# An Android Application for Keeping Up with the Latest Headlines

# INTRODUCTION

## 1.1 OVERVIEW

A headline's purpose is to quickly and briefly draw attention to the story. It is generally written by a copy editor, but may also be written by the writer, the page layout designer, or other editors.

Research in 1980 classified news headlines into four broad categories: commands, statements, and explanations. Advertisers and marketers classify advertising headlines slightly differently into questions, commands, benefits, news/information, and provocation.

A study indicates there has been a substantial increase of sentiment negativity and decrease of emotional neutrality in headlines across written popular U.S.-based news media since 2000.

Skilled newspaper readers "spend most of their reading time scanning the headlines—rather than reading [all or most of] the stories".

Headlines can bias readers toward a specific interpretation and readers struggle to update their memory in order to correct initial misconceptions in the cases of misleading or inappropriate one headline

One approach investigated as a potential countermeasure to online misinformation is "attaching warnings to headlines of news stories that have been disputed by third-party fact-checkers", albeit its potential problems include e.g. that false headlines that fail to get tagged are considered validated by readers

Headlines are becoming increasingly important by reading a list of search engine results as by scanning a newspaper page.

Headlines should be clear and specific, telling the reader what the story is about, and be interesting enough to draw them into reading the article.

in the internet age. Not only do they capture the reader's attention, they serve as source material for search engines. Today a reader is just as likely to come across an article

## 1.2 PURPOSE

The title above a story in a newspaper, magazine or newsletter is called a headline, or "head" ("head") in print journalism, or a "heading" in online pages. It has the same function in mass media writing as a lead, to call attention to the story, to snare people in.

It attracts the attention of the readers, holds their interest, and tells them about the story. A headline should: o Attract the reader's attention, o Summarize the story, o Depict the mood of the story, o Help set the tone of the newspapers, and Provide adequate typographic relief.

## Headline can perform four different tasks:

- Get attention.
- Select the audience.
- Deliver a complete message.
- Draw the reader into the body copy.

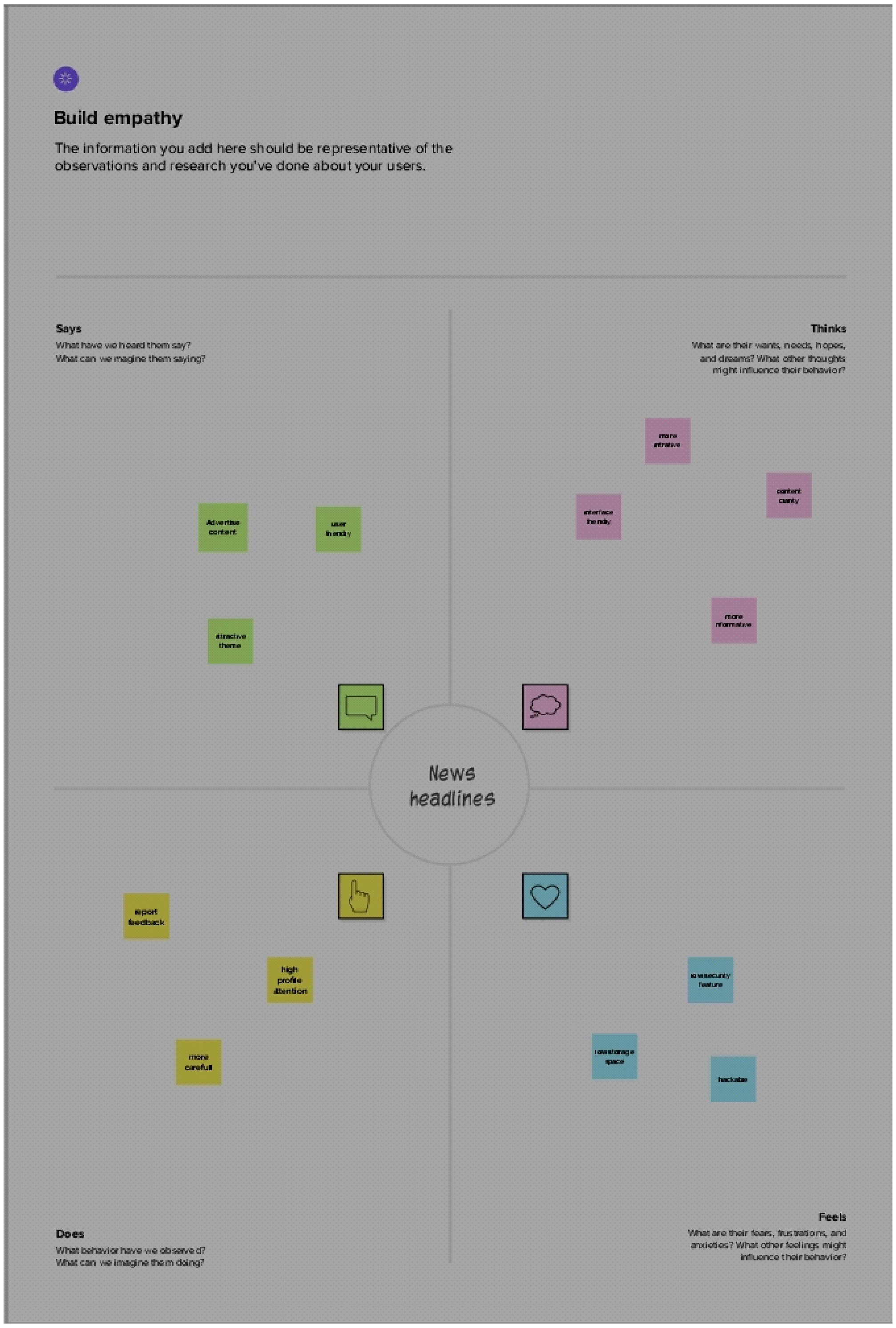
What are power words for headlines? Power words for headlines are words with emotional meanings that resonate with readers and convince them to read or engage with your content. Headlines refer to the main title or the subheads for online content like blogs, social media posts and email newsletters

