CHAT CONNECT - A REAL TIME CHAT AND COMMUNICATION APP

1.1 INTRODUCTION :

Connect Chat **allows you to interact with other ITIL staff in a**

productive way using a familiar chat tool similar to SMS on your smartphone. There aretwo ways to view Connect Chat, via the Sidebar or Workspace.

- overview :

They can **discuss a range of topics, and even help each other understand thingsthat might confuse them**. Chatting with peers online can help young people to: discuss homework or ideas from school they didn't understand. talk to a friend about something that's happened at school.

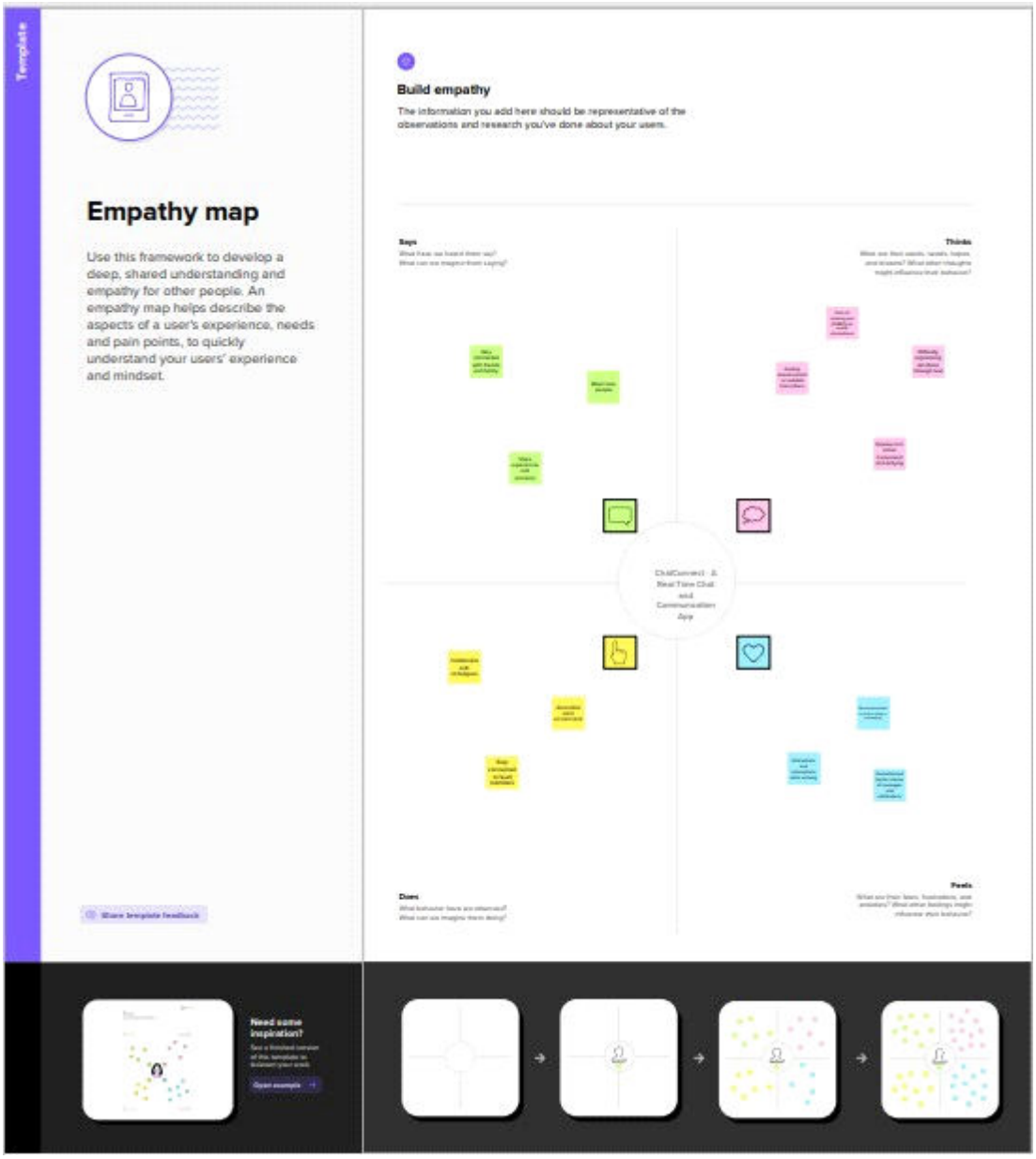
- purpose :

Application software is a computer program that responds to user input and helps them perform personal, professional and educational tasks. This software often isimportant because it **allows you to perform activities that express creativity, fulfil productivity and improve communication**

1.2 PROBLEM DEFINATION & DESIGN THINKING :

There are three types of problem in designthinking: **Simple Problems**. Ill-Defined Problems. Wicked Problems

