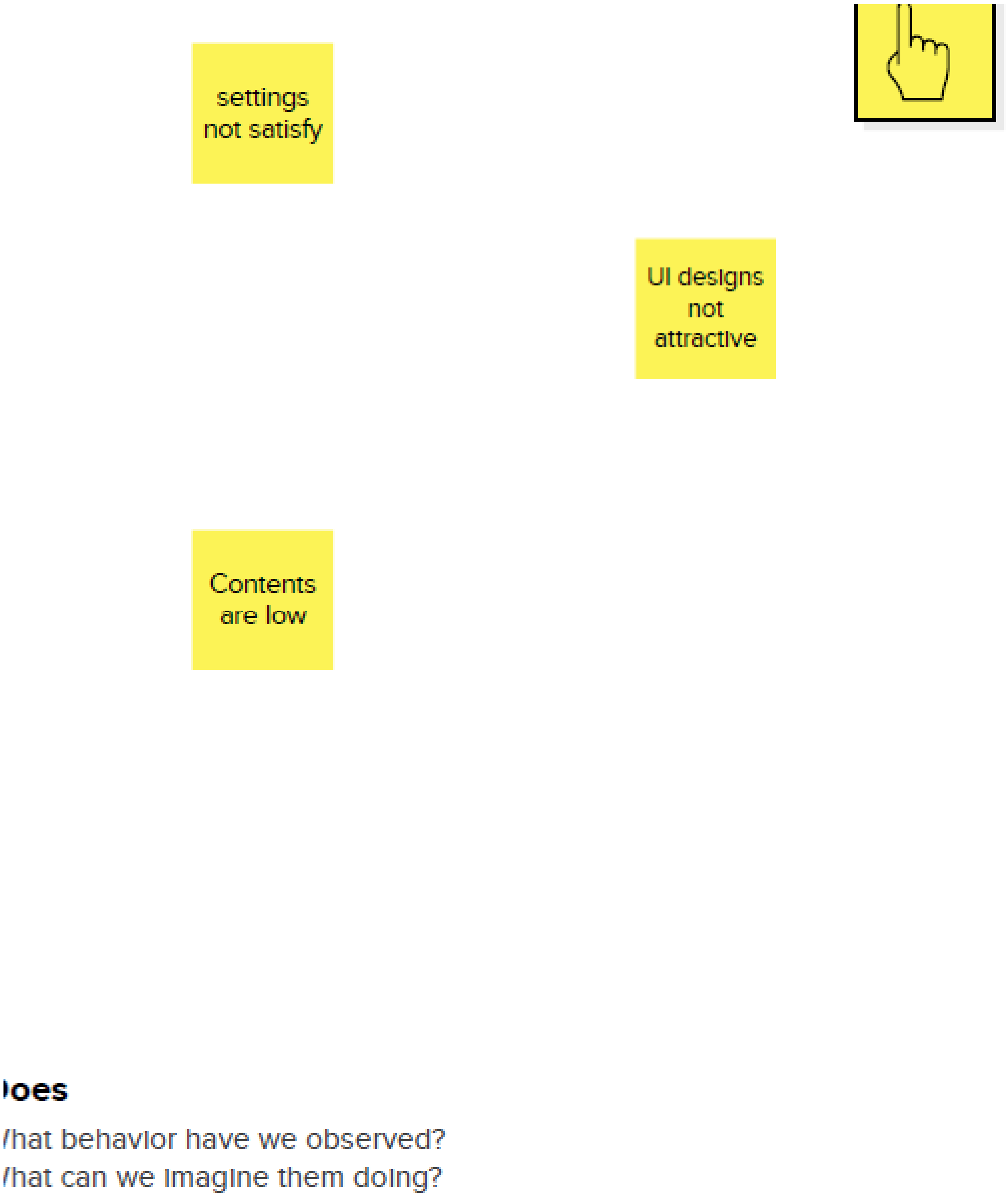
What are their wants, needs, hopes, and dreams? What other thoughts might influence their behavior?