An ad headline is a group of words or sentences that promote a product or service. A headline is typically in large font and used to grab the attention of the reader. A lot of elements go into creating a powerful ad headline that you should keep in mind before launching a campaign.

Its focus is vague. Context is absent. You are reading a news report of some form, but you are not reading a news story. Headlines introduce, frame, and contextualize. They award significance, communicate gravitas, and reinforce status. Headlines inform and misinform. They are a crucial part of how news turns into a story. In the long term, they play a part in how stories are retold and recorded, thus eventually turning into memories and histories.

## 2 Problem definition & Design Thinking

### 2.1 Empathy Map:


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?

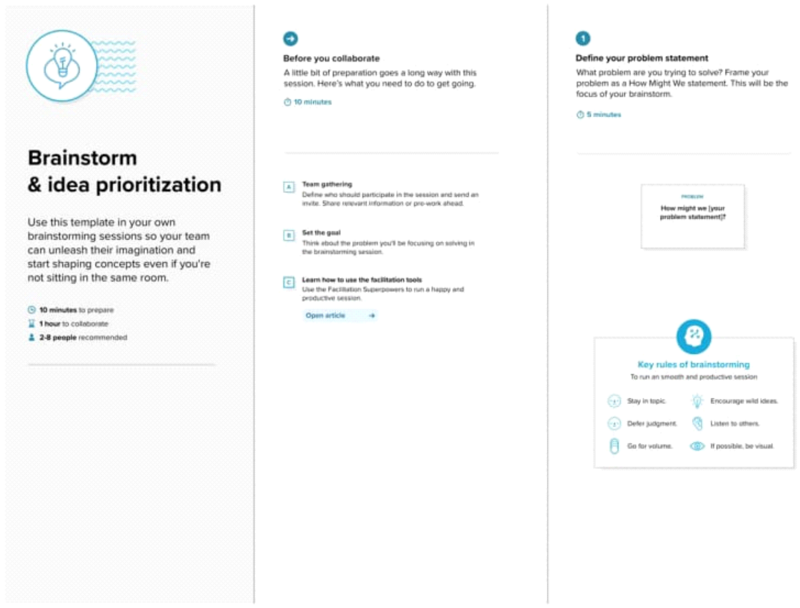
Thinks
What are their wants, needs, hopes, and dreams? What other thoughts might influence their behavior?

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?

## 2.2 Ideation & Brainstorming Map

Step-1: Team Gathering, Collaboration and Select the Problem Statement



Brainstorm
& idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

**10 minutes** to prepare
**1 hour** to collaborate
**2-8 people** recommended

**Before you collaborate**
A little bit of preparation goes a long way with this session. Here's what you need to do to get going.
10 minutes

A **Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B **Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.

C **Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.