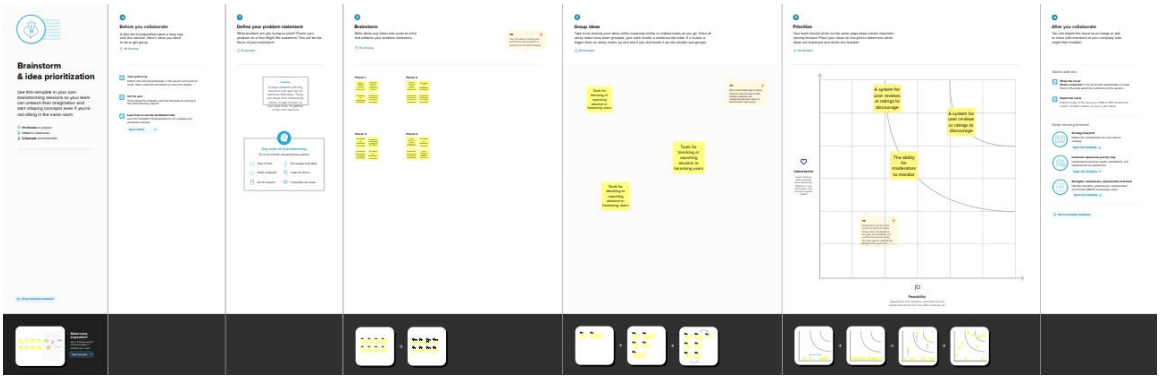
- EMPATHY MAP :



1.3 RESULT :

o check the CAT 2022 result, candidates need to follow the below mentioned steps:

- Visit the official website (iimcat.ac.in)
- Click on 'CAT 2022 scorecard download'
- Enter CAT 2022 ID and password.
- Click on the 'scorecard' tab.
- Download the CAT result 2022 PDF.



ADVANTAGES AND DISADVANTAGES :

The 37 Advantages and Disadvantages of Live Chat

- Faster support. ...
- Real-time text preview. ...
- Instant customer feedback. ...

- Less drama. ...
- Prevents agent fatigue. ...
- No waiting queues. ...
- Non-intrusive. ...
- On-site.

APPLICATION :

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 ableto receive the same engaging and lively interactions through custom messaging features, just as they would in person

CONCLUSION :

chat room is **an online platform that enables users to**

communicate with each other in real time. Chat rooms are typically hosted on a server withan internet connection, enabling members from around the world to hold conversations about various topics.

FUTURE SCOPE :

The future of chatbots is **transforming the way businesses interact with their customers**. From handling customer inquiries and offering real-time support to providing personalized product recommendations, chatbots are becoming increasingly important for all types of businesses in the digital ag