Friendly interface

Good looking display

Easy to access

Favour results



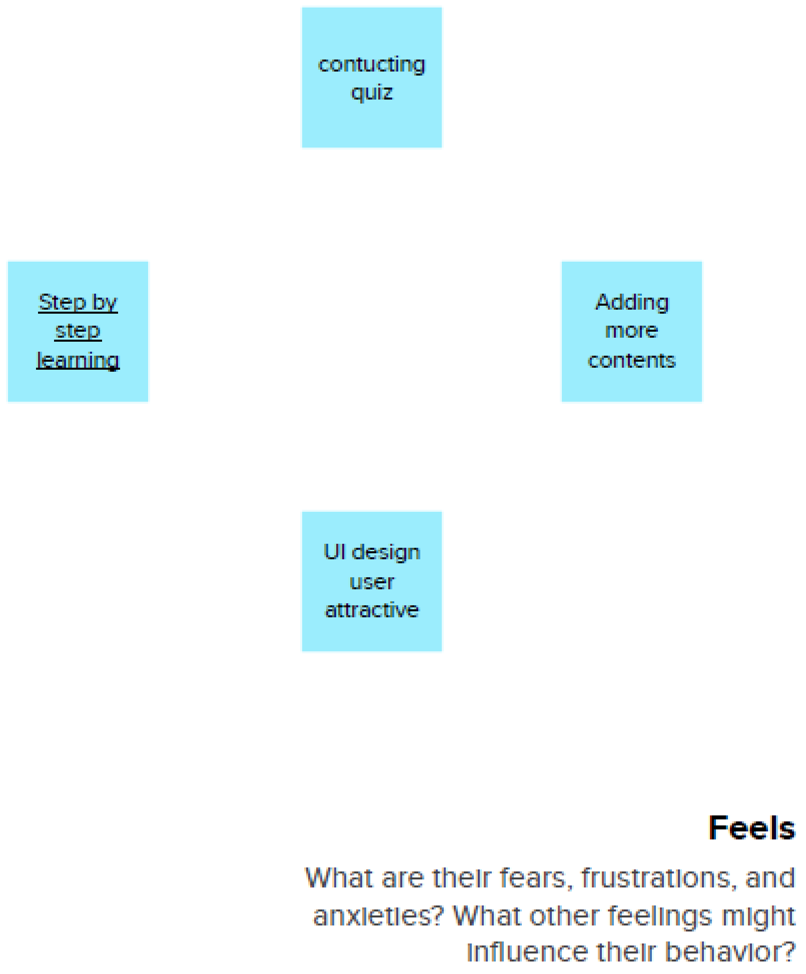
settings not satisfy