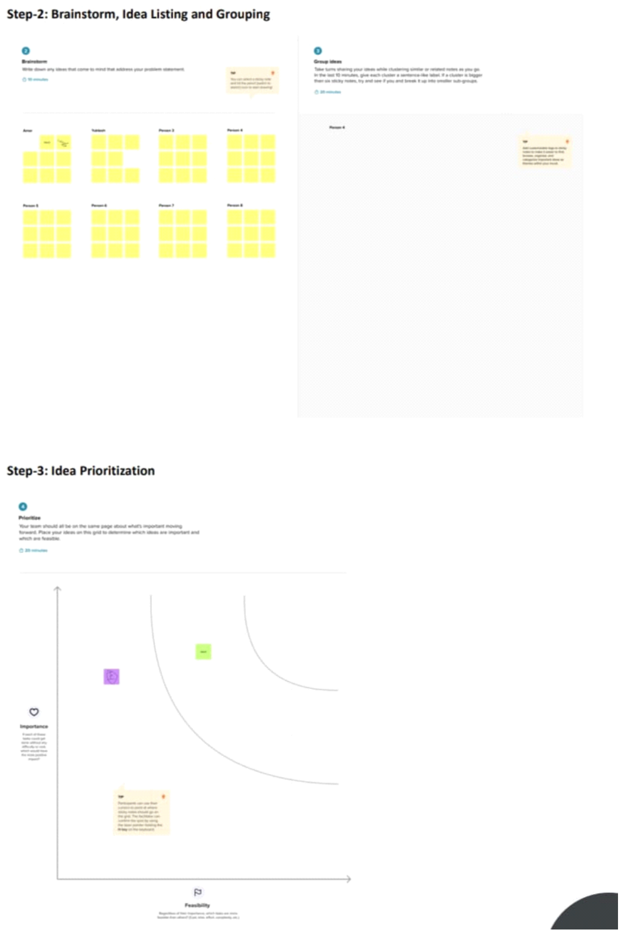
Open article →

1 **Define your problem statement**
What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.
5 minutes

PROBLEM
**How might we [your problem statement]?**

**Key rules of brainstorming**
To run an smooth and productive session

| | |
|---|---|
| Stay in topic. | Encourage wild ideas. |
| Defer judgment. | Listen to others. |
| Go for volume. | If possible, be visual. |


Step-2: Brainstorm, Idea Listing and Grouping

Brainstorm

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

10 minutes

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

Step-3: Idea Prioritization

Prioritize

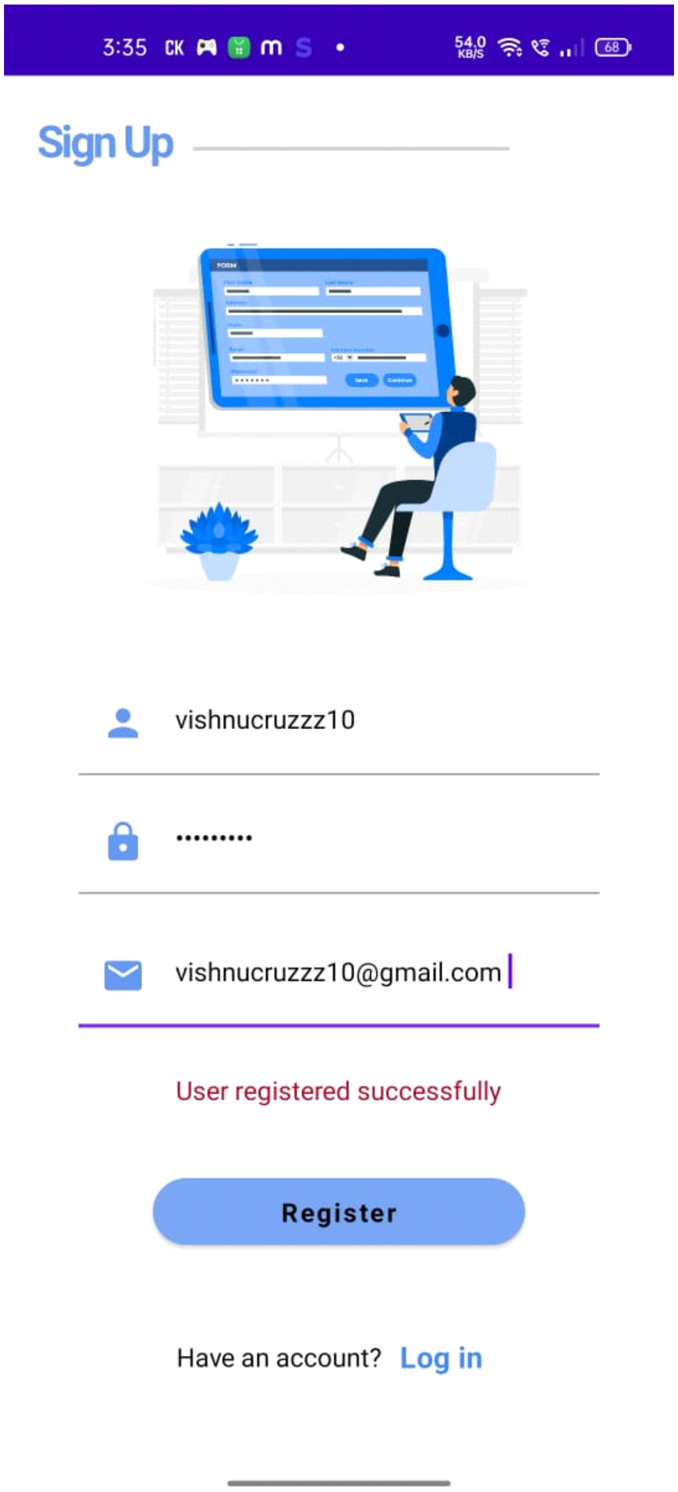
Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

