**MINI PROJECT**

**ACTNOW: A COMMUNTITY SERVICE CONNECTOR APP**

**MainActivity.kt**

package com.example.communityconnector  
  
import android.content.Intent  
import android.content.SharedPreferences  
import android.os.Bundle  
import android.widget.\*  
import androidx.appcompat.app.AppCompatActivity  
  
class MainActivity : AppCompatActivity() {  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.activity\_main)  
  
 val nameEditText = findViewById<EditText>(R.id.nameEditText)  
 val emailEditText = findViewById<EditText>(R.id.emailEditText)  
 val roleGroup = findViewById<RadioGroup>(R.id.roleGroup)  
 val submitButton = findViewById<Button>(R.id.submitButton)  
  
 submitButton.setOnClickListener {  
 val name = nameEditText.text.toString()  
 val email = emailEditText.text.toString()  
 val selectedRoleId = roleGroup.checkedRadioButtonId  
  
 if (name.isNotEmpty() && email.isNotEmpty() && selectedRoleId != -1) {  
 val selectedRole = findViewById<RadioButton>(selectedRoleId).text.toString()  
  
 // Save login details locally  
 val sharedPref = getSharedPreferences("UserPrefs", MODE\_PRIVATE)  
 with(sharedPref.edit()) {  
 putString("userName", name)  
 putString("userEmail", email)  
 putString("userRole", selectedRole)  
 apply()  
 }  
  
 if (selectedRole == "Volunteer") {  
 startActivity(Intent(this, VolunteerHomeActivity::class.java))  
 } else if (selectedRole == "Organization") {  
 startActivity(Intent(this, OrganizationHomeActivity::class.java))  
 }  
 } else {  
 Toast.makeText(this, "Please fill in all details", Toast.LENGTH\_SHORT).show()  
 }  
 }  
 }  
}

**VolunteerHomeActivity.kt**  
package com.example.communityconnector  
import android.content.Intent  
import android.os.Bundle  
import android.widget.Toast  
import androidx.activity.ComponentActivity  
import androidx.activity.compose.setContent  
import androidx.compose.foundation.clickable  
import androidx.compose.foundation.layout.\*  
import androidx.compose.foundation.verticalScroll  
import androidx.compose.foundation.rememberScrollState  
import androidx.compose.material3.\*  
import androidx.compose.runtime.\*  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.platform.*LocalContext*import androidx.compose.ui.unit.dp  
  
class VolunteerHomeActivity : ComponentActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *setContent* **{** VolunteerHomeScreen()  
 **}** }  
}  
  
@Composable  
fun VolunteerHomeScreen() {  
 val context = *LocalContext*.current  
  
 val defaultEvents = *listOf*(  
 Event("Beach Cleanup", "Join us to clean the local beach.", "Marina Beach"),  
 Event("Tree Plantation", "Help plant trees in the community park.", "Green Park")  
 )  
  
 val events = remember **{** *mutableStateListOf*<Event>().*apply* **{** addAll(defaultEvents); addAll(EventRepository.getEvents()) **} }** val registeredEvents = remember **{** *mutableStateListOf*<Event>() **}** var selectedEvent by remember **{** *mutableStateOf*<Event?>(null) **}** var showRegisteredDialog by remember **{** *mutableStateOf*(false) **}** val scrollState = rememberScrollState()  
  
 Scaffold (  
 containerColor = MaterialTheme.colorScheme.background  
 )**{** innerPadding **->** Column(  
 modifier = Modifier  
 .*padding*(innerPadding)  
 .*padding*(16.*dp*)  
 .*verticalScroll*(scrollState)  
 ) **{** // Back Button  
 Button(  
 onClick = **{** (context as? VolunteerHomeActivity)?.finish() **}**,  
 colors = ButtonDefaults.buttonColors(containerColor = MaterialTheme.colorScheme.primary),  
 // colors = ButtonDefaults.buttonColors(containerColor = MaterialTheme.colorScheme.primaryContainer),  
 modifier = Modifier.*fillMaxWidth*()  
 // onClick = { (context as? VolunteerHomeActivity)?.finish() },  
 // colors = ButtonDefaults.buttonColors(containerColor = MaterialTheme.colorScheme.primaryContainer),  
 // modifier = Modifier.fillMaxWidth()  
 ) **{** Text("Back to Main", color = MaterialTheme.colorScheme.onPrimary)  
 **}** Spacer(modifier = Modifier.*height*(12.*dp*))  
  