UI designs not attractive

Contents are low

oes

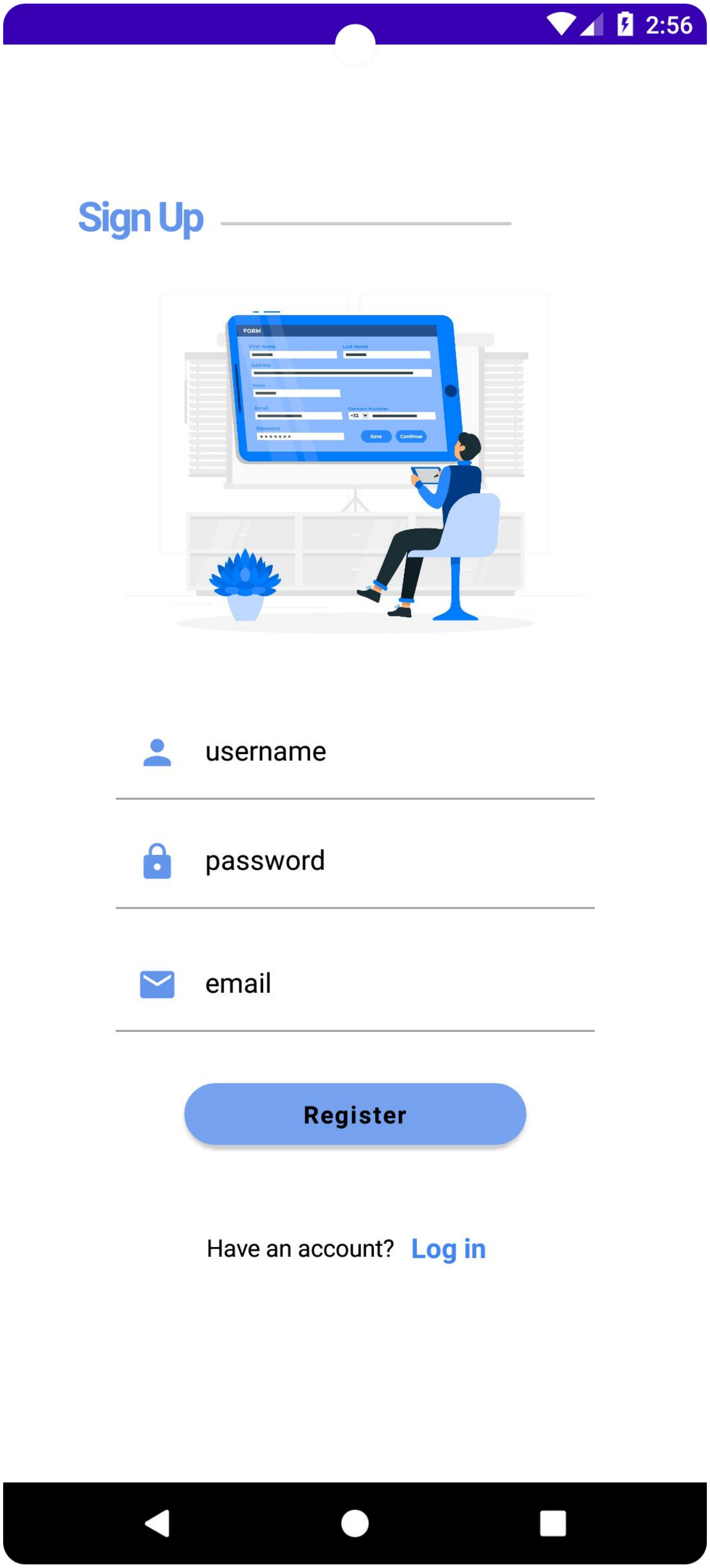
hat behavior have we observed?
hat can we imagine them doing?