# 3. RESULT

## Final Output of the application:

## Admin Page:



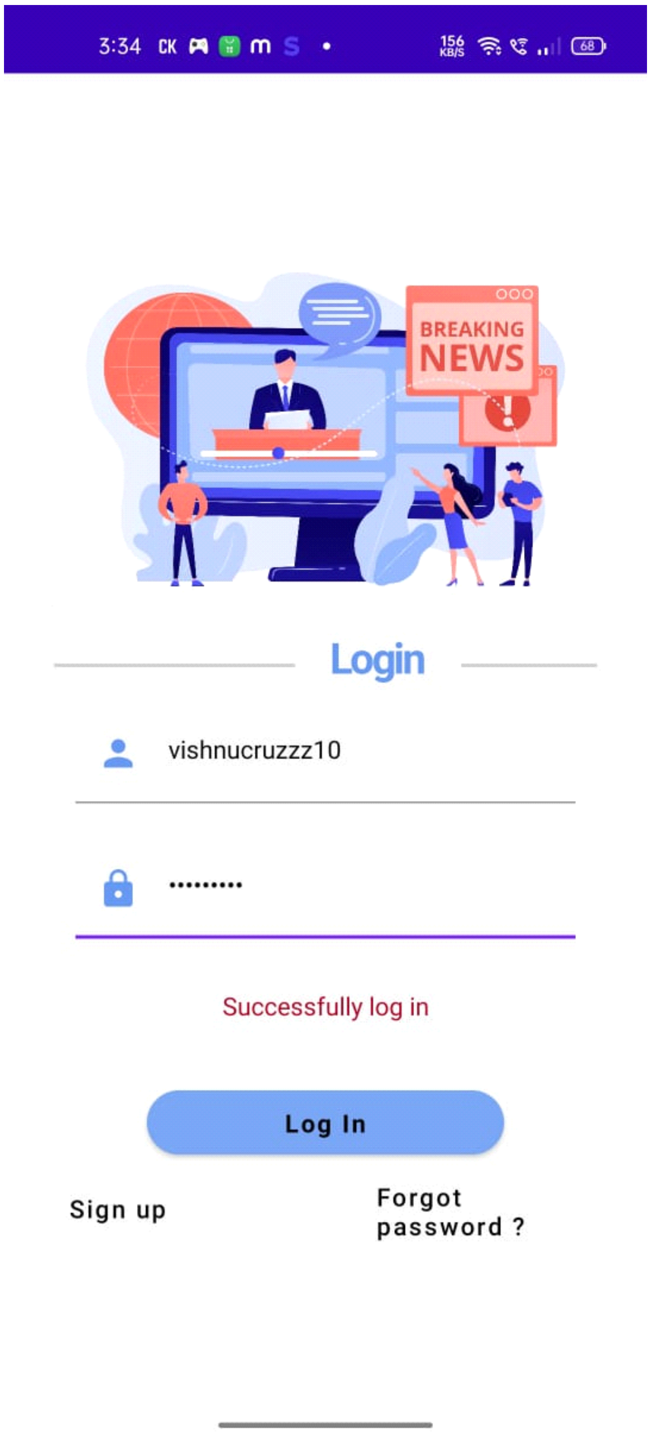
vishnucruzzz10

•••••••••

vishnucruzzz10@gmail.com

User registered successfully

## Login Page:



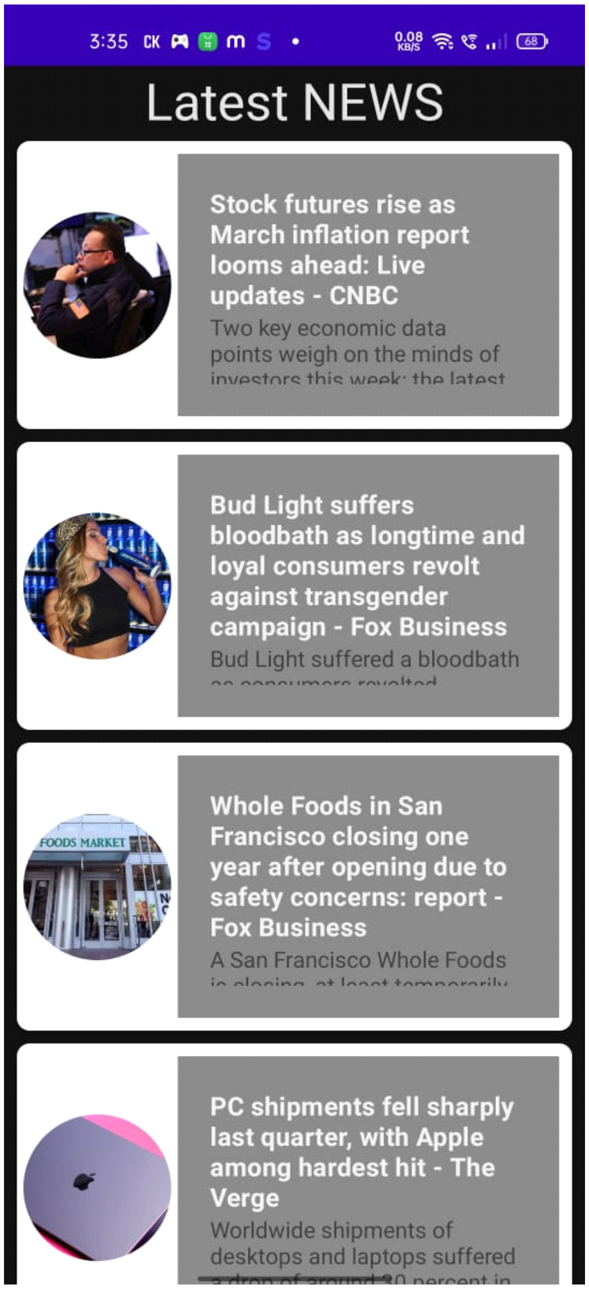
Login

vishnucruzzz10

•••••••••

Successfully log in

**Main page:**


3:35

Latest NEWS

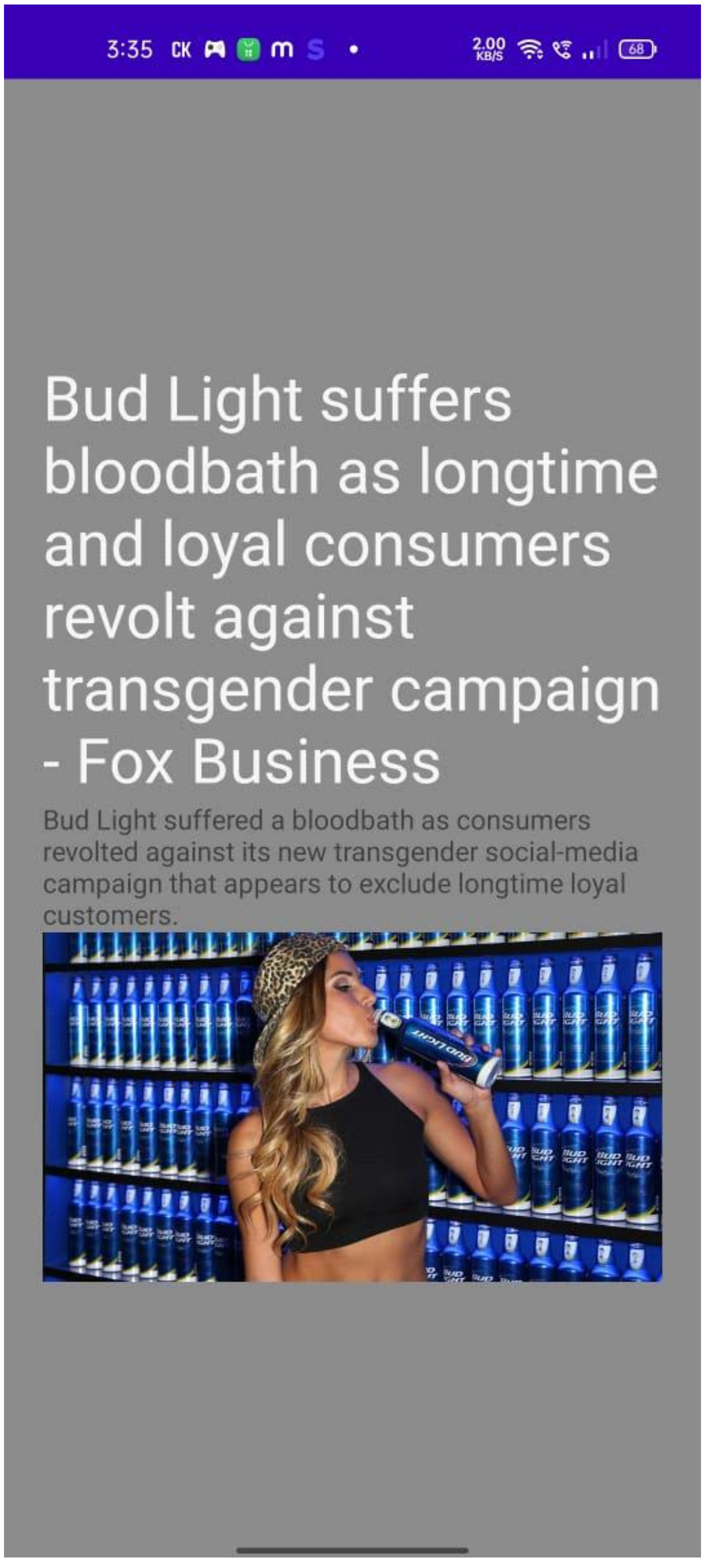
**Stock futures rise as March inflation report looms ahead: Live updates - CNBC**
Two key economic data points weigh on the minds of

**Bud Light suffers bloodbath as longtime and loyal consumers revolt against transgender campaign - Fox Business**
Bud Light suffered a bloodbath

**Whole Foods in San Francisco closing one year after opening due to safety concerns: report - Fox Business**
A San Francisco Whole Foods

**PC shipments fell sharply last quarter, with Apple among hardest hit - The Verge**
Worldwide shipments of desktops and laptops suffered

**Output:**


3:35

Bud Light suffers bloodbath as longtime and loyal consumers revolt against transgender campaign - Fox Business

