

The background features abstract, overlapping green geometric shapes, primarily triangles and polygons, in various shades of green, creating a modern and dynamic visual effect.

# ***A Real-time chat and communication app***


# Description

- ▶ A chat application makes it easy to communicate with people anywhere in the world by sending and receiving messages in real time. With a web or mobile chat app, users are able to receive the same engaging and lively interactions through custom messaging features, just as they would in person. This also keeps users conversing on your platform instead of looking elsewhere for a messaging solution. Whether it's private chat, group chat, or large-scale chat, adding personalized chat features to your app can help ensure that your users have a memorable experience.
- ▶
- ▶ In this chat application guide, we'll explore:
- ▶
- ▶ • How custom in-app chat features enable user engagement
- ▶
- ▶ • The long-term factors to consider when building a real-time chat application

Mainactivity.kt

Package com.project.pradyotprakash.flashchat

```
▶ import android.os.Bundle
▶ import androidx.activity.ComponentActivity
▶ import androidx.activity.compose.setContent
▶ import com.google.firebase.Firebase
▶ class MainActivity : ComponentActivity() {
▶     override fun onCreate(savedInstanceState: Bundle?) {
▶         super.onCreate(savedInstanceState)
▶         FirebaseApp.initializeApp(this)
▶         setContent {
▶             NavComposeApp()
▶         }
▶     }
▶ }
```




- ▶ Navcomposeapp.kt
- ▶ Package com.project.pradyotprakash.flashchat
- ▶ import androidx.compose.runtime.Composable
- ▶ import androidx.compose.runtime.remember
- ▶ import androidx.navigation.compose.NavHost
- ▶ import androidx.navigation.compose.composable
- ▶ import androidx.navigation.compose.rememberNavController
- ▶ import com.google.firebase.auth.FirebaseAuth
- ▶ import com.project.pradyotprakash.flashchat.nav.Action
- ▶ import com.project.pradyotprakash.flashchat.nav.Destination.AuthenticationOption
- ▶ import com.project.pradyotprakash.flashchat.nav.Destination.Home
- ▶ import com.project.pradyotprakash.flashchat.nav.Destination.Login
- ▶ import com.project.pradyotprakash.flashchat.nav.Destination.Register
- ▶ import com.project.pradyotprakash.flashchat.ui.theme.FlashChatTheme
- ▶ import com.project.pradyotprakash.flashchat.view.AuthenticationView
- ▶ import com.project.pradyotprakash.flashchat.view.home.HomeView
- ▶ import com.project.pradyotprakash.flashchat.view.login.LoginView
- ▶ import com.project.pradyotprakash.flashchat.view.register.Register

```
▶ @Composable
▶ fun NavComposeApp() {
▶     val navController = rememberNavController()
▶     val actions = remember(navController) { Action(navController) }
▶     FlashChatTheme {
▶         NavHost(
▶             navController = navController,
▶             startDestination =
▶             if (FirebaseAuth.getInstance().currentUser != null)
▶                 Home
▶             else
▶                 AuthenticationOption
▶         ) {
▶             composable(AuthenticationOption) {
▶                 AuthenticationView(
▶                     register = actions.register,
▶                     login = actions.login
▶                 )
▶             }
▶         }
▶     }
▶ }
```

- ▶ Constants
- ▶ `.package com.project.pradyotprakash.flashchat`
- ▶ object Constants {
- ▶     `const val TAG = "flash-chat"`
- ▶     `const val MESSAGES = "messages"`
- ▶     `const val MESSAGE = "message"`
- ▶     `const val SENT_BY = "sent_by"`
- ▶     `const val SENT_ON = "sent_on"`
- ▶     `const val IS_CURRENT_USER = "is_current_user"`
- ▶ }
- ▶ `Package com.project.pradyotprakash.flashchat.nav`
- ▶
- ▶ `import androidx.navigation.NavHostController`
- ▶ `import com.project.pradyotprakash.flashchat.nav.Destination.Home`
- ▶ `Import com.project.pradyotprakash.flashchat.nav.Destination.Login`



```
▶ Nav package.kt
▶ Import com.project.pradyotprakash.flashchat.nav.Destination.Register
▶ object Destination {
▶     const val AuthenticationOption = "authenticationOption"
▶     const val Register = "register"
▶     const val Login = "login"
▶     const val Home = "home"
▶ }
▶ class Action(navController: NavHostController) {
▶     val home: () -> Unit = {
▶         navController.navigate(Home) {
▶             popUpTo(Login) {
▶                 inclusive = true
▶             }
▶             popUpTo(Register) {
▶                 inclusive = true
▶             }
▶         }
▶     }
▶ }
▶ val login: () -> Unit = { navController.navigate(Login) }
▶ val register: () -> Unit = { navController.navigate(Register) }
```