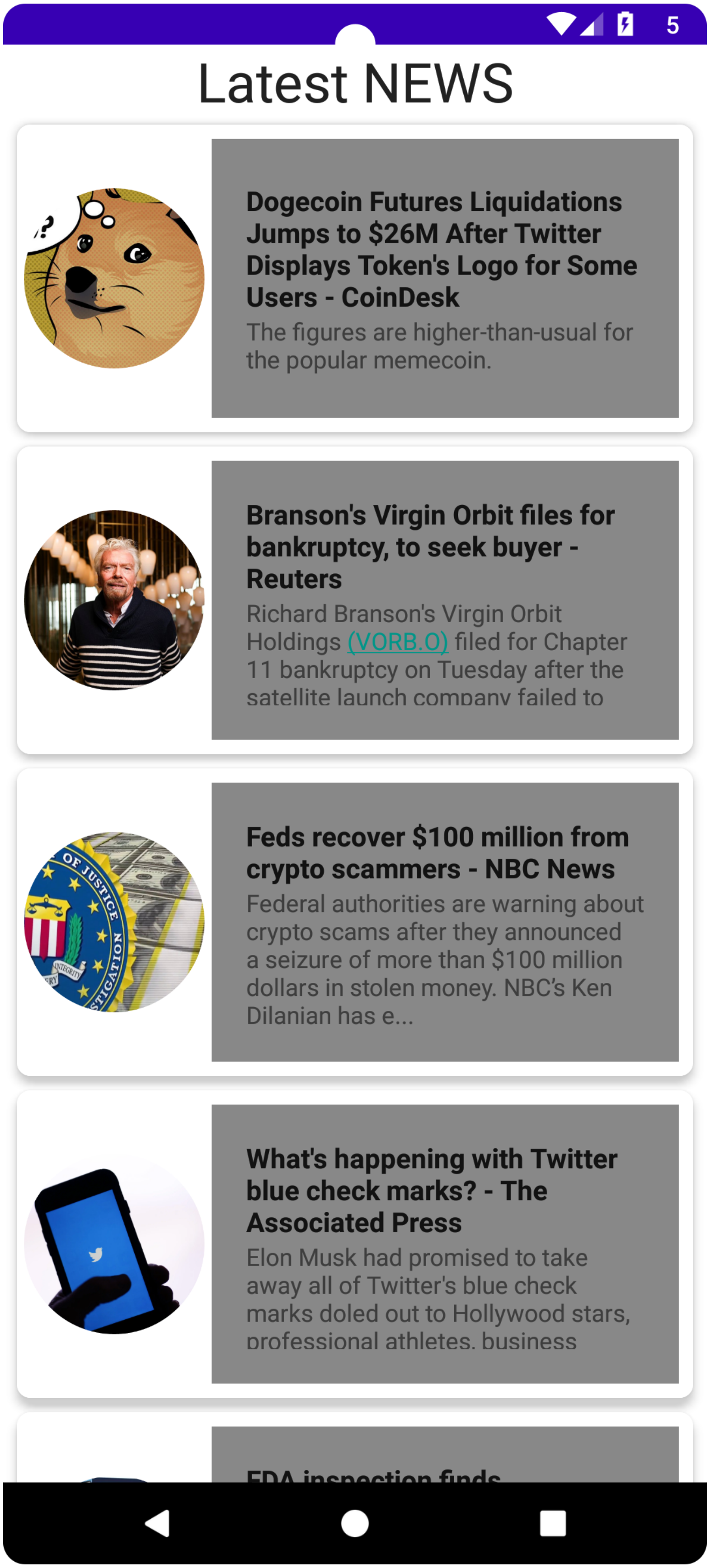


Feels

What are their fears, frustrations, and anxieties? What other feelings might influence their behavior?

Screenshots:


2:56
Sign Up
FORM
username
password
email
Register
Have an account? Log in



Latest NEWS

Dogecoin Futures Liquidations Jumps to $26M After Twitter Displays Token's Logo for Some Users - CoinDesk

The figures are higher-than-usual for the popular memecoin.

Branson's Virgin Orbit files for bankruptcy, to seek buyer - Reuters

Richard Branson's Virgin Orbit Holdings (VORB.O) filed for Chapter 11 bankruptcy on Tuesday after the satellite launch company failed to

