Project Report

Podcast Plus: A Redux-Inspired Podcast App With Dynamic Themes

1. Introduction

1.1 Overview

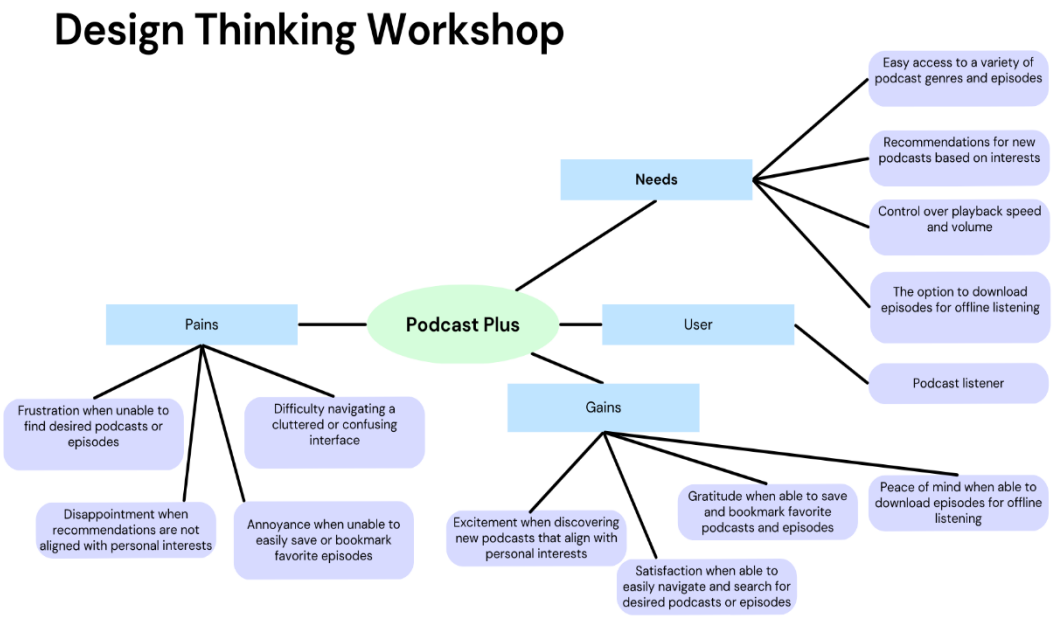
We have developed a podcast app Podcast Plus using Kotlin and Jetpack Compose. It is a compact app that you can listen anywhere everywhere.

1.2 Purpose

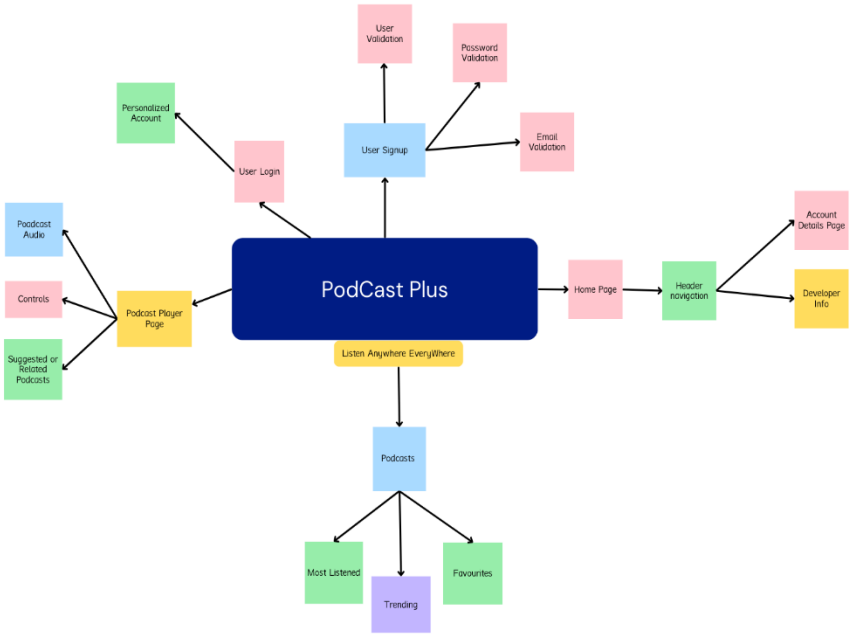
The main purpose of Podcast Plus is to develop an app that is comfortable to use and to pretty much have a very less learning curve when it comes to using the app for everyone