- ▶ View package
- ▶ Package com.project.pradyotprakash.flashchat.view
- ▶ import androidx.compose.foundation.layout.Arrangement
- ▶ import androidx.compose.foundation.layout.Column
- ▶ import androidx.compose.foundation.layout.fillMaxHeight
- ▶ import androidx.compose.foundation.layout.fillMaxWidth
- ▶ 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 com.project.pradyotprakash.flashchat.ui.theme.FlashChatTheme
- ▶ @Composable
- ▶ fun AuthenticationView(register: () -> Unit, login: () -> Unit) {
- ▶     FlashChatTheme {



```
▶ Surface(color = MaterialTheme.colors.background) {  
▶     Column(  
▶         modifier = Modifier  
▶             .fillMaxWidth()  
▶             .fillMaxHeight(),  
▶         horizontalAlignment = Alignment.CenterHorizontally,  
▶         verticalArrangement = Arrangement.Bottom  
▶     ) {  
▶         Title(title = “⚡ Chat Connect”)  
▶         Buttons(title = “Register”, onClick = register, backgroundColor =  
▶             Color.Blue)  
▶         Buttons(title = “Login”, onClick = login, backgroundColor =  
▶             Color.Magenta)  
▶     }  
▶ }  
▶ }  
▶ }
```

- ▶ Widgets. Kt
- ▶ Package com.project.pradyotprakash.flashchat.view
- ▶ import androidx.compose.foundation.layout.fillMaxHeight
- ▶ import androidx.compose.foundation.layout.fillMaxWidth
- ▶ import androidx.compose.foundation.layout.padding
- ▶ import androidx.compose.foundation.shape.RoundedCornerShape
- ▶ import androidx.compose.foundation.text.KeyboardOptions
- ▶ import androidx.compose.material.\*
- ▶ import androidx.compose.material.icons.Icons
- ▶ import androidx.compose.material.icons.filled.ArrowBack
- ▶ Import androidx.compose.runtime.Composable
- ▶ import androidx.compose.ui.Modifier
- ▶ import androidx.compose.ui.graphics.Color
- ▶ import androidx.compose.ui.text.font.FontWeight
- ▶ import androidx.compose.ui.text.input.KeyboardType
- ▶ import androidx.compose.ui.text.input.VisualTransformation



- ▶ Home
- ▶ Import `androidx.compose.ui.text.style.TextAlign`
- ▶ `import androidx.compose.ui.unit.dp`
- ▶ `import androidx.compose.ui.unit.sp`
- ▶ `import com.project.pradyotprakash.flashchat.`
- ▶ `@Composable`
- ▶ `fun Title(title: String) {`
- ▶     `Text(`
- ▶         `text = title,`
- ▶         `fontSize = 30.sp,`
- ▶         `fontWeight = FontWeight.Bold,`
- ▶         `modifier = Modifier.fillMaxHeight(0.5f)`
- ▶     `)`
- ▶ `}`
- ▶ `@Composable`
- ▶ `fun Buttons(title: String, onClick: () -> Unit, backgroundColor: Color) {`
- ▶     `Button(`
- ▶         `onClick = onClick,`

```
▶ colors = ButtonDefaults.buttonColors(
▶     backgroundColor = backgroundColor,
▶     contentColor = Color.White
▶ ),
▶ modifier = Modifier.fillMaxWidth(),
▶ shape = RoundedCornerShape(0),
▶ ) {
▶     Text(
▶         text = title
▶     )
▶ }
▶ }
▶ @Composable
▶ fun AppBar(title: String, action: () -> Unit) {
▶     TopAppBar(
▶         title = {
▶             Text(text = title)
▶         },
▶         navigationIcon = {
▶             IconButton(
```

```
▶ Icon(  
▶     imageVector = Icons.Filled.ArrowBack,  
▶     contentDescription = "Back button"  
▶ )  
▶ }  
▶ }  
▶ )  
▶ }
```

```
▶ @Composable
```

```
▶ fun TextFormField(value: String, onValueChange: (String) -> Unit, label: String, keyboardType:  
KeyboardType, visualTransformation: VisualTransformation) {
```

```
▶     OutlinedTextField(  
▶         value = value,  
▶         onValueChange = onValueChange,  
▶         label = {  
▶             Text(  
▶                 label  
▶             )  
▶         },  
▶         maxLines = 1,  
▶         modifier = Modifier
```

```
▶ .padding(horizontal = 20.dp, vertical = 5.dp)
▶ .fillMaxWidth(),
▶ keyboardOptions = KeyboardOptions(
▶     keyboardType = keyboardType
▶ ),
▶ singleLine = true,
▶ visualTransformation = visualTransformation
▶ )
▶ }
▶
▶ @Composable
▶ fun SingleMessage(message: String, isCurrentUser: Boolean) {
▶     Card(
▶         shape = RoundedCornerShape(16.dp),
▶         backgroundColor = if (isCurrentUser) MaterialTheme.colors.primary else Color.White
▶     ) Text(
▶         text = message,
▶         textAlign =
▶         if (isCurrentUser)
```

```
▶      TextAlign.End
▶  else
▶      TextAlign.Start,
▶  modifier = Modifier.fillMaxWidth().padding(16.dp),
▶  color = if (!isCurrentUser) MaterialTheme.colors.primary else Color.White
▶  )
▶  }
▶ }
```