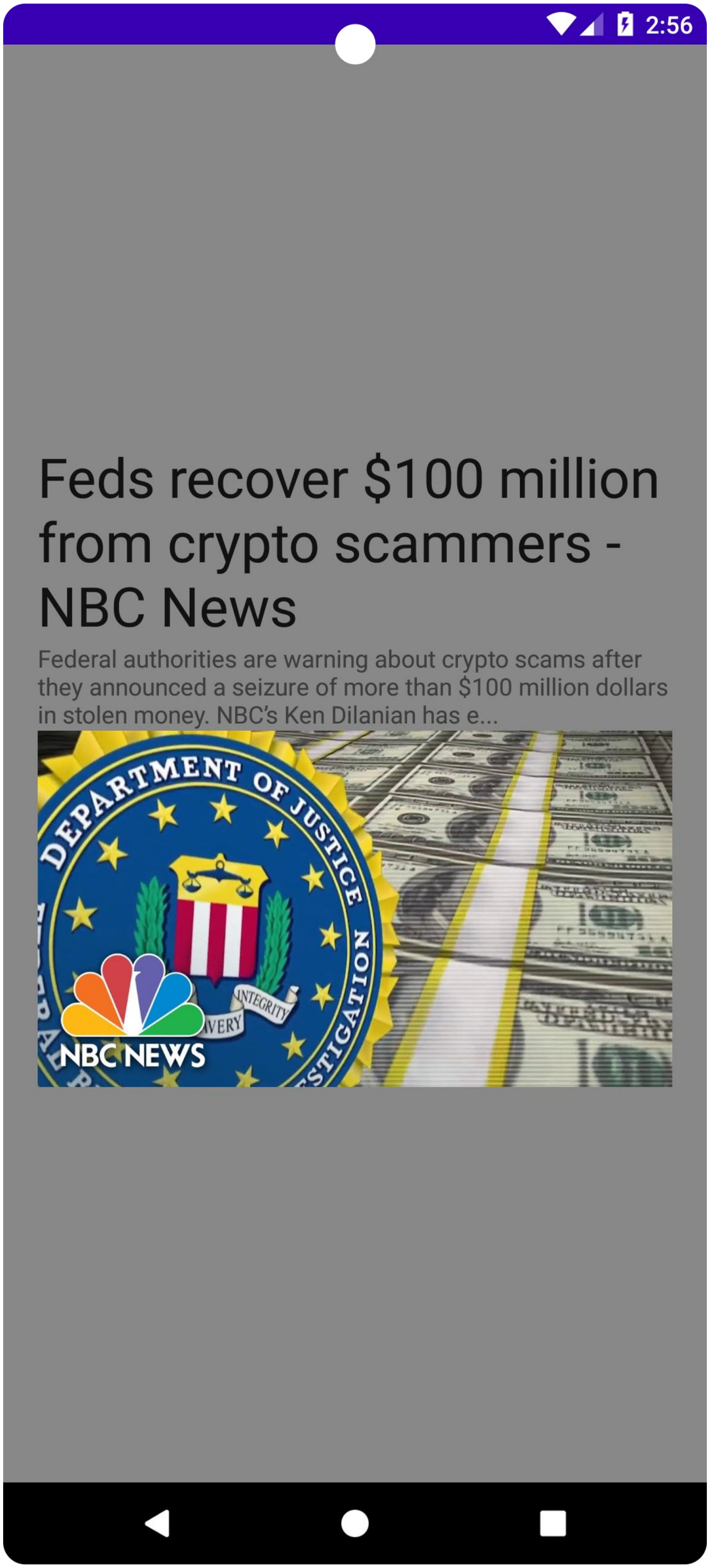
Feds recover $100 million from crypto scammers - NBC News

Federal authorities are warning about crypto scams after they announced a seizure of more than $100 million dollars in stolen money. NBC's Ken Dilanian has e...

What's happening with Twitter blue check marks? - The Associated Press

Elon Musk had promised to take away all of Twitter's blue check marks doled out to Hollywood stars, professional athletes, business

FDA inspection finds



Feds recover $100 million from crypto scammers - NBC News

Federal authorities are warning about crypto scams after they announced a seizure of more than $100 million dollars in stolen money. NBC's Ken Dilanian has e...

4. Advantages:

Convenience: An Android application for keeping up with the latest headlines provides users with the convenience of accessing news on the go. Users can access the latest news articles from their mobile devices, which makes it easier to stay informed about the latest developments.

Personalization: The application allows users to customize their newsfeed by selecting the topics they are interested in, making it easier for them to consume news that is relevant to them.

Reputable news sources: The integration of reputable news sources in the application ensures that users receive accurate and reliable news. This is particularly important in today's era of fake news.

Sharing: The application allows users to share news articles on social media platforms, making it easier for them to share important news stories with their friends and family.

Notifications: The application can provide push notifications for breaking news stories and important events, ensuring that users do not miss out on important news.

Disadvantages:

Overwhelming: The sheer amount of news available on the application can be overwhelming for some users, especially if they are not accustomed to consuming news regularly.

Bias: Despite the integration of reputable news sources, some users may perceive the news presented on the application to be biased or one-sided.