 // Registered Events Button  
 Button(  
 onClick = **{** showRegisteredDialog = true **}**,  
 colors = ButtonDefaults.buttonColors(containerColor = MaterialTheme.colorScheme.secondary),  
 modifier = Modifier.*fillMaxWidth*()  
 ) **{** Text("View Registered Events", color = MaterialTheme.colorScheme.onSecondary)  
 **}** Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 // Text(text = "Volunteer Home", style = MaterialTheme.typography.headlineMedium)  
 Text(text = "ActNow - Volunteer Home", style = MaterialTheme.typography.headlineMedium,color = MaterialTheme.colorScheme.primary)  
  
 Spacer(modifier = Modifier.*height*(16.*dp*))  
  
 events.*forEach* **{** event **->** Card(  
 colors = CardDefaults.cardColors(  
 containerColor = MaterialTheme.colorScheme.tertiaryContainer  
 ),  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(vertical = 4.*dp*)  
 .*clickable* **{** selectedEvent = event **}**,  
 elevation = CardDefaults.cardElevation(4.*dp*)  
 ) **{** Column(modifier = Modifier.*padding*(16.*dp*)) **{** Text(text = event.name, style = MaterialTheme.typography.titleMedium)  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Button(onClick = **{** if (!registeredEvents.contains(event)) {  
 registeredEvents.add(event)  
 Toast.makeText(context, "Registered for ${event.name}", Toast.*LENGTH\_SHORT*).show()  
 } else {  
 Toast.makeText(context, "Already registered!", Toast.*LENGTH\_SHORT*).show()  
 }  
 **}**) **{** Text("Register")  
 **}  
 }  
 }  
 }** // Event Details Dialog  
 selectedEvent?.*let* **{** event **->** AlertDialog(  
 onDismissRequest = **{** selectedEvent = null **}**,  
 confirmButton = **{** TextButton(onClick = **{** selectedEvent = null **}**) **{** Text("Close")  
 **}  
 }**,  
 title = **{** Text(event.name) **}**,  
 text = **{** Column **{** Text("Description: ${event.description}")  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
 Text("Location: ${event.location}")  
 **}  
 }** )  
 **}** // Registered Events Dialog  
 if (showRegisteredDialog) {  
 AlertDialog(  
 onDismissRequest = **{** showRegisteredDialog = false **}**,  
 confirmButton = **{** TextButton(onClick = **{** showRegisteredDialog = false **}**) **{** Text("Close")  
 **}  
 }**,  
 title = **{** Text("Registered Events") **}**,  
 text = **{** if (registeredEvents.*isNotEmpty*()) {  
 Column **{** registeredEvents.*forEach* **{** event **->** Text("- ${event.name}")  
 Spacer(modifier = Modifier.*height*(4.*dp*))  
 **}  
 }** } else {  
 Text("No events registered yet.")  
 }  
 **}** )  
 }  
 **}  
 }**}