▶ Home package

▶ Package com.project.pradyotprakash.flashchat.view.home

▶ import androidx.compose.foundation.background

▶ import androidx.compose.foundation.layout.\*

▶ import androidx.compose.foundation.lazy.LazyColumn

▶ import androidx.compose.foundation.lazy.items

▶ import androidx.compose.foundation.text.KeyboardOptions

▶ import androidx.compose.material.\*

▶ import androidx.compose.material.icons.Icons

▶ import androidx.compose.material.icons.filled.Send

## ► Home viewmodel

Import androidx.compose.runtime.getValue

import androidx.compose.runtime.livedata.observeAsState

import androidx.compose.ui.Alignment

import androidx.compose.ui.Modifier

import androidx.compose.ui.graphics.Color

import androidx.compose.ui.text.input.KeyboardType

import androidx.compose.ui.unit.dp

import androidx.lifecycle.viewmodel.compose.viewModel

import com.project.pradyotprakash.flashchat.Constants

import com.project.pradyotprakash.flashchat.view.

@Composable

fun HomeView(

    homeViewModel: HomeViewModel = viewModel()

► ) {

►     val message: String by homeViewModel.message.observeAsState(initial =  
      “”)




```
▶ modifier = Modifier.fillMaxSize(),
▶ horizontalAlignment = Alignment.CenterHorizontally,
▶ verticalArrangement = Arrangement.Bottom
▶ ) {
▶ LazyColumn(
▶     modifier = Modifier
▶         .fillMaxWidth()
▶         .weight(weight = 0.85f, fill = true),
▶     contentPadding = PaddingValues(horizontal = 16.dp, vertical = 8.dp),
▶     verticalArrangement = Arrangement.spacedBy(4.dp),
▶     reverseLayout = true
▶ ) {
▶     items(messages) { message ->
▶         val isCurrentUser = message[Constants.IS_CURRENT_USER] as Boolean
▶
▶         SingleMessage(
▶             message = message[Constants.MESSAGE].toString(),
▶             isCurrentUser = isCurrentUser
```

```
    )  
  }  
}  
OutlinedTextField(  
  value = message,  
  onChange = {  
    homeViewModel.updateMessage(it)  
  },  
  label = {  
    Text(  
      "Type Your Message"  
    )  
  },  
  maxLines = 1,  
  modifier = Modifier  
    .padding(horizontal = 15.dp, vertical = 1.dp)  
    .fillMaxWidth()  
    .weight(weight = 0.09f, fill = true),  
  keyboardOptions = KeyboardOptions(  
    keyboardType = KeyboardType.Text
```

```
singleLine = true,  
trailingIcon = {  
    IconButton(  
        onClick = {  
            homeViewModel.sendMessage()  
        }  
    ) {  
        Icon(  
            imageVector = Icons.Default.Send,  
            contentDescription = "Send Button"  
        )  
    }  
}  
)  
}  
}
```

