**AIM:**

Implement an application that writes data to the SD card

TO BUILD AN APPLICATION USING KOTLIN TO WRITE INTO SD.

**CODE:**

**package com.example.sd**

**import android.Manifest**

**import android.content.ContentValues**

**import android.content.pm.PackageManager**

**import android.os.Build**

**import android.os.Bundle**

**import android.os.Environment**

**import android.provider.MediaStore**

**import android.widget.Button**

**import android.widget.EditText**

**import android.widget.TextView**

**import android.widget.Toast**

**import androidx.appcompat.app.AppCompatActivity**

**import androidx.core.app.ActivityCompat**

**import androidx.core.content.ContextCompat**

**import java.io.OutputStream**

**import kotlinx.coroutines.Dispatchers**

**import kotlinx.coroutines.GlobalScope**

**import kotlinx.coroutines.launch**

**class MainActivity : AppCompatActivity() {**

**private val REQUEST\_CODE = 101**

**private lateinit var statusTextView: TextView // Declare a reference to the TextView**

**override fun onCreate(savedInstanceState: Bundle?) {**

**super.onCreate(savedInstanceState)**

**setContentView(R.layout.activity\_main)**

**val editText = findViewById<EditText>(R.id.editText)**

**val saveButton = findViewById<Button>(R.id.saveButton)**

**statusTextView = findViewById(R.id.statusTextView) // Initialize the TextView**

**saveButton.setOnClickListener {**

**val text = editText.text.toString()**

**if (checkPermission()) {**

**// Save to file asynchronously to avoid blocking the UI thread**

**GlobalScope.launch(Dispatchers.IO) {**

**saveToFile(text) // Save to shared storage using MediaStore**

**}**

**} else {**

**requestPermission()**

**}**

**}**

**}**

**private fun saveToFile(data: String) {**

**// Prepare the content values for the file to be created**

**val contentValues = ContentValues().apply {**

**put(MediaStore.MediaColumns.DISPLAY\_NAME, "myfile.txt") // File name**

**put(MediaStore.MediaColumns.MIME\_TYPE, "text/plain")**

**put(MediaStore.MediaColumns.RELATIVE\_PATH, Environment.DIRECTORY\_DOCUMENTS) // or Environment.DIRECTORY\_DOWNLOADS for Downloads folder**

