Name: Sayed Hamza

Roll no : 107

Subject : Mobile App Development

Semester: 6th (B)

## ANSWER NO: 01

import 'package:flutter/material.dart';

void main() { runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) { return MaterialApp( home: FormScreen(),

);

}

}

class FormScreen extends StatefulWidget {

@override

\_FormScreenState createState() => \_FormScreenState();

}

class \_FormScreenState extends State<FormScreen> { final \_formKey = GlobalKey<FormState>();

// Controllers (optional for further usage like clearing fields) final TextEditingController nameController = TextEditingController(); final TextEditingController emailController = TextEditingController(); final TextEditingController cnicController = TextEditingController(); final TextEditingController phoneController = TextEditingController(); final TextEditingController addressController = TextEditingController(); final TextEditingController passwordController = TextEditingController();

// Validation functions

String? \_validateName(String? value) { if (value == null || value.isEmpty) { return 'Name is required';

} else if (!RegExp(r'^[a-zA-Z]+$').hasMatch(value)) { return 'Name must contain only alphabetic characters';

}

return null;

}

String? \_validateEmail(String? value) { if (value == null || value.isEmpty) {

return 'Email is required';

} else if (!RegExp(r'^[^@]+@[^@]+\.[^@]+').hasMatch(value)) { return 'Enter a valid email';

}

return null;

}

String? \_validateCNIC(String? value) { if (value == null || value.isEmpty) { return 'CNIC is required';

} else if (!RegExp(r'^\d{13}$').hasMatch(value)) { return 'CNIC must be exactly 13 digits';

}

return null;

}

String? \_validatePhone(String? value) { if (value == null || value.isEmpty) { return 'Contact number is required';

} else if (!RegExp(r'^\d{10,12}$').hasMatch(value)) { return 'Contact number must be 10-12 digits';

}

return null;

}

String? \_validateAddress(String? value) { if (value == null || value.isEmpty) {

return 'Address is required';

}

return null;

}

String? \_validatePassword(String? value) { if (value == null || value.isEmpty) { return 'Password is required'; } else if (value.length < 8) {

return 'Password must be at least 8 characters';

} else if (!RegExp(

r'^(?=.\*[a-zA-Z])(?=.\*\d)(?=.\*[@$!%\*?&])[A-Za-z\d@$!%\*?&]+$')

.hasMatch(value)) {

return 'Password must contain letters, numbers, and symbols';

}

return null;

}

// Submit function void \_submitForm() {

if (\_formKey.currentState?.validate() ?? false) {

// Form is valid, proceed with further logic (e.g., send data)

ScaffoldMessenger.of(context).showSnackBar(

SnackBar(content: Text('Form submitted successfully!')),

);

}

}

@override

Widget build(BuildContext context) { return Scaffold(

appBar: AppBar(title: Text('Form Validation Example')), body: Padding( padding: EdgeInsets.all(16.0), child: Form( key: \_formKey, child: ListView( children: [ TextFormField( controller: nameController, decoration: InputDecoration(labelText: 'Name'), validator: \_validateName,

),

TextFormField( controller: emailController, decoration: InputDecoration(labelText: 'Email'), validator: \_validateEmail,

keyboardType: TextInputType.emailAddress,

),

TextFormField( controller: cnicController, decoration: InputDecoration(labelText: 'CNIC'),

validator: \_validateCNIC,

keyboardType: TextInputType.number,

),

TextFormField( controller: phoneController,

decoration: InputDecoration(labelText: 'Contact Number'), validator: \_validatePhone, keyboardType: TextInputType.phone,

),

TextFormField( controller: addressController,

decoration: InputDecoration(labelText: 'Address'), validator: \_validateAddress,

),

TextFormField( controller: passwordController,

decoration: InputDecoration(labelText: 'Password'), obscureText: true, validator: \_validatePassword,

),

SizedBox(height: 20),

ElevatedButton( onPressed: \_submitForm,

child: Text('Submit'),

),

],

),

),

),

);