Bud Light suffered a bloodbath as consumers revolted against its new transgender social-media campaign that appears to exclude longtime loyal customers.

# 4 ADVANTAGES & DISADVANTAGES

## ADVANTAGES

- It needs to capture more attention of the reader without, ideally, compromising the essence of the story.
- Headlines not only lure attention of the creator but it also needs to get the direct attention from the viewer.
- It provides information and knowledge about the specific news by only viewing the headlines.
- It must be well shortened form so that everyone must understand the content of the news by viewing the headlines.
- There are also variety of news widely spread ll/.our entire world and to cover the specified news, we can personalised the according to the mood and wish of your need.
- The words bring the crucial elements of the story to the forefront and the grammar if needed.
- Framing a news headlines depends the overall popularity of the news the includes whether the news is global or national, if certain steps needed to be done before preparing the news headlines.
- We can access the news headlines and its detailed information from any part of the world, the news

headlines mainly shows the regular news happens daily in our country.

- Though there are still people who only see news headlines but still there are many views, the news by reading the news headlines.

## DISADVANTAGES

- The major disadvantage is that the not every news headlines can gain more popularity as the previous headlines; it will change according to time and popularity.
- People of this era do not view news headlines even though it is important to know the day to day news about our surroundings and environment.
- Creators also loss their interest because of the news headlines not reaching to others and only receives partial viewers all around the world.
- Day by day the people loss their interest in viewing news headlines because the people does not allocate time for viewing the news headlines.
- It will leads to more loss in interest for creating the news headlines by the developing teams of the news headlines and so on.
- It also rely too heavily personalities, emotions and opinions.