**}**

**// Get content resolver and insert into MediaStore**

**val resolver = contentResolver**

**val uri = resolver.insert(MediaStore.Files.getContentUri("external"), contentValues)**

**uri?.let {**

**try {**

**// Open an output stream to write the data**

**val outputStream: OutputStream? = resolver.openOutputStream(it)**

**outputStream?.write(data.toByteArray())**

**outputStream?.close()**

**// Show success in the TextView (Make sure to update UI on main thread)**

**runOnUiThread {**

**statusTextView.text = "File saved successfully!" // Update TextView**

**Toast.makeText(this, "File saved successfully!", Toast.LENGTH\_LONG).show() // Optional Toast**

**}**

**} catch (e: Exception) {**

**// Handle any errors (Make sure to update UI on main thread)**

**runOnUiThread {**

**statusTextView.text = "Error: ${e.message}" // Update TextView with error message**

**Toast.makeText(this, "Error: ${e.message}", Toast.LENGTH\_LONG).show()**

**}**

**}**

**} ?: run {**

**// Handle failure if URI is null (Make sure to update UI on main thread)**

**runOnUiThread {**

**statusTextView.text = "Failed to create file" // Update TextView with failure message**

**Toast.makeText(this, "Failed to create file", Toast.LENGTH\_LONG).show()**

**}**

**}**

**}**

**private fun checkPermission(): Boolean {**

**return if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.Q) {**

**// No need to request storage permission on Android 10+ for app-specific storage**

**true**

**} else {**

**// For Android versions below API 29 (Android 9 and below), check**

**WRITE\_EXTERNAL\_STORAGE permission**

**val result = ContextCompat.checkSelfPermission(**

**this, Manifest.permission.WRITE\_EXTERNAL\_STORAGE**

**)**

**result == PackageManager.PERMISSION\_GRANTED**

**}**

**}**

**private fun requestPermission() {**

**// Request storage permission if needed (for Android versions below 10)**

**ActivityCompat.requestPermissions(**

**this,**

**arrayOf(Manifest.permission.WRITE\_EXTERNAL\_STORAGE),**

**REQUEST\_CODE**

**)**

**}**

**override fun onRequestPermissionsResult(requestCode: Int, permissions: Array<String>, grantResults: IntArray) {**

**super.onRequestPermissionsResult(requestCode, permissions, grantResults)**

**if (requestCode == REQUEST\_CODE) {**

**if (grantResults.isNotEmpty() && grantResults[0] == PackageManager.PERMISSION\_GRANTED) {**

**Toast.makeText(this, "Permission Granted", Toast.LENGTH\_SHORT).show()**

**} else {**

**Toast.makeText(this, "Permission Denied", Toast.LENGTH\_SHORT).show()**

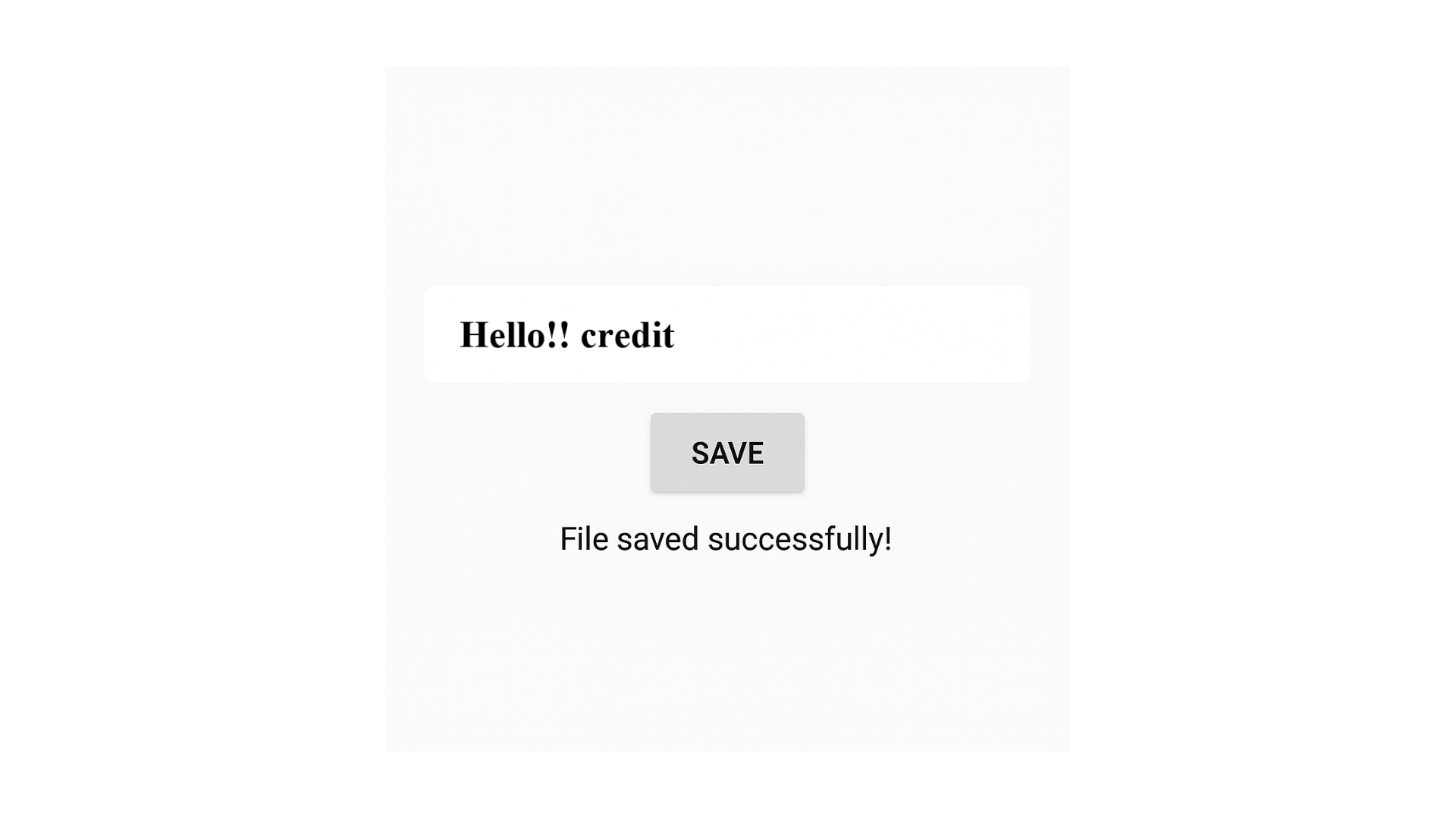
**}**

**}**

**}**

**}**

**OUTPUT:**

****

**Result:**

**Thus,** To build an application using kotlin to write into sd.