}

}

## ANSWER NO: 02

dependencies: flutter: sdk: flutter http: ^0.13.5 import 'package:flutter/material.dart';

import'package:http/http.dart' as http; import 'dart:convert';

void main() { runApp(MyApp());

}

class MyApp extends StatelessWidget { @override

Widget build(BuildContext context) { return MaterialApp( title: 'API Search ListView', theme: ThemeData( primarySwatch: Colors.blue,

),

home: PostListScreen(),

);

}

}

class PostListScreen extends StatefulWidget {

@override

\_PostListScreenState createState() => \_PostListScreenState();

}

class \_PostListScreenState extends State<PostListScreen> {

List posts = [];

List filteredPosts = []; bool isLoading = true;

TextEditingController searchController = TextEditingController();

@override void initState() { super.initState(); fetchPosts();

searchController.addListener(() { filterPosts();

});

}

Future<void> fetchPosts() async {

final response = await http.get(Uri.parse('https://jsonplaceholder.typicode.com/posts')); if (response.statusCode == 200) { setState(() {

posts = json.decode(response.body); filteredPosts = posts; isLoading = false;

});

} else {

throw Exception('Failed to load posts');

}

}

void filterPosts() {

List results = posts .where((post) =>

post['title'].toLowerCase().contains(searchController.text.toLowerCase()))

.toList(); setState(() { filteredPosts = results;

});

}

@override

Widget build(BuildContext context) { return Scaffold( appBar: AppBar(

title: Text('API Search ListView'),

),

body: isLoading

? Center(child: CircularProgressIndicator())

: Column( children: [ Padding(

padding: const EdgeInsets.all(8.0), child: TextField( controller: searchController, decoration: InputDecoration( labelText: 'Search', border: OutlineInputBorder(),

),

),

),

Expanded( child: ListView.builder( itemCount: filteredPosts.length, itemBuilder: (context, index) { return ListTile(

title: Text(filteredPosts[index]['title']), subtitle: Text(filteredPosts[index]['body']),

);

},

),

),

],

),

);

}

@override void dispose() { searchController.dispose(); super.dispose();

}

}

## ANSWER NO: 03

import 'package:flutter/material.dart'; import 'package:shared\_preferences/shared\_preferences.dart'; import 'dart:convert';

void main() { runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) { return MaterialApp( home: TodoListScreen(),

);

}

}

class TodoListScreen extends StatefulWidget {

@override

\_TodoListScreenState createState() => \_TodoListScreenState();

}

class \_TodoListScreenState extends State<TodoListScreen> { List<Map<String, dynamic>> tasks = []; final TextEditingController taskController = TextEditingController();

@override void initState() { super.initState();

\_loadTasks();

}

// Load tasks from SharedPreferences

Future<void> \_loadTasks() async {

SharedPreferences prefs = await SharedPreferences.getInstance(); String? tasksString = prefs.getString('tasks'); if (tasksString != null) { setState(() {

tasks = List<Map<String, dynamic>>.from(json.decode(tasksString));

});

}

}

// Save tasks to SharedPreferences

Future<void> \_saveTasks() async {

SharedPreferences prefs = await SharedPreferences.getInstance(); prefs.setString('tasks', json.encode(tasks));

}

// Add a new task void \_addTask(String task) { setState(() { tasks.add({'task': task, 'completed': false});

});

\_saveTasks(); taskController.clear();

}

// Toggle task completion status void \_toggleTask(int index) { setState(() { tasks[index]['completed'] = !tasks[index]['completed'];

});

\_saveTasks();

}

// Delete a task void \_deleteTask(int index) { setState(() { tasks.removeAt(index);

});

\_saveTasks();

}

@override

Widget build(BuildContext context) { return Scaffold( appBar: AppBar(title: Text('To-Do List')), body: Padding( padding: EdgeInsets.all(8.0), child: Column( children: [ TextField( controller: taskController, decoration: InputDecoration( labelText: 'New Task', border: OutlineInputBorder(),