APPENDIX :

```
//Mainactivty.kt
        package com.project.pradyotprakash.flashchat

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import com.google.firebase.FirebaseApp

/**
 * The initial point of the application from where it gets started.
 *
 * Here we do all the initialization and other things which will be required
 * thought out the application.
 */
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.RegisterView

/**
 * The main Navigation composable which will handle all the navigation stack.
 */

@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
                )
            }
            composable(Register) {
                RegisterView(
                    home = actions.home,
                    back = actions.navigateBack
                )
            }
            composable(Login) {
                LoginView(
                    home = actions.home,
                    back = actions.navigateBack
                )
            }
            composable(Home) {
                HomeView()
            }
        }
    }
}

//Constants.kt
```

```
        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"
}

//Navigation.kt

        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
import com.project.pradyotprakash.flashchat.nav.Destination.Register

/**
 * A set of destination used in the whole application
 */
object Destination {
   const val AuthenticationOption = "authenticationOption"
   const val Register = "register"
   const val Login = "login"
   const val Home = "home"
}

/**
 * Set of routes which will be passed to different composable so that
 * the routes which are required can be taken.
 */
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) }
   val navigateBack: () -> Unit = { navController.popBackStack() }
}

//AuthenticationOption.kt
```

```
        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

/**
 * The authentication view which will give the user an option to choose between
 * login and register.
 */

@Composable
fun AuthenticationView(register: () -> Unit, login: () -> Unit) {
   FlashChatTheme {
      // A surface container using the 'background' color from the theme
      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
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import com.project.pradyotprakash.flashchat.Constants

/**
 * Set of widgets/views which will be used throughout the application.
 * This is used to increase the code usability.
 */

@Composable
fun Title(title: String) {
    Text(
        text = title,
        fontSize = 30.sp,
        fontWeight = FontWeight.Bold,
        modifier = Modifier.fillMaxHeight(0.5f)
    )
}

// Different set of buttons in this page
@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(
```

```
            onClick = action
        ) {
            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.kt

        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
import androidx.compose.runtime.Composable
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.SingleMessage

/**
 * The home view which will contain all the code related to the view for HOME.
 *
 * Here we will show the list of chat messages sent by user.
 * And also give an option to send a message and logout.
 */

@Composable
fun HomeView(
    homeViewModel: HomeViewModel = viewModel()
) {
    val message: String by homeViewModel.message.observeAsState(initial = "")
    val messages: List<Map<String, Any>> by homeViewModel.messages.observeAsState(
        initial = emptyList<Map<String, Any>>().toMutableList()
    )

    Column(
        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,
            onValueChange = {
                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.addMessage()
                    }
                ) {
                    Icon(
                        imageVector = Icons.Default.Send,
                        contentDescription = "Send Button"
                    )
                }
            }
        )
    }
}

//HomeViewModel.kt

        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

/**
 * Home view model which will handle all the logic related to HomeView
 */
class HomeViewModel : ViewModel() {
    init {
        getMessages()
    }

    private val _message = MutableLiveData("")
    val message: LiveData<String> = _message

    private var _messages = MutableLiveData(emptyList<Map<String, Any>>().toMutableList())
    val messages: LiveData<MutableList<Map<String, Any>>> = _messages

    /**
     * Update the message value as user types
     */
    fun updateMessage(message: String) {
        _message.value = message
    }

    /**
     * Send 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 = ""
            }
        }
    }

    /**
     * Get the messages
     */
    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)
            }
    }

    /**
     * Update the list after getting the details from firestore
     */
    private fun updateMessages(list: MutableList<Map<String, Any>>) {
        _messages.value = list.asReversed()
    }
}

//Login.kt

        package com.project.pradyotprakash.flashchat.view.login

import androidx.compose.foundation.layout.*
import androidx.compose.material.CircularProgressIndicator
import androidx.compose.runtime.Composable
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.text.input.PasswordVisualTransformation
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
import com.project.pradyotprakash.flashchat.view.TextFormField
```

```
/**
 * The login view which will help the user to authenticate themselves and go to the
 * home screen to show and send messages to others.
 */

@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,
        modifier = Modifier.fillMaxSize()
    ) {
        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 = KeyboardType.Email,
                visualTransformation = VisualTransformation.None
            )
            TextFormField(
                value = password,
                onValueChange = { 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
            )
        }
```

```
    }
}

//LoginViewModel.kt

        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 = Firebase.auth

    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) {
        if (_loading.value == false) {
            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
        }
    }
  }
}

//Register.kt

        package com.project.pradyotprakash.flashchat.view.register

import androidx.compose.foundation.layout.*
import androidx.compose.material.CircularProgressIndicator
import androidx.compose.runtime.Composable
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.text.input.PasswordVisualTransformation
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
import com.project.pradyotprakash.flashchat.view.TextFormField

/**
 * The Register view which will be helpful for the user to register themselves into
 * our database and go to the home screen to see and send messages.
 */

@Composable
fun RegisterView(
  home: () -> Unit,
  back: () -> Unit,
  registerViewModel: RegisterViewModel = viewModel()
) {
  val email: String by registerViewModel.email.observeAsState("")
  val password: String by registerViewModel.password.observeAsState("")
  val loading: Boolean by registerViewModel.loading.observeAsState(initial = false)

  Box(
    contentAlignment = Alignment.Center,
    modifier = Modifier.fillMaxSize()
  ) {
    if (loading) {
      CircularProgressIndicator()
    }
    Column(
```

```
            modifier = Modifier.fillMaxSize(),
            horizontalAlignment = Alignment.CenterHorizontally,
            verticalArrangement = Arrangement.Top
        ) {
            Appbar(
                title = "Register",
                action = back
            )
            TextFormField(
                value = email,
                onValueChange = { registerViewModel.updateEmail(it) },
                label = "Email",
                keyboardType = KeyboardType.Email,
                visualTransformation = VisualTransformation.None
            )
            TextFormField(
                value = password,
                onValueChange = { registerViewModel.updatePassword(it) },
                label = "Password",
                keyboardType = KeyboardType.Password,
                visualTransformation = PasswordVisualTransformation()
            )
            Spacer(modifier = Modifier.height(20.dp))
            Buttons(
                title = "Register",
                onClick = { registerViewModel.registerUser(home = home) },
                backgroundColor = Color.Blue
            )
        }
    }
}

//RegisterViewModel.kt

            package com.project.pradyotprakash.flashchat.view.register

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 RegisterViewModel : ViewModel() {
    private val auth: FirebaseAuth = Firebase.auth

    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 registerUser(home: () -> Unit) {
        if (_loading.value == false) {
            val email: String = _email.value ?: throw IllegalArgumentException("email expected")
            val password: String =
                _password.value ?: throw IllegalArgumentException("password expected")

            _loading.value = true

            auth.createUserWithEmailAndPassword(email, password)
                .addOnCompleteListener {
                    if (it.isSuccessful) {
                        home()
                    }
                    _loading.value = false
                }
        }
    }
}
```

Chat Support refers to **real-time communication between a customer and customer support agent via instant messaging, usually through a pop-up dialogue box built into acompany's website**