2. Problem Definition & Design Thinking

2.1 Empathy Map

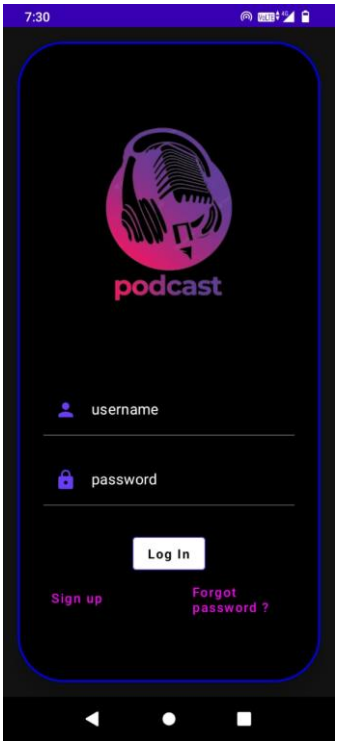


2.2 Ideation and Brainstorm Map

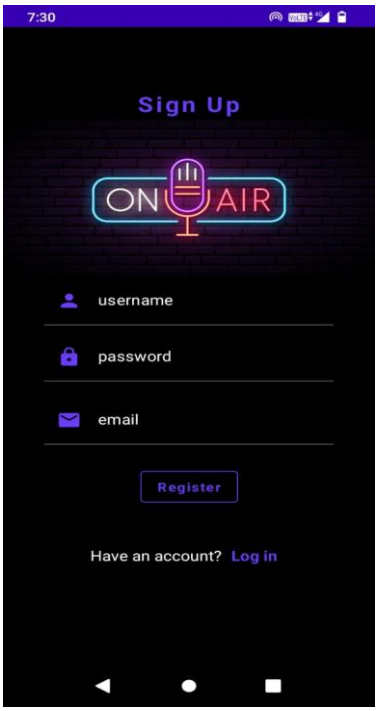


3. Result

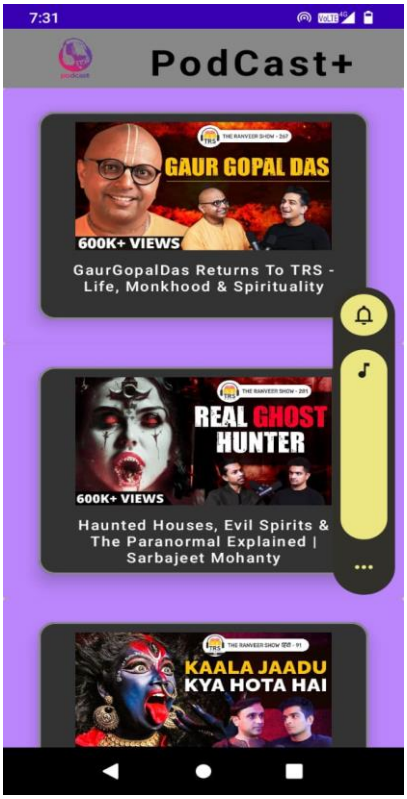
Login Page:



Register Page:



Main Page:


7:31
PodCast+
GAUR GOPAL DAS
600K+ VIEWS
GaurGopalDas Returns To TRS - Life, Monkhood & Spirituality
REAL GHOST HUNTER
600K+ VIEWS
Haunted Houses, Evil Spirits & The Paranormal Explained | Sarbajeet Mohanty
KAALA JAADU KYA HOTA HAI

Player Page:

4. Advantages:

- Easy to use: A simple podcast app is easy to use, navigate, and understand. Users can quickly find the podcasts they want to listen to, and access them without any confusion or frustration.

- Saves time: A simple podcast app can save users a lot of time by providing them with a streamlined user experience. Users can quickly find and access the podcasts they want to listen to, without having to spend a lot of time searching for them.

- Reduced complexity: A simple podcast app reduces the complexity of the user experience, making it more accessible to a wider range of users. This can help attract and retain users who might otherwise be put off by a complex or confusing app.