- It can short-change complex stories or avoid them altogether.
- There are many companies that comes in the news headline platform so it can bring down many of the third party content creators and other part time workers as they spend less time in creating the same headlines altogether.

## 5 APPLICATIONS

- This is widely used in spreading information's all over the world and it will helps the victims to gain more attention in compare to regular way of approaching the news all by ourselves.
- The capability of reaching the news headlines is very high in compare to other ways of spreading the information's widely.
- The news headlines can serve a variety of functions, including story summarization, interest generation, immediacy satisfaction, and attention direction.
- The news headlines also may function to generate interest in a story.
- The main functions of news headlines are not mutually exclusive. Many are created to grab attention from the viewer and to get personalized by the viewers all over the world.

- The dominant news headlines used by the traditional media outlets, particularly by the news headlines are the results of changes in the form of news during the 20th century.
- News headlines designs integrated more white space with pictures; at the same time headline occupy the major space of the news. Because it is one of the key viewing formula for a news content.
- Changes in information or update of every news in regarding to the previous news must be provided by the creator to know each and every segment of news regarding to situations of the specific country from distance.

## 6 conclusions

- From the android app we can understand that every single action in app requires well refined and organised codes for briefly understanding the overall explanation about the app.
- It provides interest in viewing the news headlines app for well-researched information with the logical proof of being happened.
- It will summarize the overall outlet of a news headline app which we use for our day to day life for knowing

the information far and wide with the help of your android app.

- Headlines and stories that begin with most news worthy information first satisfy immediacy needs.
- They aim to be maximally efficient for the readers with the little time for the news.
- One concern with the headlines that attempts to direct attention is the possibility that the main context or understanding of news events could become distorted.
- We push the news headlines to its limit to give a better view for the viewer, sharper in contrast, crystal clear news information with a true logical proof about the news content.
- Academic researchers approach their unique characteristics way separately for the better viewer experience and well defined android app.

# 7 FUTURE SCOPES

- News remains an important part of what people do on their mobile devices—64% of tablet owners and 62% of smartphone owners say they use the devices for news at least weekly, tying news statistically with other popular activities such email and playing games on tablets and behind only email on smartphones.
- Mobile users, moreover, are not just checking headlines on their devices, although nearly all use the devices for the latest new updates.

- Many also are reading longer news stories – 73% of adults who consume news on their tablet read in-depth articles at least sometimes, including 19% who do so daily. Fully 61% of smartphone news consumers at least sometimes read longer stories, 11% regularly.
- Mobile technology allows people to get news anywhere, and any time, most people get news on these devices when they are at home—and roughly half of mobile news users get news on their device just once a day.
- These people tend to be more engaged news users than those who get news on just one device. They are more likely to read deeply (fully 82% sometimes or regularly read in depth articles on their tablet compared with 62% of those who get news on just the tablet), to send or receive news through email or social networks and to read past issues of magazines.

# 8 APPENDIXES

## **User class**:

```
package com.example.newsheadlines


import androidx.room.ColumnInfo
import androidx.room.Entity

import androidx.room.PrimaryKey


@Entity(tableName = "user_table")

data class User(

    @PrimaryKey(autoGenerate = true) val id: Int?,

    @ColumnInfo(name = "first_name") val firstName:
String?,

    @ColumnInfo(name = "last_name") val lastName:
String?,

    @ColumnInfo(name = "email") val email: String?,

    @ColumnInfo(name = "password") val password:
String?,


    )
```

## **UserDao interface**:

```
package com.example.newsheadlines


import androidx.room.*
```

```
@Dao
interface UserDao {

    @Query("SELECT * FROM user_table WHERE email
= :email")
    suspend fun getUserByEmail(email: String): User?

    @Insert(onConflict = OnConflictStrategy.REPLACE)
    suspend fun insertUser(user: User)

    @Update
    suspend fun updateUser(user: User)

    @Delete
    suspend fun deleteUser(user: User)
}
```

## **User Database class**:

```
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
            }
        }
    }
}
```