Connectivity: The application requires an internet connection to access the latest news articles. Users who do not have a stable internet connection may find it difficult to use the application.

Privacy concerns: Some users may be concerned about their privacy when using the application, particularly if the application requires access to their personal information.

Addiction: Some users may become addicted to consuming news, which can lead to information overload and affect their mental health if they do not take breaks.

5.APPLICATIONS

Latest news headline app have a wide range of applications, some of which include:

1. Keeping up with the latest news: One of the main applications of news headline apps is to stay up-to-date with the latest news and current events. Users can customize their news feed based on their interests and preferences, ensuring they receive news stories that are relevant to them.

2. Breaking news alerts: Many news headline apps offer push notifications and breaking news alerts, so users can stay informed about major news stories as they happen.

3. Personalized news recommendations: News headline apps often use machine learning and AI algorithms to recommend news stories based on a user's reading history and interests.

4. In-depth analysis and commentary: Some news headline apps provide in-depth analysis and commentary on news stories, offering users a more detailed understanding of complex issues.

5. Bookmark and save news stories: News headline apps allow users to save news stories and articles for later reading or reference.

6. Discover new topics and interests: News headline apps can introduce users to new topics and interests they may not have been aware of before, broadening their knowledge and understanding of the world.

7. Research and fact-checking: News headline apps can be used for research and fact-checking, helping users to verify information and sources.

8. Follow specific journalists and news sources: Users can follow specific journalists and news sources on news headline apps, allowing them to receive updates on their reporting and analysis.

Overall, news headline apps offer a convenient and efficient way for users to stay informed about the latest news and events, and to explore a wide range of topics and interests.

6. CONCLUSION:

An Android application for keeping up with the latest headlines is a valuable tool for anyone who wants to stay informed about the latest news and events from around the world. The application provides users with a convenient and personalized way to access news, allowing them to customize their newsfeed and filter out irrelevant news articles.

The integration of reputable news sources ensures that users receive accurate and reliable news, while the sharing and notification features make it easier for users to stay connected with their friends and family and stay up-to-date with breaking news.

However, there are also potential drawbacks to using such an application, including overwhelming amounts of news, potential bias, privacy concerns, and addiction to consuming news.

Overall, an Android application for keeping up with the latest headlines is a valuable tool that can help users stay informed about the latest developments in the world. To ensure the best experience, it's important for users to use the application responsibly and to take breaks to avoid information overload.

FUTURE SCOPE:

An Android application for keeping up with the latest headlines has a vast scope in the future. Here are some potential areas where the application can expand:

1. Personalization: The application can offer personalized news feeds based on users' interests, search history, and reading patterns. It can provide a tailored experience to each user, increasing engagement and retention.

2. Multimedia Content: The application can incorporate multimedia content like videos, podcasts, and infographics, making news consumption more engaging and immersive.

3. Location-based News: The application can provide news based on a user's location, making it easier for them to keep up with local events and news.

4. Social Media Integration: The application can integrate with social media platforms like Twitter and Facebook, allowing users to share news articles and receive news recommendations from their social network.

5. Artificial Intelligence: The application can use Artificial Intelligence and Machine Learning algorithms to provide personalized news summaries, article recommendations, and even predict future trends.

6. News Aggregation: The application can aggregate news from various sources, including newspapers, blogs, and social media platforms, providing a comprehensive view of the news landscape.

7. Blockchain: The application can leverage blockchain technology to provide transparency and accuracy in news reporting, preventing fake news and improving credibility.

In conclusion, an Android application for keeping up with the latest headlines has immense potential for future growth and can offer a personalized and immersive news consumption experience to users.