- Better user engagement: A simple podcast app can help increase user engagement by making it easy for users to find and access their favorite podcasts. This can encourage users to listen to more episodes and spend more time on the app, which can help increase user retention and loyalty.

- Enhanced user experience: A simple podcast app with a login page, sign up page, podcast page with some podcasts, and a player page for each podcast can provide users with a more engaging and personalized experience.

Disadvantages:

- Limited features: A simple podcast app with only basic features may not meet the needs of all users, especially those who are looking for more advanced features such as social sharing, recommendations, or personalized playlists.

- Lack of differentiation: A simple podcast app with a basic design and limited features may not stand out in a crowded marketplace, making it harder to attract and retain users.

- Limited monetization opportunities: A simple podcast app with limited features may have fewer opportunities to monetize through advertising, sponsorships, or premium subscriptions, potentially limiting revenue streams for the app.

- Limited scalability: A simple podcast app with a limited set of features may not be able to handle a large user base or high levels of traffic, potentially limiting its scalability and ability to grow.

- Limited user data: A simple podcast app with limited features may not collect enough user data to provide personalized recommendations or improve the user experience over time.

Application:

- User-friendly interface: The application should have a simple and intuitive interface that allows users to easily search for and listen to podcasts.

- Login and sign-up pages: Users should be able to create an account or log in to access personalized features such as bookmarking, creating playlists, and receiving personalized recommendations.

- Podcast page: The application should have a page dedicated to displaying a variety of podcasts.

- Player page: The application should have a player page for each podcast episode, which should include basic playback controls such as play and pause.

Conclusion:

Overall Podcast plus is a simple to use podcast app that provides a very easy to use interface for everybody. Further improvements can be added to this app which is being discussed in the next session.
Future Scope:

Personalized Suggestions:

The predominant thing that is lacking in Podcast plus is a personalized suggestions due to lack of resources.

s

The podcast can be filtered based on Genre, how frequent it is used and the duration of it.

Personalized Pages:

Allowing the user to customize the user interface like adding night mode and etc.

Appendix:

Login Activity:

```
package com.example.podcast

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.BorderStroke
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.Brush
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.text.ExperimentalTextApi
import androidx.compose.ui.text.TextStyle
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.text.input.PasswordVisualTransformation
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.em
import androidx.compose.ui.unit.sp
import androidx.core.content.ContextCompat
import com.example.podcast.ui.theme.PodcastTheme

class LoginActivity : ComponentActivity() {
    private lateinit var databaseHelper: UserDatabaseHelper
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
```

```
        databaseHelper = UserDatabaseHelper(this)
        setContent {
            PodcastTheme() {
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colors.background
                ) {
                    LoginScreen(this, databaseHelper)
                }
            }
        }
    }
}

@OptIn(ExperimentalTextApi::class)
@Composable
fun LoginScreen(context: Context, databaseHelper: UserDatabaseHelper) {
    var username by remember { mutableStateOf("") }
    var password by remember { mutableStateOf("") }
    var error by remember { mutableStateOf("") }


    Card(
        elevation = 12.dp,
        border = BorderStroke(2.dp, Color.Blue),
        shape = RoundedCornerShape(50.dp),
        modifier = Modifier
            .padding(16.dp)
            .fillMaxWidth(),
        backgroundColor = Color.Black
    ) {


        Column(
            Modifier
                .background(Color.Black)
                .fillMaxHeight()
                .fillMaxWidth()
                .padding(bottom = 28.dp, start = 28.dp, end = 28.dp),
            horizontalAlignment = Alignment.CenterHorizontally,
            verticalArrangement = Arrangement.Center
        )

        {

            Image(
                painter = painterResource(R.drawable.podcast_login),
                contentDescription = "",
                Modifier
                    .height(300.dp)
                    .fillMaxWidth()
            )

            Text(
                text = "LOGIN",
                color = Color.Black,
                fontWeight = FontWeight.Bold,
                fontSize = 26.sp,
                style = MaterialTheme.typography.h1,
                letterSpacing = 0.1.em
            )
```

```
            TextField(
                value = username,
                onValueChange = { username = it },
                leadingIcon = {
                    Icon(
                        imageVector = Icons.Default.Person,
                        contentDescription = "personIcon",
                        tint = Color(0xFF6a3ef9),
                    )
                },
                placeholder = {
                    Text(
                        text = "username",
                        color = Color.White
                    )
                },
                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(0xFF6a3ef9)
                    )
                },
                placeholder = { Text(text = "password", color =
Color.White) },
                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,
                                    MainActivity::class.java
                                )
                            )
                            //onLoginSuccess()
                        } else {
                            error = "Invalid username or password"
                        }
                    } else {
                        error = "Please fill all fields"
                    }
                },
                border = BorderStroke(1.dp, Color(0xFF6a3ef9)),
                colors = ButtonDefaults.buttonColors(backgroundColor =
Color.White),
                modifier = Modifier.padding(top = 16.dp)
            ) {
                Text(text = "Log In", fontWeight = FontWeight.Bold, color =
Color.Black)
            }

            Row(modifier = Modifier.fillMaxWidth()) {
                TextButton(onClick = {
                    context.startActivity(
                        Intent(
                            context,
                            RegistrationActivity::class.java
                        ))})
                {
                    Text(
                        text = "Sign up",
                        color = Color.Magenta
                    )
                }

                Spacer(modifier = Modifier.width(80.dp))

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

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

Register Activity:

```
package com.example.podcast

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.BorderStroke
import androidx.compose.foundation.Image
import androidx.compose.foundation.background
import androidx.compose.foundation.layout.*
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.draw.alpha
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.layout.ContentScale
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.em
import androidx.compose.ui.unit.sp
import androidx.core.content.ContextCompat
import com.example.podcast.ui.theme.PodcastTheme

class RegistrationActivity : ComponentActivity() { private lateinit var
databaseHelper: UserDatabaseHelper
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        databaseHelper = UserDatabaseHelper(this)
        setContent {
            PodcastTheme() {
                // A surface container using the 'background' color from
the theme
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colors.background
                ) {
                    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.Black)
            .fillMaxHeight()
            .fillMaxWidth(),
        horizontalAlignment = Alignment.CenterHorizontally,
        verticalArrangement = Arrangement.Center
    )

    {
        Row {
            Text(
                text = "Sign Up",
                color = Color(0xFF6a3ef9),
                fontWeight = FontWeight.Bold,
                fontSize = 24.sp, style = MaterialTheme.typography.h1,
                letterSpacing = 0.1.em
            )
        }

        Image(
            painter = painterResource(id = R.drawable.podcast_signup),
            contentDescription = ""
        )
        TextField(
            value = username,
            onValueChange = { username = it },
            leadingIcon = {
                Icon(
                    imageVector = Icons.Default.Person,
                    contentDescription = "personIcon",
                    tint = Color(0xFF6a3ef9)
                )
            },
            placeholder = {
                Text(
                    text = "username",
                    color = Color.White
                )
            },
            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(0xFF6a3ef9)
                )
            },
            placeholder = { Text(text = "password", color = Color.White) },
            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(0xFF6a3ef9)
                )
            },
            placeholder = { Text(text = "email", color = Color.White) },
            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()) {
                    if (username.length < 5){
                        error = "Username should be at least 5 characters"
                    }

                    else if (password.length < 8){
                        error = "Password should be at least 8 characters"
                    }

                    else if(!isValidEmail(email)) error ="Enter a valid
email"

                    else{
                        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"
                }
            },
            border = BorderStroke(1.dp, Color(0xFF6a3ef9)),
            colors = ButtonDefaults.buttonColors(backgroundColor =
Color.Black),
            modifier = Modifier.padding(top = 16.dp)
        ) {
            Text(text = "Register",
                fontWeight = FontWeight.Bold,
                color = Color(0xFF6a3ef9)
            )
        }


        Row(
            modifier = Modifier.padding(30.dp),
            verticalAlignment = Alignment.CenterVertically,
            horizontalArrangement = Arrangement.Center
        ) {
            Text(text = "Have an account?", color = Color.White)

            TextButton(onClick = {
                context.startActivity(
                    Intent(
                        context,
                        LoginActivity::class.java
                    )
                )
            })
            {
                Text(text = "Log in",
                    fontWeight = FontWeight.Bold,
                    style = MaterialTheme.typography.subtitle1,
                    color = Color(0xFF6a3ef9)
                )
            }