**OrganizationHomeActivity.kt**

package com.example.communityconnector  
  
import android.os.Bundle  
import android.widget.Toast  
import androidx.activity.ComponentActivity  
import androidx.activity.compose.setContent  
import androidx.compose.foundation.layout.\*  
import androidx.compose.foundation.rememberScrollState  
import androidx.compose.foundation.verticalScroll  
import androidx.compose.foundation.shape.RoundedCornerShape  
import androidx.compose.material3.\*  
import androidx.compose.runtime.\*  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.platform.*LocalContext*import androidx.compose.ui.unit.dp  
import androidx.compose.ui.draw.shadow  
import com.example.communityconnector.Event  
import com.example.communityconnector.EventRepository  
  
class OrganizationHomeActivity : ComponentActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *setContent* **{** OrganizationHomeScreen()  
 **}** }  
}  
@Composable  
fun OrganizationHomeScreen() {  
 val context = *LocalContext*.current  
  
 var eventName by remember **{** *mutableStateOf*("") **}** var eventDescription by remember **{** *mutableStateOf*("") **}** var eventLocation by remember **{** *mutableStateOf*("") **}** val events = remember **{** *mutableStateListOf*<Event>().*apply* **{** addAll(EventRepository.eventList) **} }** val scrollState = rememberScrollState()  
  
 Surface(  
 modifier = Modifier.*fillMaxSize*(),  
 color = MaterialTheme.colorScheme.background  
 ) **{** Column(  
 modifier = Modifier  
 .*padding*(16.*dp*)  
 .*verticalScroll*(scrollState)  
 ) **{** Text(  
 text = "ActNow - Organization Home",  
 style = MaterialTheme.typography.headlineMedium  
 //color = MaterialTheme.colorScheme.primary  
 )  
  
   
 Button(  
 onClick = **{** (context as? ComponentActivity)?.finish()   
 **}**,  
 colors = ButtonDefaults.buttonColors(containerColor = MaterialTheme.colorScheme.secondary),  
 shape = *RoundedCornerShape*(10.*dp*),  
 modifier = Modifier.*padding*(bottom = 16.*dp*)  
 ) **{** Text(text = "Back", color = MaterialTheme.colorScheme.onSecondary)  
 **}** Text(text = "Add Event", style = MaterialTheme.typography.headlineMedium)  
  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
  
 OutlinedTextField(  
 value = eventName,  
 onValueChange = **{** eventName = **it }**,  
 label = **{** Text("Event Name") **}**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*shadow*(2.*dp*, *RoundedCornerShape*(10.*dp*)),  
 shape = *RoundedCornerShape*(10.*dp*)  
 )  
  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
  
 OutlinedTextField(  
 value = eventDescription,  
 onValueChange = **{** eventDescription = **it }**,  
 label = **{** Text("Description") **}**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*shadow*(2.*dp*, *RoundedCornerShape*(10.*dp*)),  
 shape = *RoundedCornerShape*(10.*dp*)  
 )  
  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
  
 OutlinedTextField(  
 value = eventLocation,  
 onValueChange = **{** eventLocation = **it }**,  
 label = **{** Text("Location") **}**,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*shadow*(2.*dp*, *RoundedCornerShape*(10.*dp*)),  
 shape = *RoundedCornerShape*(10.*dp*)  
 )  
  
 Spacer(modifier = Modifier.*height*(12.*dp*))  
  
 Button(  
 onClick = **{** if (eventName.*isNotBlank*() && eventDescription.*isNotBlank*() && eventLocation.*isNotBlank*()) {  
 val newEvent = Event(eventName, eventDescription, eventLocation)  
 EventRepository.eventList.add(newEvent)  
 events.add(newEvent)  
 eventName = ""  
 eventDescription = ""  
 eventLocation = ""  
 Toast.makeText(context, "Event added successfully!", Toast.*LENGTH\_SHORT*).show()  
 } else {  
 Toast.makeText(context, "Please fill in all fields", Toast.*LENGTH\_SHORT*).show()  
 }  
 **}**,  
 shape = *RoundedCornerShape*(10.*dp*),  
 colors = ButtonDefaults.buttonColors(containerColor = MaterialTheme.colorScheme.primary)  
 ) **{** Text("Add Event", color = MaterialTheme.colorScheme.onPrimary)  
 **}** Spacer(modifier = Modifier.*height*(20.*dp*))  
  