7.APPENDIX

Main Page:

```
<?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>
```

Login Page : \

```
<?xml version="1.0" encoding="utf-8"?>
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="108dp"
    android:height="108dp"
    android:viewportWidth="108"
    android:viewportHeight="108">
    <path
        android:fillColor="#3DDC84"
        android:pathData="M0,0h108v108h-108z" />
    <path
        android:fillColor="#00000000"
        android:pathData="M9,0L9,108"
        android:strokeWidth="0.8"
        android:strokeColor="#33FFFFFF" />
    <path
        android:fillColor="#00000000"
        android:pathData="M19,0L19,108"
        android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M29,0L29,108"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M39,0L39,108"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M49,0L49,108"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M59,0L59,108"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
```

```
<path
    android:fillColor="#00000000"
    android:pathData="M69,0L69,108"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M79,0L79,108"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M89,0L89,108"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M99,0L99,108"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
```

```
    android:fillColor="#00000000"

    android:pathData="M0,9L108,9"

    android:strokeWidth="0.8"

    android:strokeColor="#33FFFFFF" />

<path

    android:fillColor="#00000000"

    android:pathData="M0,19L108,19"

    android:strokeWidth="0.8"

    android:strokeColor="#33FFFFFF" />

<path

    android:fillColor="#00000000"

    android:pathData="M0,29L108,29"

    android:strokeWidth="0.8"

    android:strokeColor="#33FFFFFF" />

<path

    android:fillColor="#00000000"

    android:pathData="M0,39L108,39"

    android:strokeWidth="0.8"

    android:strokeColor="#33FFFFFF" />

<path

    android:fillColor="#00000000"
    android:pathData="M0,49L108,49"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,59L108,59"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,69L108,69"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,79L108,79"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,89L108,89"
    android:strokeWidth="0.8"

    android:strokeColor="#33FFFFFF" />

<path

    android:fillColor="#00000000"

    android:pathData="M0,99L108,99"

    android:strokeWidth="0.8"

    android:strokeColor="#33FFFFFF" />

<path

    android:fillColor="#00000000"

    android:pathData="M19,29L89,29"

    android:strokeWidth="0.8"

    android:strokeColor="#33FFFFFF" />

<path

    android:fillColor="#00000000"

    android:pathData="M19,39L89,39"

    android:strokeWidth="0.8"

    android:strokeColor="#33FFFFFF" />

<path

    android:fillColor="#00000000"

    android:pathData="M19,49L89,49"

    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M19,59L89,59"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M19,69L89,69"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M19,79L89,79"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M29,19L29,89"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
```

```
<path
    android:fillColor="#00000000"
    android:pathData="M39,19L39,89"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M49,19L49,89"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M59,19L59,89"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M69,19L69,89"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
```

```
        android:fillColor="#00000000"

        android:pathData="M79,19L79,89"

        android:strokeWidth="0.8"

        android:strokeColor="#33FFFFFF" />

</vector>
```

News:

```
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 screen : 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)

}
```

Main Page :

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) }

        )

    }

}
```

Register Page :

```
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)
}
```

Data base room : package com.example.newsheadlines

```
import android.content.Context

import androidx.room.Database

import androidx.room.Room

import androidx.room.RoomDatabase


@Database(entities = [User::class], version = 1)

abstract class UserDatabase : RoomDatabase() {


    abstract fun userDao(): UserDao


    companion object {


        @Volatile

        private var instance: UserDatabase? = null


        fun getDatabase(context: Context): UserDatabase {

            return instance ?: synchronized(this) {

                val newInstance = Room.databaseBuilder(

                    context.applicationContext,

                    UserDatabase::class.java,

                    "user_database"
                ).build()
                instance = newInstance
                newInstance
            }
        }
    }
}
```

Data base :

```
 package com.example.newsheadlines


import android.annotation.SuppressLint
import android.content.ContentValues
import android.content.Context
import android.database.Cursor
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper


class UserDatabaseHelper(context: Context) :
    SQLiteOpenHelper(context, DATABASE_NAME, null, DATABASE_VERSION) {
```

```
companion object {

    private const val DATABASE_VERSION = 1

    private const val DATABASE_NAME = "UserDatabase.db"



    private const val TABLE_NAME = "user_table"

    private const val COLUMN_ID = "id"

    private const val COLUMN_FIRST_NAME = "first_name"

    private const val COLUMN_LAST_NAME = "last_name"

    private const val COLUMN_EMAIL = "email"

    private const val COLUMN_PASSWORD = "password"