## User Database Helper:

```
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_FIRS
T_NAME)),

                lastName =
cursor.getString(cursor.getColumnIndex(COLUMN_LAST
_NAME)),

                email =
cursor.getString(cursor.getColumnIndex(COLUMN_EMAI
L)),

                password =
cursor.getString(cursor.getColumnIndex(COLUMN_PASS
WORD)),

            )

        }

        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_FIRS
T_NAME)),

                lastName =
cursor.getString(cursor.getColumnIndex(COLUMN_LAST
_NAME)),

                email =
cursor.getString(cursor.getColumnIndex(COLUMN_EMAI
L)),
                password =
cursor.getString(cursor.getColumnIndex(COLUMN_PASS
WORD)),
            )
        }
        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_FIRS
T_NAME)),
                    lastName =
cursor.getString(cursor.getColumnIndex(COLUMN_LAST
_NAME)),
                    email =
cursor.getString(cursor.getColumnIndex(COLUMN_EMAI
L)),
                    password =
cursor.getString(cursor.getColumnIndex(COLUMN_PASS
WORD)),
                )
                users.add(user)
            } while (cursor.moveToNext())
        }
        cursor.close()
        db.close()
        return users
    }
```

```
}
```

## **ApiService interface**:

```
package com.example.newsheadlines


import retrofit2.Retrofit

import retrofit2.converter.gson.GsonConverterFactory

import retrofit2.http.GET


interface ApiService {


    //@GET("movielist.json")

    @GET("top-
headlines?country=us&category=business&apiKey=684cb8
93caf7425abeffad82ac1d0f4e")

    ///@GET("search?q=chatgpt")

    suspend fun getMovies() :News


    companion object {
        var apiService: ApiService? = null
        fun getInstance() : ApiService {
            if (apiService == null) {
                apiService = Retrofit.Builder()
                    //
.baseUrl("https://howtodoandroid.com/apis/")
                    .baseUrl("https://newsapi.org/v2/")
                    //.baseUrl("https://podcast-
episodes.p.rapidapi.com/")


.addConverterFactory(GsonConverterFactory.create())
                    .build().create(ApiService::class.java)
            }
            return apiService!!
        }
    }

}
```

## Model class:

```
package com.example.newsheadlines



data class Movie(val name: String,
                 val imageUrl: String,
                 val desc: String,
                 val category: String)
```

News class:

```
package com.example.newsheadlines

import com.example.example.Articles
import com.google.gson.annotations.SerializedName



data class News (
  @SerializedName("status") var status:String?= null,
  @SerializedName("totalResults") var totalResults : Int?
= null,
  @SerializedName("articles") var articles     :
ArrayList<Articles> = arrayListOf()
)
```

## Model class:

```
package com.example.example


import com.google.gson.annotations.SerializedName



data class Source (


  @SerializedName("id"   ) var id   : String? = null,
  @SerializedName("name" ) var name : String? = null


)
```

## Article class:

```
ge"  ) var urlToImage  : String? = null,
package com.example.example
```

```
import com.google.gson.annotations.SerializedName



data class Articles (

  @SerializedName("title"       ) var title       : String? = null,
  @SerializedName("description" ) var description : String?
= null,
  @SerializedName("urlToIma
  )
```

## Main View Model:

```
package com.example.newsheadlines

import android.util.Log
import androidx.compose.runtime.getValue
import androidx.compose.runtime.mutableStateOf
import androidx.compose.runtime.setValue
import androidx.lifecycle.ViewModel
import androidx.lifecycle.viewModelScope
import com.example.example.Articles
import kotlinx.coroutines.launch


class MainViewModel : ViewModel() {
    var movieListResponse:List<Articles> by
mutableStateOf(listOf())
    var errorMessage: String by mutableStateOf("")
    fun getMovieList() {
        viewModelScope.launch {
            val apiService = ApiService.getInstance()
            try {
                val movieList = apiService.getMovies()
                movieListResponse = movieList.articles
            }
            catch (e: Exception) {
                errorMessage = e.message.toString()
            }
        }
    }
}
```

## Display 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.NewsHeadlinesThem
e


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:

```
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.PasswordVisualTransformat
ion

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.NewsHeadlinesThem
e


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.NewsHeadlinesThem
e


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) }
    )
}
}
```

## Registration Activity:

```
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.PasswordVisualTransformat
ion

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.NewsHeadlinesThem
e


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)
    }
```

## Source Code:

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

```
import com.google.gson.annotations.SerializedName



data class Source (


  @SerializedName("id"   ) var id   : String? = null,
  @SerializedName("name" ) var name : String? = null


)
```