 Text(text = "Your Events", style = MaterialTheme.typography.titleLarge)  
  
 Spacer(modifier = Modifier.*height*(10.*dp*))  
  
 events.*forEachIndexed* **{** index, event **->** Card(  
  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(vertical = 4.*dp*)  
 .*shadow*(4.*dp*, *RoundedCornerShape*(16.*dp*)),  
 shape = *RoundedCornerShape*(16.*dp*),  
 // colors = CardDefaults.cardColors(  
 // containerColor = MaterialTheme.colorScheme.tertiaryContainer surfaceVariant  
 //),  
 colors = CardDefaults.cardColors(  
 containerColor = MaterialTheme.colorScheme.tertiaryContainer  
 )  
 ) **{** Column(modifier = Modifier.*padding*(16.*dp*)) **{** Text(text = "Name: ${event.name}", style = MaterialTheme.typography.titleMedium)  
 Text(text = "Description: ${event.description}", style = MaterialTheme.typography.bodyMedium)  
 Text(text = "Location: ${event.location}", style = MaterialTheme.typography.bodySmall)  
  
 Spacer(modifier = Modifier.*height*(8.*dp*))  
  
 Button(  
 onClick = **{** EventRepository.eventList.remove(event)  
 events.removeAt(index)  
 Toast.makeText(context, "Event deleted", Toast.*LENGTH\_SHORT*).show()  
 **}**,  
 colors = ButtonDefaults.buttonColors(containerColor = MaterialTheme.colorScheme.error),  
 //colors = ButtonDefaults.buttonColors(containerColor = MaterialTheme.colorScheme.errorContainer),  
 shape = *RoundedCornerShape*(10.*dp*)  
 ) **{** Text("Delete Event", color = MaterialTheme.colorScheme.onError)  
 **}  
 }  
 }  
 }  
 }  
 }**}

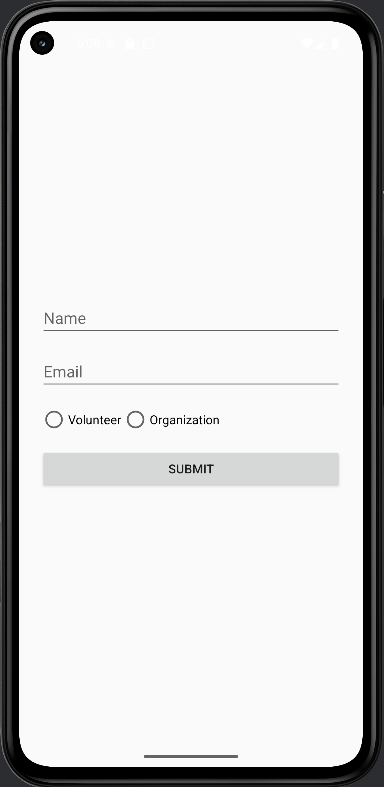
**HomeActivity.kt**

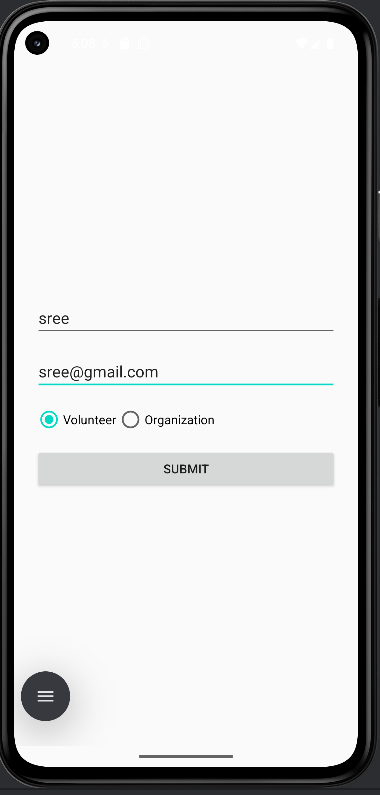
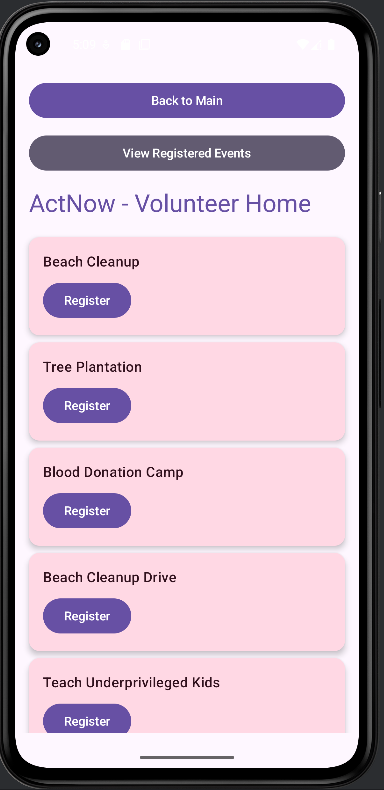
package com.example.communityconnector  
  