- ▶ View model
- ▶ Package com.project.pradyotprakash.flashchat.view.home
- ▶ import android.util.Log
- ▶ import androidx.lifecycle.LiveData
- ▶ import androidx.lifecycle.MutableLiveData
- ▶ import androidx.lifecycle.ViewModel
- ▶ import com.google.firebase.auth.ktx.auth
- ▶ import com.google.firebase.firestore.ktx.firestore
- ▶ import com.google.firebase.ktx.Firebase
- ▶ import com.project.pradyotprakash.flashchat.Constants
- ▶ import java.lang.IllegalArgumentException
- ▶ init {
- ▶     getMessages()
- ▶ } private val \_message = MutableLiveData("")
- ▶ val message: LiveData<String> = \_message
- ▶
- ▶ private var \_messages = MutableLiveData(emptyList<Map<String, Any>>().toMutableList())



```
fun updateMessage(message: String) {  
    _message.value = message  
}  
fun addMessage() {  
    val message: String = _message.value ?: throw  
    IllegalArgumentException("message empty")  
    if (message.isNotEmpty()) {  
        Firebase.firestore.collection(Constants.MESSAGES).document().set(  
            hashMapOf(  
                Constants.MESSAGE to message,  
                Constants.SENT_BY to Firebase.auth.currentUser?.uid,  
                Constants.SENT_ON to System.currentTimeMillis()  
            )  
        ).addOnSuccessListener {  
            _message.value = ""  
        }  
    }  
}
```

```
private fun getMessages() {  
    Firebase.firestore.collection(Constants.MESSAGES)  
        .orderBy(Constants.SENT_ON)  
        .addSnapshotListener { value, e ->  
            if (e != null) {  
                Log.w(Constants.TAG, "Listen failed.", e)  
                return@addSnapshotListener  
            }  
  
            val list = emptyList<Map<String, Any>>().toMutableList()  
  
            if (value != null) {  
                for (doc in value) {  
                    val data = doc.data  
                    data[Constants.IS_CURRENT_USER] =  
                        Firebase.auth.currentUser?.uid.toString() ==  
data[Constants.SENT_BY].toString()  
                }  
            }  
        }  
}
```


```
▶          list.add(data)
▶      }
▶  }
▶
▶      updateMessages(list)
▶  }
▶ }
▶
▶ private fun updateMessages(list: MutableList<Map<String, Any>>) {
▶     _messages.value = list.asReversed()
▶ }
▶ }
▶
▶ Import androidx.compose.ui.text.input.VisualTransformation
▶ import androidx.compose.ui.unit.dp
▶ import androidx.lifecycle.viewmodel.compose.viewModel
▶ import com.project.pradyotprakash.flashchat.view.Appbar
▶ import com.project.pradyotprakash.flashchat.view.Buttons
```

```
▶ Login.kt
▶ Import com.project.pradyotprakash.flashchat.view.Text@Composable
▶ fun LoginView(
▶     home: () -> Unit,
▶     back: () -> Unit,
▶     loginViewModel: LoginViewModel = viewModel()
▶ ) {
▶     val email: String by loginViewModel.email.observeAsState("")
▶     val password: String by loginViewModel.password.observeAsState("")
▶     val loading: Boolean by loginViewModel.loading.observeAsState(initial = false)
▶
▶     Box(
▶         contentAlignment = Alignment.Center,
▶         if (loading) {
▶             CircularProgressIndicator()
▶         }
▶     )
▶     Column(
▶         modifier = Modifier.fillMaxSize(),
▶         horizontalAlignment = Alignment.CenterHorizontally,
```




```
verticalArrangement = Arrangement.Top
) {
  AppBar(
    title = "Login",
    action = back
  )
  TextFormField(
    value = email,
    onValueChange = { loginViewModel.updateEmail(it) },
    label = "Email",
    keyboardType = TextInputType.Email,
    visualTransformation = VisualTransformation.None
  )
  TextFormField(
```

```
TextField(  
    value = password,  
    onChange = { loginViewModel.updatePassword(it) },  
    label = "Password",  
    keyboardType = KeyboardType.Password,  
    visualTransformation = PasswordVisualTransformation()  
)  
Spacer(modifier = Modifier.height(20.dp))  
Buttons(  
    title = "Login",  
    onClick = { loginViewModel.loginUser(home = home) },  
    backgroundColor = Color.Magenta  
)  
}  
}  
}
```



- ▶ Login viewmodel
- ▶ Package com.project.pradyotprakash.flashchat.view.login
- ▶
- ▶ import androidx.lifecycle.LiveData
- ▶ import androidx.lifecycle.MutableLiveData
- ▶ import androidx.lifecycle.ViewModel
- ▶ import com.google.firebase.auth.FirebaseAuth
- ▶ import com.google.firebase.auth.ktx.auth
- ▶ import com.google.firebase.ktx.Firebase
- ▶ import java.lang.IllegalArgumentException
- ▶
- ▶ /\*\*
- ▶ \* View model for the login view.
- ▶ \*/
- ▶ class LoginViewModel : ViewModel() {
- ▶ private val auth: FirebaseAuth = FirebaseAuth
- ▶
- ▶ private val \_email = MutableLiveData("")
- ▶ val email: LiveData<String> = \_email



```
▶ private val _password = MutableLiveData("")
▶ val password: LiveData<String> = _password
▶
▶ private val _loading = MutableLiveData(false)
▶ val loading: LiveData<Boolean> = _loading
▶
▶ // Update email
▶ fun updateEmail(newEmail: String) {
▶     _email.value = newEmail
▶ }
▶
▶ // Update password
▶ fun updatePassword(newPassword: String) {
▶     _password.value = newPassword
▶ }
▶
▶ // Register user
▶ fun loginUser(home: () -> Unit) {
```

val email: String = \_email.value ?: throw IllegalArgumentException("email expected")

val password: **String** =

    \_password.value ?: throw IllegalArgumentException("password expected")

\_loading.value = true

auth.signInWithEmailAndPassword(email, password)

    .addOnCompleteListener {

        if (it.isSuccessful) {

            home()

        }

        \_loading.value = false

    }

}

}

}