}


override fun onCreate(db: SQLiteDatabase?) {

    val createTable = "CREATE TABLE $TABLE_NAME (" +

            "$COLUMN_ID INTEGER PRIMARY KEY AUTOINCREMENT, " +

            "$COLUMN_FIRST_NAME TEXT, " +

            "$COLUMN_LAST_NAME TEXT, " +

            "$COLUMN_EMAIL TEXT, " +

            "$COLUMN_PASSWORD TEXT" +

            ")"
    db?.execSQL(createTable)

}



override fun onUpgrade(db: SQLiteDatabase?, oldVersion: Int, newVersion: Int) {

    db?.execSQL("DROP TABLE IF EXISTS $TABLE_NAME")

    onCreate(db)

}



fun insertUser(user: User) {

    val db = writableDatabase

    val values = ContentValues()

    values.put(COLUMN_FIRST_NAME, user.firstName)

    values.put(COLUMN_LAST_NAME, user.lastName)

    values.put(COLUMN_EMAIL, user.email)

    values.put(COLUMN_PASSWORD, user.password)

    db.insert(TABLE_NAME, null, values)

    db.close()

}



@SuppressLint("Range")

fun getUserByUsername(username: String): User? {
        val db = readableDatabase
        val cursor: Cursor = db.rawQuery("SELECT * FROM $TABLE_NAME WHERE $COLUMN_FIRST_NAME = ?", arrayOf(username))
        var user: User? = null
        if (cursor.moveToFirst()) {
            user = User(
                id = cursor.getInt(cursor.getColumnIndex(COLUMN_ID)),
                firstName = cursor.getString(cursor.getColumnIndex(COLUMN_FIRST_NAME)),
                lastName = cursor.getString(cursor.getColumnIndex(COLUMN_LAST_NAME)),
                email = cursor.getString(cursor.getColumnIndex(COLUMN_EMAIL)),
                password = cursor.getString(cursor.getColumnIndex(COLUMN_PASSWORD)),
            )
        }
        cursor.close()
        db.close()
        return user
    }
    @SuppressLint("Range")
    fun getUserById(id: Int): User? {
        val db = readableDatabase
        val cursor: Cursor = db.rawQuery("SELECT * FROM $TABLE_NAME WHERE $COLUMN_ID = ?", arrayOf(id.toString()))
    var user: User? = null
    if (cursor.moveToFirst()) {
        user = User(
            id = cursor.getInt(cursor.getColumnIndex(COLUMN_ID)),
            firstName = cursor.getString(cursor.getColumnIndex(COLUMN_FIRST_NAME)),
            lastName = cursor.getString(cursor.getColumnIndex(COLUMN_LAST_NAME)),
            email = cursor.getString(cursor.getColumnIndex(COLUMN_EMAIL)),
            password = cursor.getString(cursor.getColumnIndex(COLUMN_PASSWORD)),
        )
    }
    cursor.close()
    db.close()
    return user
}


@SuppressLint("Range")
fun getAllUsers(): List<User> {
    val users = mutableListOf<User>()
    val db = readableDatabase
    val cursor: Cursor = db.rawQuery("SELECT * FROM $TABLE_NAME", null)
    if (cursor.moveToFirst()) {
            do {
                val user = User(
                    id = cursor.getInt(cursor.getColumnIndex(COLUMN_ID)),
                    firstName = cursor.getString(cursor.getColumnIndex(COLUMN_FIRST_NAME)),
                    lastName = cursor.getString(cursor.getColumnIndex(COLUMN_LAST_NAME)),
                    email = cursor.getString(cursor.getColumnIndex(COLUMN_EMAIL)),
                    password = cursor.getString(cursor.getColumnIndex(COLUMN_PASSWORD)),
                )
                users.add(user)
            } while (cursor.moveToNext())
        }
        cursor.close()
        db.close()
        return users
    }


}
```