import android.os.Bundle  
import android.widget.Toast  
import androidx.activity.ComponentActivity  
import androidx.activity.compose.setContent  
import androidx.compose.foundation.layout.Column  
import androidx.compose.foundation.layout.padding  
import androidx.compose.foundation.text.BasicText  
import androidx.compose.material3.Button  
import androidx.compose.material3.Scaffold  
import androidx.compose.runtime.Composable  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.platform.*LocalContext*import androidx.compose.ui.unit.dp  
import com.example.communityconnector.ui.theme.CommunityConnectorTheme  
  
import androidx.compose.foundation.layout.\*  
import androidx.compose.material3.\*  
  
import androidx.compose.ui.Alignment  
  
  
  
class HomeActivity : ComponentActivity() {  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 *setContent* **{** CommunityConnectorTheme **{** EventListScreen()  
 **}  
 }** }  
}  
  
@Composable  
fun EventListScreen() {  
 val events = *listOf*(  
 Event("Blood Donation", "Save lives by donating blood", "Location A"),  
 Event("Beach Cleanup", "Help clean up the beach", "Location B"),  
 Event("Teach Kids", "Help teach kids in your community", "Location C")  
 )  
  
 Scaffold **{** paddingValues **->** Column(  
 modifier = Modifier  
 .*padding*(paddingValues)  
 .*padding*(16.*dp*)  
 ) **{** events.*forEach* **{** event **->** EventCard(event)  
 **}  
 }  
 }**}  
  
  
@Composable  
fun EventCard(event: Event) {  
 val context = *LocalContext*.current  
  
 Column(modifier = Modifier.*padding*(8.*dp*)) **{** Row(  
 verticalAlignment = Alignment.CenterVertically,  
 modifier = Modifier.*fillMaxWidth*()  
 ) **{** BasicText(  
 text = event.name,  
 style = MaterialTheme.typography.bodyLarge,  
 modifier = Modifier.*weight*(1f)  
 )  
 Button(onClick = **{** Toast.makeText(context, "Registered for ${event.name}", Toast.*LENGTH\_SHORT*).show()  
 **}**) **{** BasicText(text = "Join Event")  
 **}  
 }** Spacer(modifier = Modifier.*height*(4.*dp*))  
  
 BasicText(  
 text = "Description: ${event.description}",  
 style = MaterialTheme.typography.bodyMedium,  
 modifier = Modifier.*padding*(start = 8.*dp*)  
 )  
 BasicText(  
 text = "Location: ${event.location}",  
 style = MaterialTheme.typography.bodyMedium,  
 modifier = Modifier.*padding*(start = 8.*dp*)  
 )  
 **}**}