),

onSubmitted: (value) { if (value.isNotEmpty) {

\_addTask(value);

}

},

),

SizedBox(height: 10), Expanded( child: ListView.builder( itemCount: tasks.length, itemBuilder: (context, index) { final task = tasks[index]; return ListTile( title: Text( task['task'], style: TextStyle(

decoration: task['completed'] ? TextDecoration.lineThrough : null,

),

),

leading: Checkbox( value: task['completed'], onChanged: (value) {

\_toggleTask(index);

},

),

trailing: IconButton( icon: Icon(Icons.delete, color: Colors.red), onPressed: () => \_deleteTask(index),

),

);

},

),

),

],

),

),

);

}

}

## ANSWER NO:04

dependencies: flutter: sdk: flutter contacts\_service: ^0.6.1 permission\_handler: ^10.2.0

import 'package:flutter/material.dart'; import 'package:contacts\_service/contacts\_service.dart'; import 'package:permission\_handler/permission\_handler.dart';

void main() { runApp(MyApp());

}

class MyApp extends StatelessWidget {

@override

Widget build(BuildContext context) { return MaterialApp( title: 'Contact List with Search', theme: ThemeData( primarySwatch: Colors.blue,

),

home: ContactListScreen(),

);

}

}

class ContactListScreen extends StatefulWidget {

@override

\_ContactListScreenState createState() => \_ContactListScreenState();

}

class \_ContactListScreenState extends State<ContactListScreen> {

List<Contact> contacts = []; List<Contact> filteredContacts = []; bool isLoading = true;

TextEditingController searchController = TextEditingController();

@override void initState() { super.initState(); requestContactPermission(); searchController.addListener(() { filterContacts();

});

}

Future<void> requestContactPermission() async { var status = await Permission.contacts.status; if (!status.isGranted) {

status = await Permission.contacts.request();

}

if (status.isGranted) { fetchContacts();

} else { setState(() { isLoading = false;

});

}

}

Future<void> fetchContacts() async { final allContacts = await ContactsService.getContacts(); setState(() { contacts = allContacts.toList(); filteredContacts = contacts; isLoading = false;

});

}

void filterContacts() {

List<Contact> results = contacts.where((contact) { final name = contact.displayName?.toLowerCase() ?? ''; final query = searchController.text.toLowerCase(); return name.contains(query);

}).toList(); setState(() { filteredContacts = results;

});

}

@override

Widget build(BuildContext context) { return Scaffold( appBar: AppBar( title: Text('Contacts with Search'),

),

body: isLoading

? Center(child: CircularProgressIndicator())

: Column( children: [ Padding(

padding: const EdgeInsets.all(8.0), child: TextField( controller: searchController, decoration: InputDecoration( labelText: 'Search Contacts', border: OutlineInputBorder(), prefixIcon: Icon(Icons.search),

),

),

),

Expanded( child: ListView.builder( itemCount: filteredContacts.length, itemBuilder: (context, index) { final contact = filteredContacts[index]; return ContactListItem(contact: contact);

},

),

),

],

),

);

}

@override void dispose() { searchController.dispose(); super.dispose();

}

}

class ContactListItem extends StatelessWidget { final Contact contact;

ContactListItem({required this.contact});

@override

Widget build(BuildContext context) {

final phone = contact.phones?.isNotEmpty == true

? contact.phones!.first.value

: 'No phone number';

return Card(

margin: EdgeInsets.symmetric(vertical: 5, horizontal: 10), child: ListTile( leading: CircleAvatar(

child: Text(contact.initials(), style: TextStyle(color: Colors.white)), backgroundColor: Colors.blueAccent,

),

title: Text(contact.displayName ?? 'No Name', style: TextStyle(fontWeight: FontWeight.bold)), subtitle: Text(phone),

),

);

}

}