        }
    }
}


private fun isValidEmail(email: String): Boolean {
    val emailRegex = Regex("[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\\.[a-zA-
Z]{2,}")
    return emailRegex.matches(email)
}


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

MainActivity:

```
package com.example.podcast

import android.annotation.SuppressLint
import android.content.Context
import android.content.Intent
import android.media.MediaPlayer
import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.*
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.material.*
import androidx.compose.runtime.*
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.graphics.painter.Painter
import androidx.compose.ui.res.painterResource
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.em
import androidx.compose.ui.unit.sp
import com.example.podcast.ui.theme.PodcastTheme

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            PodcastTheme {
                // A surface container using the 'background' color from
the theme
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colors.background
                ) {
                    playAudio(this)
                }
            }
        }
    }
}


@SuppressLint("DiscouragedApi")
@Composable
fun ImageCard(
    title: String,
    painter: Painter,
    audioFile: MediaPlayer,
    context: Context
) {
    Button(onClick = { context.startActivity(
        Intent(context, PodcastPlayer::class.java).putExtra("title", title)
    ) }, modifier = Modifier.background(color=Color.LightGray)) {
        Card(
```

```
            elevation = 12.dp,
            border = BorderStroke(2.dp, Color.Gray),
            shape = RoundedCornerShape(15.dp),
            modifier = Modifier
                .padding(16.dp)
                .fillMaxWidth()
                .height(210.dp)
        ) {
            val mp: MediaPlayer = audioFile

            Column(
                modifier = Modifier.fillMaxSize(),
                horizontalAlignment = Alignment.CenterHorizontally
            ) {
                Image(
                    painter = painter,
                    contentDescription = null,
                    modifier = Modifier
                        .height(150.dp)
                        .width(230.dp)
                )

                Text(
                    text = title,
                    textAlign = TextAlign.Center,
                    modifier = Modifier.padding(start = 20.dp, end = 20.dp)
                )

            }

        }
    }
}


@Composable
fun playAudio(context: Context) {

    Column(modifier = Modifier
        .fillMaxSize()
        .background(color = Color.Gray)) {
        Row(modifier = Modifier.height(60.dp)){
            Image(
                painter = painterResource(id = R.drawable.podcast_login),
                contentDescription = null,
                modifier = Modifier
                    .height(100.dp)
                    .width(130.dp),
            )
            Box(contentAlignment = Alignment.Center){
                Text(text = "PodCast+",
                    modifier = Modifier
                        .fillMaxWidth()
                        .padding(top = 9.dp),
                    color = Color.Black,
                    fontWeight = FontWeight.Bold,
                    fontSize = 36.sp,
                    style = MaterialTheme.typography.h1,
                    letterSpacing = 0.1.em
                )
            }
```

```
        }

        Column(modifier = Modifier
            .fillMaxSize()
            .verticalScroll(rememberScrollState())
        ){

                ImageCard(
                    title = "GaurGopalDas Returns To TRS - Life, Monkhood &
Spirituality",
                    painter = painterResource(id = R.drawable.img),
                    audioFile = MediaPlayer.create(context, R.raw.audio),
                    context = context
                )


            ImageCard(
                title = "Haunted Houses, Evil Spirits & The Paranormal
Explained | Sarbajeet Mohanty",
                painter = painterResource(id = R.drawable.img_1),
                audioFile = MediaPlayer.create(context, R.raw.audio_1),
                context = context
            )

            ImageCard(
                title = "Kaali Mata ki kahani - Black Magic & Aghoris ft.
Dr Vineet Aggarwal",
                painter = painterResource(id = R.drawable.img_2),
                audioFile = MediaPlayer.create(context, R.raw.audio_2),
                context = context
            )

            ImageCard(
                title = "Tantra Explained Simply | Rajarshi Nandy - Mata,
Bhairav & Kamakhya Devi",
                painter = painterResource(id = R.drawable.img_3),
                audioFile = MediaPlayer.create(context, R.raw.audio_3),
                context = context
            )

            ImageCard(
                title = "Complete Story Of Shri Krishna - Explained In 20
Minutes",
                painter = painterResource(id = R.drawable.img_4),
                audioFile = MediaPlayer.create(context, R.raw.audio_4),
                context = context
            )

            ImageCard(
                title = "Mahabharat Ki Poori Kahaani - Arjun, Shri Krishna
& Yuddh - Ami Ganatra ",
                painter = painterResource(id = R.drawable.img_5),
                audioFile = MediaPlayer.create(context, R.raw.audio_5),
                context = context
            )
        }
    }
}
```