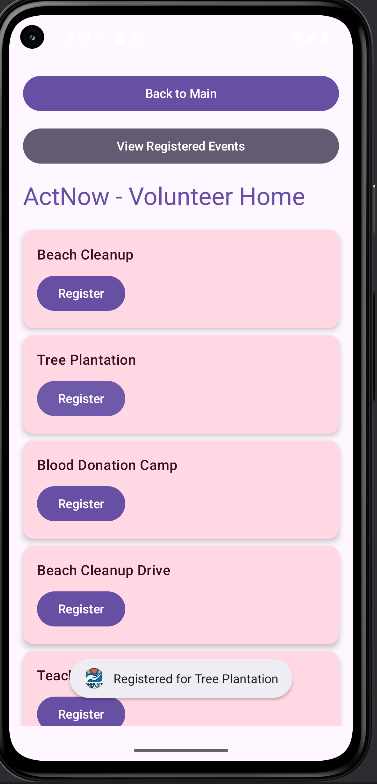
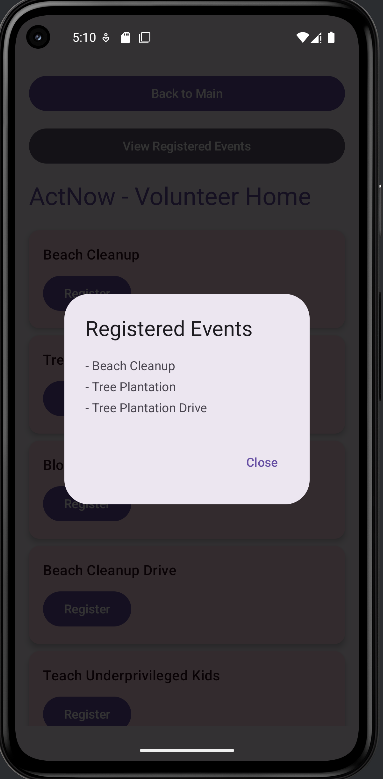
**Activity\_main.xml**

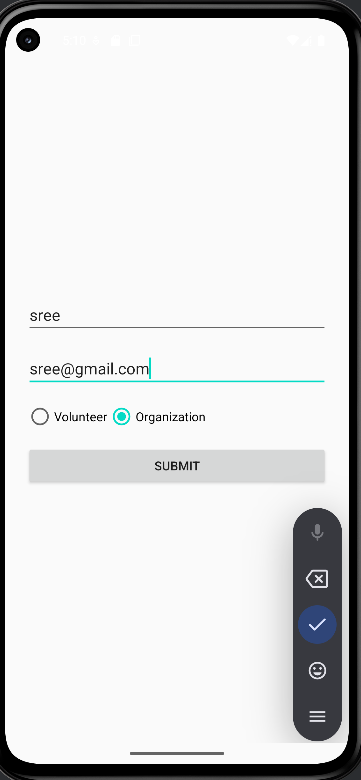
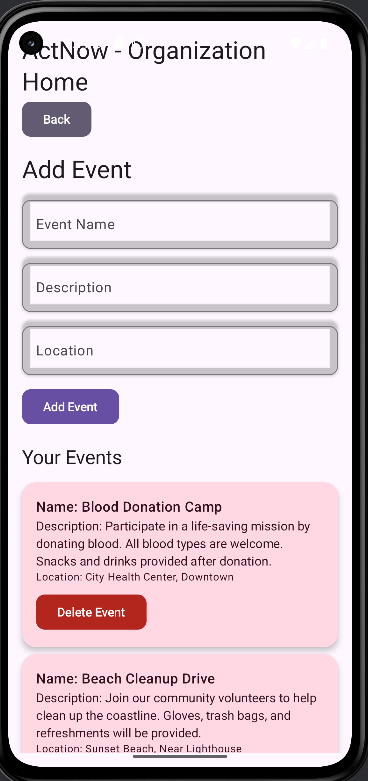
<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:gravity="center"  
 android:padding="24dp">  
  
 <EditText  
 android:id="@+id/nameEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Name"  
 android:inputType="text"  
 android:layout\_marginBottom="16dp" />  
  
 <EditText  
 android:id="@+id/emailEditText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Email"  
 android:inputType="textEmailAddress"  
 android:layout\_marginBottom="16dp" />  
  
 <RadioGroup  
 android:id="@+id/roleGroup"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 android:layout\_marginBottom="16dp">  
  
 <RadioButton  
 android:id="@+id/volunteerRadio"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Volunteer" />  
  
 <RadioButton  
 android:id="@+id/organizationRadio"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Organization" />  
 </RadioGroup>  
  
 <Button  
 android:id="@+id/submitButton"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Submit" />  
</LinearLayout>

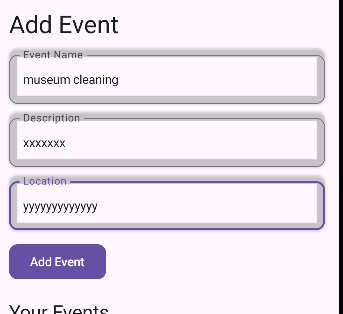
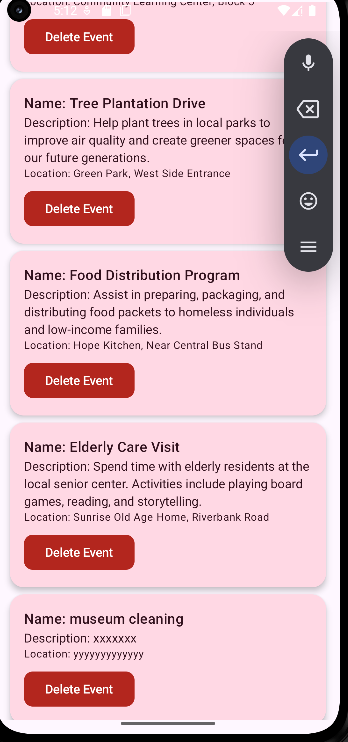
**OUTPUT SCREENSHOTS:**

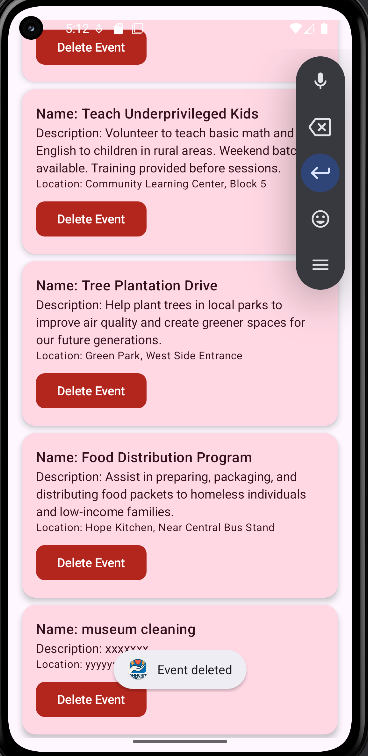
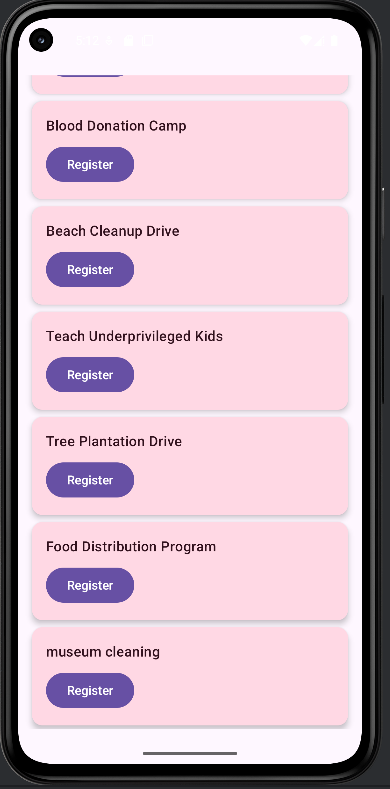
** **

** **

** **

** **

** **

** **