PodcastPlayer:

```
package com.example.podcast

import android.content.Context
import android.content.Intent
import android.media.Image
import android.media.MediaPlayer
import android.os.Bundle
import android.os.PersistableBundle
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.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.graphics.painter.Painter
import androidx.compose.ui.res.painterResource
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.em
import androidx.compose.ui.unit.sp
import com.example.podcast.ui.theme.PodcastTheme

class PodcastPlayer : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
            setContent {
                PodcastTheme {
                    Surface(modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colors.background) {
                        val title :String =
intent.getStringExtra("title").toString()
                        audioRedirect(title = title, context = this)
                    }
                }
            }
    }
}

@Composable
fun audioRedirect(
    title: String,
    context: Context
){
    if(title == "GaurGopalDas Returns To TRS - Life, Monkhood &
Spirituality")
        audioPlayer(title = title, context = context, audio =
MediaPlayer.create(context, R.raw.audio), painter = painterResource(
            id = R.drawable.img
        ))
    else if(title == "Haunted Houses, Evil Spirits & The Paranormal
Explained | Sarbajeet Mohanty")
        audioPlayer(title = title, context = context, audio =
```

```
MediaPlayer.create(context, R.raw.audio_1), painter = painterResource(
            id = R.drawable.img_1
        ))
    else if(title == "Kaali Mata ki kahani - Black Magic & Aghoris ft. Dr
Vineet Aggarwal")
        audioPlayer(title = title, context = context, audio =
MediaPlayer.create(context, R.raw.audio_2), painter = painterResource(
            id = R.drawable.img_2
        ))
    else if(title == "Tantra Explained Simply | Rajarshi Nandy - Mata,
Bhairav & Kamakhya Devi")
        audioPlayer(title = title, context = context, audio =
MediaPlayer.create(context, R.raw.audio_3), painter = painterResource(
            id = R.drawable.img_3
        ))
    else if(title == "Complete Story Of Shri Krishna - Explained In 20
Minutes")
        audioPlayer(title = title, context = context, audio =
MediaPlayer.create(context, R.raw.audio_4), painter = painterResource(
            id = R.drawable.img_4
        ))
    else
        audioPlayer(title = title, context = context, audio =
MediaPlayer.create(context, R.raw.audio_5), painter = painterResource(
            id = R.drawable.img_5
        ))
}

@Composable
fun audioPlayer(title :String, audio: MediaPlayer, painter: Painter,
context: Context) {
    Card(modifier = Modifier.fillMaxSize().background(color =
Color.LightGray),
    ) {
        Row{
            Box(contentAlignment = Alignment.Center){
                IconButton(onClick = {context.startActivity(
                    Intent(context, MainActivity::class.java))
                }) {
                    Icon(painter = painterResource(id =
R.drawable.backarrow), contentDescription = "",
                        Modifier
                            .size(80.dp)
                            .padding(start = 30.dp, end = 10.dp))
                }
            }

        }

        Column(modifier = Modifier
            .fillMaxWidth()
            .padding(20.dp),
            horizontalAlignment = Alignment.CenterHorizontally,
            verticalArrangement = Arrangement.Center

        ) {

            Card(modifier = Modifier.height(180.dp)){
                Image(
                    painter = painter,
                    contentDescription = "",
```

```
                    )
            }
            Spacer(modifier = Modifier.height(30.dp))
            Text(
                text = title,
                textAlign = TextAlign.Center,
                modifier = Modifier.padding(start = 30.dp, end = 20.dp)
            )
            Spacer(modifier = Modifier.height(30.dp))
            Row() {
                IconButton(onClick = {audio.start()}) {
                    Icon(painter = painterResource(id = R.drawable.play),
contentDescription = "", Modifier.size(50.dp))
                }
                IconButton(onClick = {audio.pause()}) {
                    Icon(painter = painterResource(id = R.drawable.pause),
contentDescription = "", Modifier.size(50.dp))
                }
            }
        }
    }
}
```